



*electronics*

# Machine Learning and Embedded Computing in Advanced Driver Assistance Systems (ADAS)

---

Edited by  
John Ball and Bo Tang

Printed Edition of the Special Issue Published in *Electronics*

# **Machine Learning and Embedded Computing in Advanced Driver Assistance Systems (ADAS)**



# Machine Learning and Embedded Computing in Advanced Driver Assistance Systems (ADAS)

Special Issue Editors

**John Ball**

**Bo Tang**

MDPI • Basel • Beijing • Wuhan • Barcelona • Belgrade



*Special Issue Editors*

John Ball  
Mississippi State University,  
USA

Bo Tang  
Mississippi State University,  
USA

*Editorial Office*

MDPI  
St. Alban-Anlage 66  
4052 Basel, Switzerland

This is a reprint of articles from the Special Issue published online in the open access journal *Electronics* (ISSN 2079-9292) from 2018 to 2019 (available at: [https://www.mdpi.com/journal/electronics/special\\_issues/ML\\_EmbeddedComputing\\_ADAS](https://www.mdpi.com/journal/electronics/special_issues/ML_EmbeddedComputing_ADAS))

For citation purposes, cite each article independently as indicated on the article page online and as indicated below:

LastName, A.A.; LastName, B.B.; LastName, C.C. Article Title. <i>Journal Name</i> <b>Year</b> , Article Number, Page Range.
---

**ISBN 978-3-03921-375-7 (Pbk)**

**ISBN 978-3-03921-376-4 (PDF)**

Cover image courtesy of pxhere.com: <https://pxhere.com/en/photo/54410>

© 2019 by the authors. Articles in this book are Open Access and distributed under the Creative Commons Attribution (CC BY) license, which allows users to download, copy and build upon published articles, as long as the author and publisher are properly credited, which ensures maximum dissemination and a wider impact of our publications.

The book as a whole is distributed by MDPI under the terms and conditions of the Creative Commons license CC BY-NC-ND.

# Contents

About the Special Issue Editors . . . . . vii

**John E. Ball and Bo Tang**

Machine Learning and Embedded Computing in Advanced Driver Assistance Systems (ADAS)  
Reprinted from: *Electronics* **2019**, *8*, 748, doi:10.3390/electronics8070748 . . . . . 1

**Yifeng Xu, Huigang Wang, Xing Liu, Henry Ren He, Qingyue Gu and Weita Sun**

Learning to See the Hidden Part of the Vehicle in the Autopilot Scene  
Reprinted from: *Electronics* **2019**, *8*, 331, doi:10.3390/electronics8030331 . . . . . 5

**Yiming Zhao, Lin Bai, Yecheng Lyu and Xinming Huang**

Camera-Based Blind Spot Detection with a General Purpose Lightweight Neural Network  
Reprinted from: *Electronics* **2019**, *8*, 233, doi:10.3390/electronics8020233 . . . . . 21

**Hyeonjeong Lee, Jaewon Lee and Miyoung Shin**

Using Wearable ECG/PPG Sensors for Driver Drowsiness Detection Based on Distinguishable  
Pattern of Recurrence Plots  
Reprinted from: *Electronics* **2019**, *8*, 192, doi:10.3390/electronics8020192 . . . . . 31

**Xinqing Wang, Xia Hua, Feng Xiao, Yuyang Li, Xiaodong Hu and Pengyu Sun**

Multi-Object Detection in Traffic Scenes Based on Improved SSD  
Reprinted from: *Electronics* **2018**, *7*, 302, doi:10.3390/electronics7110302 . . . . . 46

**Jiyoung Jung and Sung-Ho Bae**

Real-Time Road Lane Detection in Urban Areas Using LiDAR Data  
Reprinted from: *Electronics* **2018**, *7*, 276, doi:10.3390/electronics7110276 . . . . . 74

**Yong Li, Guofeng Tong, Huashuai Gao, Yuebin Wang, Liqiang Zhang and Huairong Chen**

Pano-RSOD: A Dataset and Benchmark for Panoramic Road Scene Object Detection  
Reprinted from: *Electronics* **2019**, *8*, 329, doi:10.3390/electronics8030329 . . . . . 88

**Alex Dominguez-Sanchez, Miguel Cazorla and Sergio Orts-Escolano**

A New Dataset and Performance Evaluation of a Region-Based CNN for Urban Object  
Detection  
Reprinted from: *Electronics* **2018**, *7*, 301, doi:10.3390/electronics7110301 . . . . . 110

**Martin Dendaluce Jahnke, Francesco Cosco, Rihards Novickis, Joshué Pérez Rastelli and  
Vicente Gomez-Garay**

Efficient Neural Network Implementations on Parallel Embedded Platforms Applied to  
Real-Time Torque-Vectoring Optimization Using Predictions for Multi-Motor Electric Vehicles  
Reprinted from: *Electronics* **2019**, *8*, 250, doi:10.3390/electronics8020250 . . . . . 129

**Alejandro Said, Yasser Davizón, Rogelio Soto, Carlos Félix-Herrán,  
Carlos Hernández-Santos and Piero Espino-Román**

An Infinite-Norm Algorithm for Joystick Kinematic Control of Two-Wheeled Vehicles  
Reprinted from: *Electronics* **2018**, *7*, 164, doi:10.3390/electronics7090164 . . . . . 156

**Jian Wei and Feng Liu**

Coupled-Region Visual Tracking Formulation Based on a Discriminative Correlation Filter Bank  
Reprinted from: *Electronics* **2018**, *7*, 244, doi:10.3390/electronics7100244 . . . . . 171

**Tiwen Han, Lijia Wang and Binbin Wen**

The Kernel Based Multiple Instances Learning Algorithm for Object Tracking

Reprinted from: *Electronics* **2018**, *7*, 97, doi:10.3390/electronics7060097 . . . . . 191

**Christopher Goodin, Daniel Carruth, Matthew Doude, Christopher Hudson**

Predicting the Influence of Rain on LIDAR in ADAS

Reprinted from: *Electronics* **2019**, *8*, 89, doi:10.3390/electronics8010089 . . . . . 204

**Christopher Goodin, Matthew Doude, Christopher R. Hudson, Daniel W. Carruth**

Enabling Off-Road Autonomous Navigation-Simulation of LIDAR in Dense Vegetation

Reprinted from: *Electronics* **2018**, *7*, 154, doi:10.3390/electronics7090154 . . . . . 213

**Pan Wei, Lucas Cagle, Tasmia Reza, John Ball, Jim Gafford**

LiDAR and Camera Detection Fusion in a Real-Time Industrial Multi-Sensor Collision Avoidance System

Reprinted from: *Electronics* **2018**, *7*, 84, doi:10.3390/electronics7060084 . . . . . 230

**Yaping Liao, Junyou Zhang, Shufeng Wang, Sixian Li and Jian Han**

Study on Crash Injury Severity Prediction of Autonomous Vehicles for Different Emergency Decisions Based on Support Vector Machine Model

Reprinted from: *Electronics* **2018**, *7*, 381, doi:10.3390/electronics7120381 . . . . . 262

**Sixian Li, Junyou Zhang, Shufeng Wang, Pengcheng Li and Yaping Liao**

Ethical and Legal Dilemma of Autonomous Vehicles: Study on Driving Decision-Making Model under the Emergency Situations of Red Light-Running Behaviors

Reprinted from: *Electronics* **2018**, *7*, 264, doi:10.3390/electronics7100264 . . . . . 282

**Felipe Jiménez, José Eugenio Naranjo, Sofía Sánchez, Francisco Serradilla, Elisa Pérez, María José Hernández and Trinidad Ruiz**

Communications and Driver Monitoring Aids for Fostering SAE Level-4 Road Vehicles Automation

Reprinted from: *Electronics* **2018**, *7*, 228, doi:10.3390/electronics7100228 . . . . . 300

**Edgar Talavera, José J. Anaya, Oscar Gómez, Felipe Jiménez and José E. Naranjo**

Performance Comparison of Geobroadcast Strategies for Winding Roads

Reprinted from: *Electronics* **2018**, *7*, 32, doi:10.3390/electronics7030032 . . . . . 318

## About the Special Issue Editors

**John Ball** is an Associate Professor and Robert Guyton Chair of Teaching Excellence at the Department of Electrical and Computer Engineering, Mississippi State University (MSU). Dr. Ball earned his Ph.D. from Mississippi State in 2007. He received his B.S. and Ph.D. in Electrical Engineering at MSU in 1987 and 2007, respectively, and his M.S. in Electrical Engineering from the Georgia Institute of Technology in 1991. He specializes in sensor processing, signal and image processing, and deep learning. His research focuses on a variety of sensors and sensor processing algorithm development, including body sensor networks for sports rehab/prehab, automotive autonomy (camera, thermal, LiDAR, radar), and remote sensing (camera, multispectral, hyperspectral, and radar). He has published over 90 conference and journal papers, technical reports, and training seminars, and has secured almost \$8 million in research funding (2013–2019) from sponsors such as the U.S. Army, TARDEC, AFRL, NSF, NIJ, and several industrial companies. He is a codirector of the Sensor Analysis and Intelligence Lab (SAIL) at Mississippi State’s Center for Advanced Vehicular Systems (CAVS). He has more than 22 years of experience in industry, government, and academic sectors. Dr. Ball is a senior member of IEEE and a member of SAE International, SPIE, and ASEE.

**Bo Tang** is an Assistant Professor at the Department of Electrical and Computer Engineering, Mississippi State University. Dr. Tang received his Ph.D. degree in electrical engineering from the University of Rhode Island (Kingstown, RI) in 2016. From 2016 to 2017, he worked as an Assistant Professor in the Department of Computer Science at Hofstra University, Hempstead, NY. His research interests lie in the general areas of statistical machine learning and data mining, as well as their various applications in cyber–physical systems, including robotics, autonomous driving, and remote sensing.







Editorial

# Machine Learning and Embedded Computing in Advanced Driver Assistance Systems (ADAS)

John E. Ball <sup>\*,†</sup> and Bo Tang <sup>†</sup>

Electrical and Computer Engineering, Mississippi State University, 406 Hardy Road, Mississippi State, MS 39762, USA

\* Correspondence: jeball@ece.msstate.edu; Tel.: +1-662-325-4169

† These authors contributed equally to this work.

Received: 25 June 2019; Accepted: 26 June 2019; Published: 2 July 2019

## 1. Introduction

Advanced driver assistance systems (ADAS) are rapidly being developed for autonomous vehicles. Two driving factors enabling these efforts are machine learning and embedded computing. Advanced machine learning algorithms allow the ADAS system to detect objects, obstacles, other vehicles, pedestrians, and lanes, and also enables the estimation of object trajectories and intents (e.g., this car will change lanes ahead). The Special Issue [1] has 18 high-quality papers covering a diversity of focus areas in ADAS:

1. Communications: [2,3];
2. Object detection and tracking: [4–10];
3. Sensor modeling and simulation: [11,12];
4. Decision-making: [13,14];
5. New datasets: [9,10,15,16];
6. Driver monitoring: [17];
7. New applied hardware for ADAS: [9,18,19].

Some papers fit into multiple categories (e.g., [9,10]). It is also worth noting that three papers were selected as feature papers for the Special Issue:

- “Performance Comparison of Geobroadcast Strategies for Winding Roads” by Talavera et al. [2];
- “LiDAR and Camera Detection Fusion in a Real-Time Industrial Multi-Sensor Collision Avoidance System” by Wei et al. [4]; and
- “A New Dataset and Performance Evaluation of a Region-Based CNN for Urban Object Detection” by Dominguez-Sanchez et al. [15].

## 2. The Present Special Issue

### 2.1. Communications

V2X (vehicle to another vehicle ( $X = V$ ) or infrastructure ( $X = I$ )) communications is a very important part of ADAS, because these communications can improve vehicle safety and alert the autonomous system to potentially dangerous situations.

A V2X communication module was implemented and validated on a close curve in a winding road where poor visibility causes a safety risk [2]. A combination of cooperative systems is proposed to offer a wider range of information to the vehicle than on-board sensors currently provide to help support systems to transition from Society of Automotive Engineers (SAE) levels 2 and 3 to level 4 [3].

## 2.2. Object Detection and Tracking

Object tracking is a critical component in ADAS applications. Objects must be detected and tracked for obstacle avoidance, collision detection, and path planning, to name a few.

A kernel-based multiple instance learning (MIL) tracker was developed that is computationally fast and robust to partial occlusions, pose variations, and illumination changes [5]. To help combat partial object occlusions, a tracking-by-detection framework which uses multiple discriminative correlation filters called discriminative correlation filter bank (DCFB), corresponding to different target sub-regions and global region patches to combine and optimize the final correlation output in the frequency domain, is shown to produce good results compared to state-of-the-art trackers [6].

Object detection is also a critical component. A LiDAR's 3D point cloud was categorized into drivable and non-drivable regions, and an expectation-maximization method was utilized to detect parallel lines and update the 3D line parameters in real time, which allowed the generation of accurate lane-level maps of two complex urban routes [7]. To improve multi-object detection, a detection framework denoted adaptive perceive-single shot multi-box detector (AP-SSD) is proposed, where custom multi-shape Gabor filters to improve low-level object detection, bottleneck-long short term memory (LSTM) is used to refine and propagate the feature mapping between frames, and a dynamic region amplification network framework all work together to achieve better detection results when small objects, multiple objects, cluttered background, and large-area occlusions are present in the scenery [8]. To improve the quality and lower the cost of blind spot detection, a camera-based deep learning method is proposed using a lightweight and computationally efficient neural network. Camera-based methods will be much more cost-effective than using a dedicated radar in this application. In addition, a dataset with more than 10,000 labeled images was generated using a blind spot view camera mounted on a test vehicle [9].

Sensor fusion is a third important area in ADAS units because sensors have different strengths and most experts agree that fusion is required to achieve the best performance. Camera and LiDAR fusion were utilized to make object detection more robust in [4].

## 2.3. Sensor Modeling and Simulation

Many modern machine learning algorithms require significant amounts of training data, which may not be available or may be too expensive and time-consuming to collect.

To aid in LiDAR-based algorithm development, a real-time physics-based LiDAR simulator for densely vegetated environments including an improved statistical model for the range distribution of LiDAR returns in grass was developed and validated [11]. A mathematical model was developed for the performance degradation of LiDAR as a function of rain rate. This model was used to quantitatively evaluate how rain influences a LiDAR-based obstacle-detection system [12].

## 2.4. Decision-Making

Decision systems in ADAS are complicated. They require information analyzed from sensor data, proprioceptive data from the vehicle, and data from other sources (e.g., V2X communications).

To investigate crash severity prediction in emergency decisions, several support vector machine (SVM)-based decision models were analyzed to estimate crash severity prediction involving braking, turning, and braking plus turning actions [14]. Ethical and legal issues in decision systems were analyzed by using a T-S fuzzy neural network that was developed incorporating ethical and legal factors into the driving decision-making model under emergency situations evoked by red-light-running behaviors [13].

## 2.5. New Datasets

A critical aspect of developing and testing deep learning systems is the availability of high-quality datasets for algorithm training and testing.

A new dataset which includes all of the essential urban objects was collected, including weakly annotated data for training and testing weakly supervised learning techniques. Furthermore, a faster region-based convolutional neural networks (R-CNN) was evaluated using this dataset and a new R-CNN plus tracking technique to accelerate the process of real-time urban object detection was developed and evaluated [15]. A blind spot detection dataset is introduced in [9]. Refer to Section 2.2 for more information, as this paper belongs in both categories. A new benchmark dataset named Pano-RSOD was created for 360° panoramic road scene object detection. The dataset contains vehicles, pedestrians, traffic signs and guiding arrows, small objects, and imagery from diverse road scenes. Furthermore, the usefulness of the dataset was demonstrated by training state-of-the-art deep-learning algorithms for object detection in panoramic imagery [16].

### 2.6. Driver Monitoring

As ADAS levels are not yet at full autonomy (level 5), driver monitoring is critical to safety.

To investigate robust and distinguishable patterns of heart rate variability, wearable electrocardiogram (ECG) or photoplethysmogram (PPG) sensors were utilized to generate recurrence plots, which were then analyzed by a CNN to detect drowsy drivers. The proposed method showed significant improvement over conventional models [17].

### 2.7. New Applied Hardware for ADAS

An algorithm based on the mathematical  $p$ -norm was developed which improved both the traction power and the trajectory smoothness of joystick-controlled two-wheeled vehicles, such as tanks and wheelchairs [18].

To address challenges and issues in the challenging area of torque vectoring on multi-electric-motor vehicles for enhanced vehicle dynamics, a neural network is proposed for batch predictions for real-time optimization on a parallel embedded platform with a GPU and an FPGA. This work will help others who are conducting research in this technical area [19].

## 3. Concluding Remarks

The Guest Editors were pleased with the quality and breadth of the accepted papers. We were also delighted to have three papers with high-quality and very useful new datasets [9,15,16]. Looking to the future, we believe all research works enclosed in this Special Issue will promote further study in the area of ADAS.

**Author Contributions:** The authors worked together and contributed equally during the editorial process of this Special Issue.

**Funding:** This research received no external funding.

**Acknowledgments:** The Guest Editors thank all of the authors for their excellent contributions to this Special Issue. We also thank the reviewers for their dedication and suggestions to improve each of the papers. We finally thank the Editorial Board of MDPI's *Electronics* for allowing us to be Guest Editors for this Special Issue, and to the *Electronics* Editorial Office for their guidance, dedication, and support.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Electronics Special Issue: Machine Learning and Embedded Computing in Advanced Driver Assistance Systems (ADAS), 2019. Available online: [https://www.mdpi.com/journal/electronics/special\\_issues/ML\\_EmbeddedComputing\\_ADAS](https://www.mdpi.com/journal/electronics/special_issues/ML_EmbeddedComputing_ADAS) (accessed on 25 June 2019).
2. Talavera, E.; Anaya, J.J.; Gómez, O.; Jiménez, F.; Naranjo, J.E. Performance Comparison of Geobroadcast Strategies for Winding Roads. *Electronics* **2018**, *7*, 32. [CrossRef]

3. Jiménez, F.; Naranjo, J.E.; Sánchez, S.; Serradilla, F.; Pérez, E.; Hernández, M.J.; Ruiz, T. Communications and Driver Monitoring Aids for Fostering SAE Level-4 Road Vehicles Automation. *Electronics* **2018**, *7*, 228.10.3390/electronics7100228. [[CrossRef](#)]
4. Wei, P.; Cagle, L.; Reza, T.; Ball, J.; Gafford, J. LiDAR and Camera Detection Fusion in a Real-Time Industrial Multi-Sensor Collision Avoidance System. *Electronics* **2018**, *7*, 84.10.3390/electronics7060084. [[CrossRef](#)]
5. Han, T.; Wang, L.; Wen, B. The Kernel Based Multiple Instances Learning Algorithm for Object Tracking. *Electronics* **2018**, *7*, 97.10.3390/electronics7060097. [[CrossRef](#)]
6. Wei, J.; Liu, F. Coupled-Region Visual Tracking Formulation Based on a Discriminative Correlation Filter Bank. *Electronics* **2018**, *7*, 244.10.3390/electronics7100244. [[CrossRef](#)]
7. Jung, J.; Bae, S.H. Real-Time Road Lane Detection in Urban Areas Using LiDAR Data. *Electronics* **2018**, *7*, 276.10.3390/electronics7110276. [[CrossRef](#)]
8. Wang, X.; Hua, X.; Xiao, F.; Li, Y.; Hu, X.; Sun, P. Multi-Object Detection in Traffic Scenes Based on Improved SSD. *Electronics* **2018**, *7*, 302.10.3390/electronics7110302. [[CrossRef](#)]
9. Zhao, Y.; Bai, L.; Lyu, Y.; Huang, X. Camera-Based Blind Spot Detection with a General Purpose Lightweight Neural Network. *Electronics* **2019**, *8*, 233.10.3390/electronics8020233. [[CrossRef](#)]
10. Xu, Y.; Wang, H.; Liu, X.; He, H.R.; Gu, Q.; Sun, W. Learning to See the Hidden Part of the Vehicle in the Autopilot Scene. *Electronics* **2019**, *8*, 331.10.3390/electronics8030331. [[CrossRef](#)]
11. Goodin, C.; Doude, M.; Hudson, C.R.; Carruth, D.W. Enabling Off-Road Autonomous Navigation-Simulation of LIDAR in Dense Vegetation. *Electronics* **2018**, *7*, 154.10.3390/electronics7090154. [[CrossRef](#)]
12. Goodin, C.; Carruth, D.; Doude, M.; Hudson, C. Predicting the Influence of Rain on LIDAR in ADAS. *Electronics* **2019**, *8*, 89.10.3390/electronics8010089. [[CrossRef](#)]
13. Li, S.; Zhang, J.; Wang, S.; Li, P.; Liao, Y. Ethical and Legal Dilemma of Autonomous Vehicles: Study on Driving Decision-Making Model under the Emergency Situations of Red Light-Running Behaviors. *Electronics* **2018**, *7*, 264.10.3390/electronics7100264. [[CrossRef](#)]
14. Liao, Y.; Zhang, J.; Wang, S.; Li, S.; Han, J. Study on Crash Injury Severity Prediction of Autonomous Vehicles for Different Emergency Decisions Based on Support Vector Machine Model. *Electronics* **2018**, *7*, 381.10.3390/electronics7120381. [[CrossRef](#)]
15. Dominguez-Sanchez, A.; Cazorla, M.; Orts-Escolano, S. A New Dataset and Performance Evaluation of a Region-Based CNN for Urban Object Detection. *Electronics* **2018**, *7*, 301.10.3390/electronics7110301. [[CrossRef](#)]
16. Li, Y.; Tong, G.; Gao, H.; Wang, Y.; Zhang, L.; Chen, H. Pano-RSOD: A Dataset and Benchmark for Panoramic Road Scene Object Detection. *Electronics* **2019**, *8*, 329.10.3390/electronics8030329. [[CrossRef](#)]
17. Lee, H.; Lee, J.; Shin, M. Using Wearable ECG/PPG Sensors for Driver Drowsiness Detection Based on Distinguishable Pattern of Recurrence Plots. *Electronics* **2019**, *8*, 192.10.3390/electronics8020192. [[CrossRef](#)]
18. Said, A.; Davizón, Y.; Soto, R.; Félix-Herrán, C.; Hernández-Santos, C.; Espino-Román, P. An Infinite-Norm Algorithm for Joystick Kinematic Control of Two-Wheeled Vehicles. *Electronics* **2018**, *7*, 164.10.3390/electronics7090164. [[CrossRef](#)]
19. Dendaluce Jahnke, M.; Cosco, F.; Novickis, R.; Pérez Rastelli, J.; Gomez-Garay, V. Efficient Neural Network Implementations on Parallel Embedded Platforms Applied to Real-Time Torque-Vectoring Optimization Using Predictions for Multi-Motor Electric Vehicles. *Electronics* **2019**, *8*, 250.10.3390/electronics8020250. [[CrossRef](#)]



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

Article

# Learning to See the Hidden Part of the Vehicle in the Autopilot Scene

Yifeng Xu <sup>1</sup>, Huigang Wang <sup>1,\*</sup>, Xing Liu <sup>1</sup>, Henry Ren He <sup>2</sup>, Qingyue Gu <sup>1</sup> and Weitao Sun <sup>1</sup>

<sup>1</sup> School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an 710072, China; xuyifeng123@mail.nwpu.edu.cn (Y.X.); xingliu86@nwpu.edu.cn (X.L.); guqingyue@mail.nwpu.edu.cn (Q.G.); Sunwt1223@gmail.com (W.S.)

<sup>2</sup> Center for Business, Information Technology and Enterprise, Waikato Institute of Technology, Hamilton 3240, New Zealand; henry.ren@wintec.ac.nz

\* Correspondence: wanghg74@nwpu.edu.cn; Tel.: +86-029-8846-0521

Received: 31 December 2018; Accepted: 11 March 2019; Published: 18 March 2019

**Abstract:** Recent advances in deep learning have shown exciting promise in low-level artificial intelligence tasks such as image classification, speech recognition, object detection, and semantic segmentation, etc. Artificial intelligence has made an important contribution to autopilot, which is a complex high-level intelligence task. However, the real autopilot scene is quite complicated. The first accident of autopilot occurred in 2016. It resulted in a fatal crash where the white side of a vehicle appeared similar to a brightly lit sky. The root of the problem is that the autopilot vision system cannot identify the part of a vehicle when the part is similar to the background. A method called DIDA was first proposed based on the deep learning network to see the hidden part. DIDA cascades the following steps: object detection, scaling, image inpainting assuming a hidden part beside the car, object re-detection from inpainted image, zooming back to the original size, and setting an alarm region by comparing two detected regions. DIDA was tested in a similar scene and achieved exciting results. This method solves the aforementioned problem only by using optical signals. Additionally, the vehicle dataset captured in Xi'an, China can be used in subsequent research.

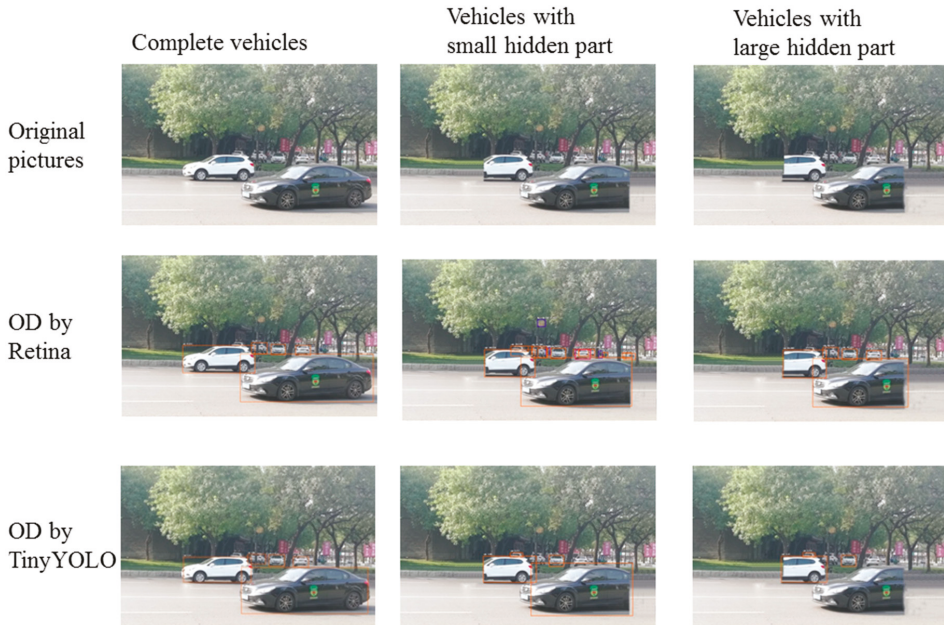
**Keywords:** driverless; autopilot; deep learning; object detection; generative adversarial nets; image inpainting

## 1. Introduction

A fatal crash occurred in a Tesla Model S on 5 July 2016. This was the first known fatality in autopilot on an autonomous vehicle. Tesla noted that "the vehicle was on a divided highway with Autopilot engaged when a tractor trailer drove across the highway perpendicular to the Model S. Neither Autopilot nor the driver noticed the white side of the tractor trailer against a brightly lit sky, so the brake was not applied." [1]. There are three sensor systems in Tesla Autopilot. The first one is the vision system named MobileEyeQ3 in the middle of the windshield. The second is the millimeter-wave radar below the front bumper. The last one is twelve ultrasonic sensors around the vehicle. All sensors unfortunately missed the back part during the unusual situation in which the fatality occurred. The measuring distance of the ultrasonic radar was too short (2 m). It could not detect obstructions at high speeds. The installed position of the millimeter-wave radar system was too low, and its vertical angle was less than five degrees, thus this sensor missed the object. Optical detection ideally emulates how human observes the world and therefore should be the most efficient method. However, it is difficult for the camera to extract features of white objects from a large white background. It cannot get enough feature information to input the MobileEyeQ3. This results in the MobileEyeQ3 missing the object. Essentially, the problem is caused by less redundancy of the image algorithm.

In addition, if a large part of the vehicle is similar to blue sky, buildings, or trees, the computer vision system of the self-driving car could miss detecting the part.

The left picture in the first row of Figure 1 was taken from a real scene. According to the Tesla accident report [1], the extreme virtual environment was settled. In this virtual environment, two virtual cars are shown in the middle and right of the first row. Let us postulate that the drivers paint the back of the black car and the front of the white SUV in a color and style that match completely with the background. Thus, the painted parts blend into the background and cannot be seen but exist nonetheless. This type of vehicle is rare in the real world and looks ridiculous. However, they cannot be excluded.



**Figure 1.** The photos demonstrate that the state of the art object detection algorithms cannot “see” the hidden part of the vehicles.

In this particular case, almost all of the object detection algorithms would fail. The second and third row respectively show the results processed by the latest object detection algorithms, Retina [2] and TinyYOLO [3]. In Figure 1, the abbreviation “OD” stands for “object detection”. Both algorithms failed to detect the entire vehicle in columns 2 and 3. Figure 1 demonstrates that the state of the art object detection algorithms cannot detect the hidden part. If these special vehicles are encountered in the real world, an aforementioned accident will still occur.

The research tries to solve the above problem by deep learning networks. Recently, with the successful application of machine learning, deep learning is becoming more effective in various fields. The researchers use deep neural networks for image classification [4–7], semantic segmentation [8,9], object detection [2,3,10], image generation [11], etc. We propose a novel method to detect the hidden part of a vehicle, thus helping to avoid traffic accidents. In other words, we learn to “see” the hidden part.

Our proposed algorithm tried to correctly handle the situation in the extreme condition shown in Figure 1. Firstly, the deep learning network model needed to be trained on an enormous number of images to achieve satisfactory performance. We collected vehicle pictures by the camera in the busiest

streets of the ancient city, Xi'an, China. In addition to our database, we prepared the other databases described in detail in Section 4.1.

After the databases were prepared, an improved Retina object detection model was trained for detecting the vehicles. In the unmanned vehicle scene, quickly identifying objects is necessary. Recently, real-time object detection methods such as Retina network [2] and YOLO network [3,10] have been proposed. To increase the performance of detecting speed and keep the safety level, a detection model with high threshold value was created. This model is used in the first stage of object detection. The proposed algorithm does not know whether or not the front vehicles have the hidden part beforehand. There is no signal "telling" the algorithm whether or not the vehicle has hidden parts. A reasonable assumption is made that all vehicles may have hidden parts. For reducing the computational work, the main vehicles in the middle of the road are detected by this model. The vehicles that occur further in the distance are not likely to cause accidents and can be ignored. An image inpainting model for inpainting the vehicles was created. Assuming the detected vehicles have hidden parts that cannot be seen by a general computer vision system, inpainting the right and left area is applied separately. When the inpainting algorithm is applied to a complete vehicle without any hidden part, inpainting does not add anything. Otherwise, the area of the inpainted vehicle increases. Then, a second stage of object detection is immediately applied to the inpainted vehicle with the same improved Retina model. The difference between the first and second stage of detected regions is defined as an alarm box. This time, the hidden part can be "seen". The alarm box is the existing region that the normal visual system cannot detect. Vehicles should avoid the alarm box region.

The whole procedure is called the DIDA approach (object detection, inpainting, object re-detection, setting alarm box). We show experimentally that the proposed DIDA can "see" the part that cannot be detected by typical optical systems in Section 3.

Our contributions in this paper are summarized as follows: (1) we proposed the DIDA approach to solve the problem of detecting the incomplete object in the autopilot scene; (2) we proposed the DIDA approach to "see" the hidden part of the vehicle using the serial joint processing of object detection and image inpainting, solving the aforementioned problem with only optical sensor signals for the first time and offering a new method for insuring further security in the autopilot system; (3) we collected the vehicle database in Xi'an, China, which can be used in subsequent research as well.

The rest of the paper is organized as follows. Related work on object detection and image inpainting is proposed in Section 2. Methods such as the flow chart about seeing hidden parts of the vehicle, the framework of the object detection model, and Generative Adversarial Nets (GANs) [12] are shown in Section 3. Detailed experimental results are shown in Section 4. Several problems that need to be further studied are addressed in Section 5. The conclusions are described in Section 6.

## 2. Related Work

This section provides an overview of research on object detection and image inpainting.

First of all, the research of object detection based on deep learning is reviewed. Each object detection model has essentially two phases—an object localization phase to search for candidate objects and a classification phase where the candidates are classified based on distinctive features. In the Region-based-Convolutional-Neural-Network-features (R-CNN) model [13], the selective search method (SS) [14] is used as an alternative to exhaustively searching for object localization. In particular, about 2000 small region proposals are generated and then merged in a bottom-up manner to obtain more accurate candidate regions. Different color-space features, saliency cues, and similarity metrics are used to guide the merging procedure. Each proposed region is then resized to match the input of the CNN model. The output of this model is a 4096-dimensional feature vector. To get the class probabilities, this generated feature vector needs to be classified by multiple Support Vector Machine (SVM) [15]. The R-CNN model has achieved a 62.4% mean Average Precision (mAP) score on the PASCAL VOC 2012 [16] benchmark and a 31.4% mAP score on the 2013 ImageNet [17] benchmark in the large scale visual recognition challenge (ILSVRC).



The major limitation of this method is that it requires a long time to analyze the proposed regions. For addressing the weakness of the time-consuming analysis process, R. Girshick introduced the Fast Region-based Convolutional Network (Fast R-CNN) [18]. The Fast R-CNN method produces output vectors that are further classified by a normalized exponential function (softmax) [19] classifier. Experimental results proved that Fast R-CNNs were capable of achieving a mAP score of 70.0% and 68.4% on the 2007 and 2012 PASCAL VOC benchmarks, respectively.

To address the limitation of high computational overhead in the prior region based methods using selective search, the Region Proposal Network (RPN) [20] were proposed, which produce region proposals directly by determine bounding boxes and detecting objects. This method led to the development of the Faster Region-based Convolutional Network (Faster R-CNN) [20] as a combination of the RPN and the Fast R-CNN models. Experimental results of Faster R-CNN proved an improvement by reporting the mAP scores of 78.8% and 75.9% on the 2007 and 2012 PASCAL VOC benchmarks, respectively. The results of Faster R-CNN are computed 34 times faster than the original Fast R-CNN.

The two models mentioned above involve detection of region proposals and finding an object in the image. However, the Region-based Fully Convolutional Network (R-FCN) [21] uses only convolutional back-propagation layers for learning and inference. The R-FCN reached an 83.6% mAP score on the 2007 PASCAL VOC benchmark. For the 2015 COCO [22] challenge, this method reached a 53.2% score for an Intersection over Union (IoU) = 0.5 and a 31.5% score for the official mAP metric. In terms of speed, the R-FCN is typically 2.5–20 times quicker than the Faster R-CNN.

The you-only-look-once (YOLO) model [3,10] was proposed for determining bounding boxes and class probabilities directly with a network in one run. This method reported mAP scores of 63.7% and 57.9% on the 2007 and 2012 PASCAL VOC benchmarks, respectively. The Fast YOLO model [23] had a lower score (52.7% mAP) in comparison to YOLO. However, it resulted in improved performance of 155 FPS in contrast to 45 FPS for YOLO in a real-time world.

As the YOLO model struggled with the detection of objects of small sizes and unusual aspect ratios, the Single-Shot Detector (SSD) [24] was developed to predict both the bounding boxes and the class probabilities with an end-to-end CNN architecture. The SSD model employs additional differential feature layers ( $10 \times 10$ ,  $5 \times 5$ , and  $3 \times 3$ ) with the aim of improving the number of relevant bounding boxes in comparison to YOLO.

Recently, the two-stage and one-stage detectors based on deep learning started to dominate modern object detection methods. The two-stage detectors include the first stage, generating a sparse set of candidate proposals, and the second stage, classifying the proposed region into the classes based on the background or foreground. The classic two-stage methods contain R-CNN [25], RPN, Fast R-CNN, Faster R-CNN, Feature Pyramid (FPN) [26], etc. The one-stage detectors include the OverFeat [27], SSD [24], and YOLO [3,10]. Generally, the latter have advantages in speed but less accuracy. YOLO focuses on the trade-off between the speed and accuracy. Recently, the Focal Loss [2] was proposed to train a sparse set of hard-detect samples and prevent the large number of easy-detect samples from overwhelming the detector. The Retina Network detector [2] outperforms all previous one-stage and two-stage detectors in both speed and accuracy except for YOLOv3 in speed.

In this work, seeing the hidden part can be considered a problem of detecting the insufficient and incomplete object. This problem is a special case in object detection. Most object detection algorithms are not specifically optimized for detecting the unusual object. The state of the art object detection algorithms such as YOLO and Retina still cannot solve this problem.

The second aspect of related work is the image inpainting. It is another main difficulty to overcome. The image inpainting can be considered as filling the missing parts in a picture. Existing methods addressing this problem fall into two groups. The first uses the traditional diffusion-based or patch-based methods. The second group is based on deep learning networks, such as CNN and GAN [12]. Traditional inpainting approaches [28,29] normally use variational algorithms or patch similarity to generate the hole information. These approaches work well for fixed textures. However,

they are weak in repairing multi-class images with holes. Recently, GANs based on deep learning have emerged as promising methods for image inpainting. Context Encoders [30] firstly train deep neural networks for image inpainting. They are trained with both  $\ell_2$  reconstruction loss and generative adversarial loss (combined adversarial losses function). However, the inpainted texture with Context Encoders appears insufficient. Besides the combined adversarial losses function, both global and local discriminators [31] are proposed for increasing receptive fields of output neurons. Considering the high-resolution image, the Multi-Scale Neural Patch Synthesis [32] was proposed based on the joint loss function, which contains three items—the holistic content constraint, the local texture constraint, and the total variation loss term. This approach shows that promising results not only preserve contextual structures but also generate high-frequency details. One downside of this method is that it is relatively slow. Recently, the method with contextual attention was proposed in [33]. This method uses the global and local WGANs [34] and spatially discounted reconstruction loss to improve the training stability and speed.

### 3. The Approach

#### 3.1. The Flow Chart of the Method

A typical computer vision system cannot detect a whole vehicle that has a large area of color or texture similar to the background. To solve the problem, the DIDA method is proposed. Figure 2 explains the DIDA method in an abstract scenario. The four character “DIDA” comes from the four important steps in the flow chart: object **D**etection (step 1), **I**npainting (step 3), object re-**D**etection (step 4), and setting **A**larm box (step 6). The single object detection algorithm cannot detect the hidden part, nor can the state of the art algorithms. After the first stage of object detection, the inpainting operation is performed. If the vehicle has a hidden part, the inpainting operation will add a part. Then, the second stage of object detection can detect the entire region including the hidden part.

Step 1 is the first stage of vehicle detection. In the optical system of an autopilot, 24–30 frames are captured in one second. Recently, proposed object detection algorithms have been implemented in real-time detection. To detect the hidden part of the vehicle in this work, the system does not need to detect all vehicles. This is due to two reasons; one is that detecting many objects consumes a great deal of computational resource, and the other is that the small vehicles in view far from the main road do not affect the safe driving. In the top picture of Figure 2, two major vehicles are detected and marked in rectangles. The left one is a complete vehicle. The right one is incomplete. The right part of the incomplete vehicle cannot be seen because it has the same color as the background. In order to describe the situation visually, the right half of the vehicle merges in the background.

The second step is padding and image scaling. If the shape of the input data is square, the subsequent convolution network should have high performance. For high performance and retaining certain background information, the detected vehicle image is padded from side, upper, and down directions. Padding is detailed in the experiments section. Additionally, high-resolution image inpainting [32] was proposed, but it uses too many resources. The DIDA focuses on hidden parts and does not need to generate the texture of the original high-resolution image. Therefore, the second step of DIDA reduces the image size after the first stage of object detection. This significantly increases the performance. The scaling operation is both reasonable and necessary.

The third step is image inpainting. The traditional image inpainting methods rely on blurring or copying patches from the side region. It cannot conduct the task of repairing the image to a satisfactory level of completeness. It can only fill the patched holes with part of the background, such as the building, trees, blue sky, white cloud, the road, etc. As of late, image inpainting algorithms based on deep learning have been able to inpaint the hole in a meaningful way. For example, if one eye is covered, the traditional method would fill the covered part with the texture of the skin. The deep learning method, however, would inpaint the covered part with a virtual eye. In the above steps, two major vehicles are detected. One vehicle is complete, while the other only shows half. The latter is

dangerous. Autopilot systems cannot judge the actual size of the vehicles only by an optical system. In the DIDA system, all detected vehicles are supposed to be uncompleted. Therefore, the holes that need to be patched are added to the left and right of the object. If the vehicle is complete, the inpainted image fragment should not be part of the vehicle but the background of the scene. In the other case, the inpainted image fragment of the hole should be a part of the vehicle. The inpainting result is shown as the dotted box tagged as “Added part” in step 3 of Figure 2.

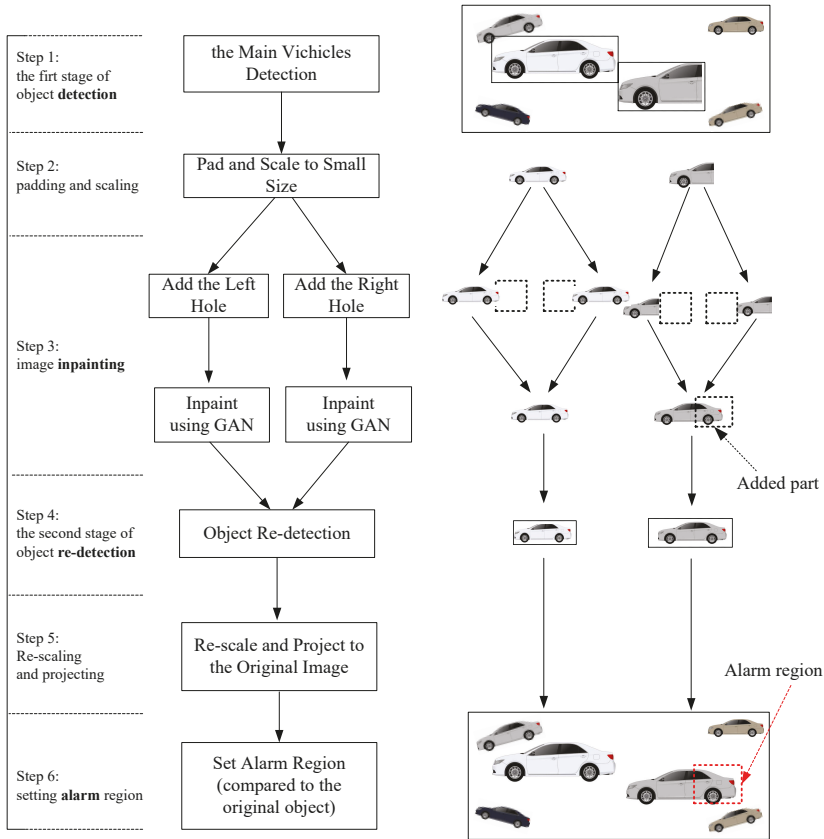


Figure 2. The flow chart demonstrates how to fill in a missing part.

The fourth step is the second stage of object detection. The inpainted images may be the same as the original image or an additional part of the vehicle added to the image. If the detected vehicle image becomes larger than the image before inpainting, this step should find the larger object box. Then, this step outputs a flag that indicates whether or not there is an alarm region.

The fifth step is re-scaling the image to the size of the original image and projecting it to the initial location.

The sixth step is setting the alarm region. If the flag in step 4 is true, we can calculate the alarm region to prevent the vehicle from getting through it.

### 3.2. The Object Detection about Vehicle

Vehicle detection plays an important role in intelligent transportation and autopilot. Considering the speed and accuracy, we adopt the improved Retina network for object detection. The original

Retina Network is a unified network composed of a primary backbone network and two task-specific subnetworks [2].

The backbone network is utilized to compute a convolutional feature map over an input image. The primary network is based on the Feature Pyramid network [26] built on the ResNet network [7].

The first classification subnet predicts the presence probability of an object at each spatial position for each of the anchors and K object classes (K equals 1 here because there is only one vehicle object). It takes an input feature map from the previous primary network's output. The subnet applies four  $3 \times 3$  convolutional layers, each layer followed by ReLU activations. Finally, sigmoid activations are attached to the outputs. Focal loss is adopted as the loss function.

The second subnet performs bounding box regression. It is similar to the classification network, but the parameters are not shared. This subnet outputs the object's location as opposed to the anchor box if an object exists. The smooth\_L1\_loss with sigma that equals three is applied as the loss function to this sub-network.

The focal loss [2] is designed to address the detection scenario in which there is an imbalance between foreground and background classes. The focal loss is shown as follows:

$$FL(p_t) = -\alpha_t(1 - p_t)^\gamma \log(p_t) \quad (1)$$

In Equation (1), gamma is the focusing parameter and alpha is the balancing parameter. The focal loss adds less weight to well classified examples and greater weight to misclassified examples.

To improve the detecting quality and speed, the following adjustments are used. First of all, more training data are added to improve the quality of detection. More vehicle images are extracted from the CompCars dataset [35], the Stanford Cars dataset [36], and the pictures captured by our team. It is described in detail in Section 4. Secondly, the detection threshold is set to 0.8. Thirdly, the class of the detecting object is set to one. Finally, because FPN makes multi-scale predictions, it is not required to detect small objects in this solution. The two minimum layers in the FPN [26] are deleted.

### 3.3. Generate Adversarial Network

Recently, the GAN [11] has been used to generate virtual images. GAN contains two sub-models, the generator model (abbreviated as G) and the discriminator model (abbreviated as D). G generates virtual pictures, which will look like real data. Based on experience, random white noise variable z is defined as the input of G. Then, function  $G(z; \theta_g)$  maps the noise variables to the data space. It is represented by a multilayer perceptron with parameters  $\theta_g$ . The sub-model D is also a multilayer perceptron.  $D(x)$  represents the probability that the picture x comes from the true data rather than the virtually generated data. The framework of G and D is indicated by Figure 3.

The model D outputs a Boolean value of one or zero, which respectively indicates real data or virtually generated data if the output data trick the discriminator into thinking the data are real. GAN trains D to maximize the probability of assigning the correct label to both the true and the virtual data. GAN concurrently trains G to minimize the same probability. In other words, GANs follow a two-player mini-max optimization process written as formula  $\min_G \max_D (D(x) - D(G(z)))$ . In order to make the calculation easy, the log function is added to D. Because the input number for log function must be greater than zero, the improved formula of GAN is changed, as shown in Equation (2):

$$\max_{\theta_D} V(D, G) = \min_G \max_D \{E_{x \sim p_D} [\log D(x)] + E_{x \sim p_G} [\log(1 - D(G(z)))]\}. \quad (2)$$

Minibatch stochastic gradient descent [37] is used in the training stage of GAN. The hyper parameter k is set as two. The k defines the times of the D for each G. The algorithm of minibatch stochastic gradient descent in the experiment is shown as Figure 4.

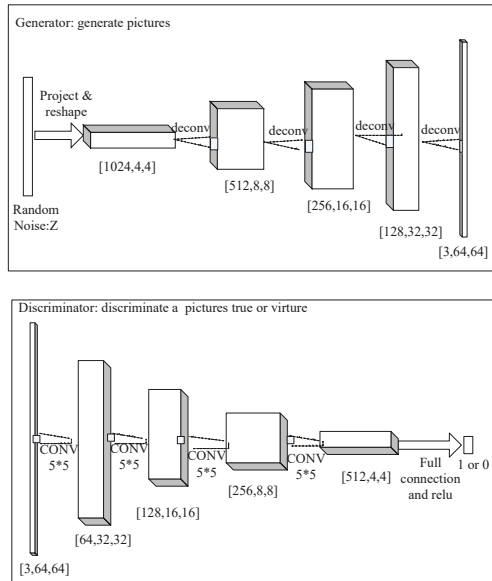


Figure 3. The framework of the generator and discriminator model in Generate Adversarial Network.

```

for times of training iterations do
  for k do
    Sample minibatch of noise samples  $\{z^{(1)}, \dots, z^{(m)}\}$  from noise distribution
    Sample minibatch of examples  $\{x^{(1)}, \dots, x^{(m)}\}$  from generating data distribution
    Update the discriminator according to the formula:
    
$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m [\log D(x^{(i)}) + \log(1 - D(G(z^{(i)})))]$$

  end for
  Sample minibatch of noise samples  $\{z^{(1)}, \dots, z^{(m)}\}$ 
  Update the generator according to the formula:  $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log(1 - D(G(z^{(i)})))$ 
end for

```

Figure 4. The algorithm of minibatch stochastic gradient descent.

The pseudo-code surrounded by the dotted rectangle achieves the optimization of the  $D$ . All of the code completes synchronal optimization of the  $G$  and  $D$ . If the various pictures are all trained at the same time, the diversity of the image class results in chaotic and meaningless virtual images. If there are numerous kinds of images, the generating process should be done category by category. In this case, only one image category (vehicle) is used.

### 3.4. The Framework of Image Inpainting

The architecture of the image inpainting to predict missing parts from their surroundings is shown in Figure 5.

Figure 5 shows the process of inpainting the right predicting region in the white box. The upper half of the figure describes the process of generating the virtue image using the auto-encoding structure [38]. The rest of the figure describes the decision process by the discriminator of GAN.

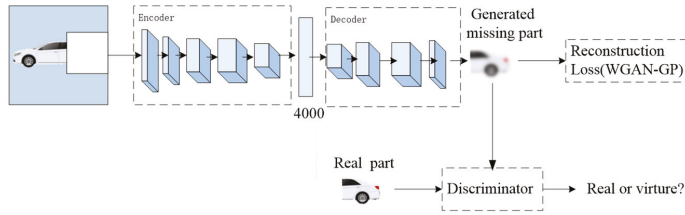


Figure 5. The architecture of the image inpainting network by GAN.

**Encoder:** The encoder is derived from the AlexNet architecture [4]. Given an input image with a size of  $128 \times 128$ , we use five convolutional layers ([64,4,4], [64,4,4], [128,4,4], [256,4,4], [512,4,4]) and the following pooling layer to compute an abstract [512,4,4] dimensional feature representation. In contrast to AlexNet, our model is not trained for ImageNet classification but for predicting the missing hole.

The layer between the encoder and the decoder is a channel-wise fully-connected layer [30]. This layer is designed to propagate information within each feature map. If the dimension of the input layer is  $[m, n, n]$ , this layer outputs the same size feature maps.

**Decoder:** The decoder generates the virtue image with the missing region using the features of the encoder output. The decoder contains a series of up-convolutional layers ([512,4,4], [256,4,4], [128,4,4], [64,4,4], [64,4,4]) [8,39,40] and ELUs activation function [41]. An up-convolutional is utilized to result in a higher resolution image. The idea behind the decoder is that the series of up-convolutions and nonlinear activation function comprises a non-linear up-sampling of the feature produced by the encoder.

**Joint Loss Function:** The reconstruction loss function is set as the following joint loss function:

$$L_{rec}(x) = \|h(x, R) - h(x_i, R)\|_2^2 + \alpha\gamma(x). \quad (3)$$

Given the input image  $x_0$ , we would like to find the unknown output image  $x$ .  $R$  is used to denote the missing hole region in  $x$ . The function  $h(\cdot)$  defines the operation of extracting a sub-image or sub-feature-map in a rectangular region, i.e.,  $h(x, R)$  returns the color content of  $x$  in  $R$ . The term  $\gamma(x)$  represents the total variation regularization to smooth the image:

$$\gamma(x) = \sum_{i,j} ((x_{i,j+1} - x_{i,j})^2 + (x_{i+1,j} - x_{i,j})^2). \quad (4)$$

Empirically,  $\alpha$  is set as  $5 \times 10^{-6}$  to balance the two losses. In this loss Equation (4), the texture loss is not adopted because each texture process should consume more than eight seconds [32].

**Discriminator:** The structure of the discriminator is the same as in Figure 2. It decides whether an image is true or not.

#### 4. Experiments and Results

This section first presents the datasets, then real world vehicle detection and inpainting are demonstrated to show the hidden part of a vehicle.

##### 4.1. Datasets

One of the reasons for the great progress in deep learning is big data. The proposed approach is based on deep learning, thus sufficient valid data are also essential for this approach to function correctly. We prepared the following datasets.

The Comprehensive Cars (CompCars) dataset [35] contains data from two scenarios, web-based and surveillance-based. The web-based data contain 136,726 images capturing entire car sets and

images capturing the car parts. The surveillance-based data contain car images captured in the front view. In this work, only cars from the web-based set are used. The Stanford Cars dataset [36] contains 16,185 images of 196 classes of cars. The data are divided into 8144 training images and 8041 testing images. This work uses both training and testing images as training data. However, the aforementioned car datasets do not have enough vehicle images captured from the sides. Therefore, our team took about 50,000 vehicle pictures. These pictures were collected from the sides of the cars beside the busiest crossroad, the ancient Bell Town and Drum Town, in Xi'an, China. The new dataset was created and called the Complex Traffic Environment (CTE-Cars). The training dataset used in this work is a combination of the CompCars dataset, the Stanford Cars dataset, and CTE-Cars.

#### 4.2. The First Stage of Object Detection (Step 1)

The test dataset contains pictures taken in a new scene. They do not belong to the training dataset. In this way, the set of the experiments ensures the validity of the test results and avoids overfitting.

The focal loss [2] is applied to all anchors [19] in each picture during the training process. The total focal loss of a picture is calculated as the sum of the focal loss over all anchors. It is normalized by the amount of anchors assigned to a ground-truth box. In general, alpha should be decreased slightly while gamma is increased in Equation (1). In the experiment, the experimental parameters gamma: 2 and alpha: 0.25 work best. The primary network of the model is ResNet-50 [7]. In the model, weight decay is set to 0.0001, momentum is set to 0.9, and initial learning rate is set to 0.01 during the first 60,000 iterations. Learning rate is reduced by 10% after every 60,000 iterations.

The next step is padding and scaling the picture size.

The orange box is the region of the detected vehicle. A right padding is to extend the right part and form a square image, as shown in the left sub-graph of Figure 6. A left padding is to extend the left part and form a square image, as shown in the right sub-graph of Figure 6. At this stage, the images are padded to form the complete image.

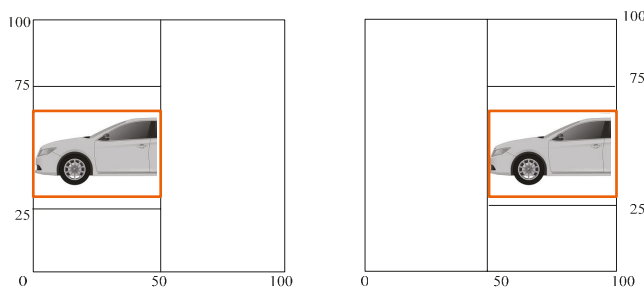


Figure 6. Right and left padding for a detected vehicle.

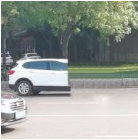
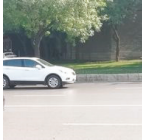


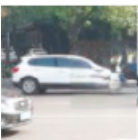




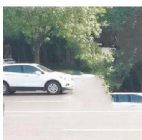
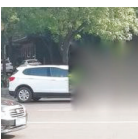
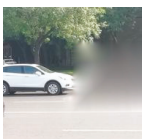
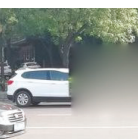
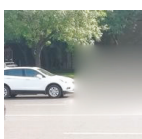

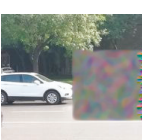
The purpose of the paper is to obtain information about a certain dangerous area to avoid accidents, not to precisely repair the appearance of the vehicle. To speed up the calculation, the detection image is scaled down to a square of  $128 \times 128$  pixels. Then, the small images are imported to the network.

#### 4.3. Image Inpainting and Re-Detection (Step 3 and Step 4)

Table 1 shows the flow chart from the padding to re-detection. Because the state of the art object detection algorithms cannot detect the vehicle with hidden parts, as shown in Figure 1, we do not compare with other object detection algorithms in this section. Pictures in the first row are the padded images, which are very clear. However, the pictures in the second row are blurrier than the pictures in the first line because they are scaled down to  $128 \times 128$  pixels. Row 3 shows the images after inpainting. The left and right inpainting are mandatory whether there are hidden parts or not. Object re-detection is shown in row 4. This is the fourth step of DIDA. If a vehicle has a hidden part, the detected region

in the second stage of object detection should be bigger than the region of the first stage of detection. This is shown in the orange box in column 3 of rows 4 and 2. Otherwise, the two regions should be same, as shown in column 4 of rows 4 and 2.

**Table 1.** The result of the inpainting and the re-detection.

Row Number	Describe	Vehicle with the Large Area Hidden Part	Vehicle without the Hidden Part
1	The image after padding		
2	The adding hole image based on scaled down image		
3	The image after inpainting using Generative Adversarial Networks (GAN)		
4	Redetection after the inpainting of our method		
5	Inpainting using texture synthesis method [42]		
6	Inpainting using Absolute Minimizing Lipschitz Extension (AMLE) [43]		
7	Inpainting using Mumford-Shah [44]		
8	Inpainting using Transport method [45]		



The images from the rows 5 to 8 show the inpainting results of the traditional methods. The methods are the texture synthesis method based on the algorithm [42] (row 5), Absolute Minimizing Lipschitz Extension (AMLE) [43] (row 6), Mumford-Shah Inpainting with Ambrosio-Tortorelli approximation [44] (row 7), and Transport Inpainting [45]. None of the traditional methods could handle the large region inpainting.

In the test phase, we randomly select 200 vehicle pictures to be used. The test dataset includes two types of pictures with and without hidden parts. In the real world, it is difficult to find the vehicle picture with a hidden part to verify the correctness of this algorithm. The test picture with a hidden part is a virtual picture created by Photoshop. No test pictures are seen by the train model. They are divided into two categories, each of which has 100 pictures. In the first category of the test dataset, the test pictures have a one quarter hidden region. In the second category of the test dataset, the test pictures have a half hidden region.

There are four different qualitative results of the algorithm—adding a part to the complete vehicle (wrong), adding a part to the incomplete vehicle (right), adding nothing to the complete one (right), and adding nothing to the incomplete one (wrong). In the research, we quantitatively define a correct result if the difference of the area between the predicted vehicle and the grand-truth is less than 10%. The precision results of the two test datasets were 91% and 82%, respectively.

4.4. Re-Scaling, Projecting (Step 5), and Setting Alarm Region (Step 6)

The left orange box in the top of Figure 7 shows the detected region of the vehicle before inpainting. The right orange box is the re-detected vehicle region after inpainting. The latter is bigger than the former. The difference between the two boxes is the hidden part. The vehicles are projected to the original image. The red box, the projected difference region, is defined as the alarm region (alarm box). Although the alarm area cannot be recognized as a part of the vehicle by a general vision system, this area is still dangerous. Figure 7 omits the procedure of the black car. The bottom picture is the ground-truth image.

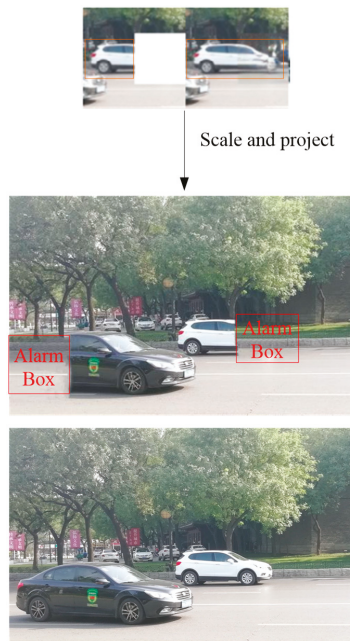


Figure 7. Scaling and projecting to the original image and setting the alarm box.

This algorithm does not lead to any risk. The DIDA algorithm performs an “adding” action based on the original vehicle image, not a “deleting” action. The worst scenario is that a part is added to a complete vehicle, which only reduces traffic efficiency.

## 5. Discussion

In uncommon situations, the color of a part of a vehicle is similar to the background color, which results in an error of optical object detection and consequently causes a failure of autopilot. The error can even cause a traffic accident. The proposed DIDA method solves this problem. As far as we know, the work is the first attempt to solve the problem by combining the object detection and image inpainting. However, the method has many steps, and the speed of the whole process needs improvement in the future. YOLOv3 [46] should be tested to improve the speed in the future.

In addition, the repaired parts are not perfect, such as the lack of tires. Although this does not affect the task, the improved GAN algorithm can be used in the future to generate the hidden part more accurately.

The category of the test pictures is not enough. In this research, only the normal condition of the test environment is considered. In the future, test pictures with sufficiently high numbers of vehicle categories should be added, such as different distances, different lighting conditions, angles of sunshine, shadows and background, etc.

In this work, we demonstrate an imaginative inpainting of a vehicle. It embodies a higher level artificial intelligence, such as prediction and imagination. The idea and method can be extended to other areas, such as underwater detection, intelligent transportation, robotics, etc.

Moreover, we share the CTE-Cars dataset that can be used for other future research of autopilot.

## 6. Conclusions

Vehicle detecting is very important in the autopilot field. If it fails to accurately detect vehicles, autopilot is dangerous. The optical detection method is ineffective in special cases where the color and texture of the parts appear similar to the background. The proposed DIDA method is a simple and effective approach to solving the problem. It is based on the deep learning network and only uses optical signals to detect the hidden part of a vehicle. The procedure includes vehicle detection by improved Retina network, scaling, image inpainting, the second stage of vehicle detection on the inpainted vehicle, zooming back to the original size, and setting an alarm region. From a scientific aspect, DIDA gives a new approach to detecting an insufficient object. From the point of engineering applications, DIDA offers a new method to insuring further security in motor vehicle auto driving systems.

**Author Contributions:** All authors contributed to the paper. Data curation, X.L.; Investigation, Q.G.; Project administration, Y.X. and H.W.; Software, W.S.; Writing—review & editing, H.R.H.

**Funding:** This work was supported by the Natural Science Foundation of NSFC under the grant No. 61571369 and No. 61471299. It was also supported by Zhejiang Provincial Natural Science Foundation (ZJNSF) No.LY18F010018.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. The Tesla Team. A Tragic Loss. Available online: <https://www.tesla.com/blog/tragic-loss> (accessed on 15 March 2019).
2. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
3. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
4. Krizhevsky, A.; Sutskever, I.; Geoffrey, H.E. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]

5. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9.
6. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:1409.1556.
7. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
8. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
9. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; pp. 234–241.
10. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
11. Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; Chen, X. Improved Techniques for Training GANs. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 2234–2242.
12. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M. Generative Adversarial Networks. *arXiv* **2014**, arXiv:1701.00160.
13. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 142–158. [[CrossRef](#)] [[PubMed](#)]
14. Uijlings, J.R.R.; van de Sande, K.E.A.; Gevers, T.; Smeulders, A.W.M. Selective search for object recognition. *Int. J. Comput. Vis.* **2013**, *104*, 154–171. [[CrossRef](#)]
15. Smola, A.J.; Schölkopf, B. A tutorial on support vector regression. *Stat. Comput.* **2004**, *14*, 199–222. [[CrossRef](#)]
16. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [[CrossRef](#)]
17. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 2–9.
18. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 13–16 December 2015; pp. 1440–1448.
19. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: New York, NY, USA, 2006.
20. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Proceedings of the Neural Information Processing Systems, Montréal Canada, 7–12 December 2015; pp. 91–99.
21. Dai, J.; Li, Y.; He, K.; Sun, J. R-FCN: Object detection via region-based fully convolutional networks. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 379–387.
22. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014.
23. Shafiee, M.J.; Chywl, B.; Li, F.; Wong, A. Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video. *arXiv* **2017**, arXiv:1709.05943. [[CrossRef](#)]
24. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
25. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 24–27 June 2014; pp. 580–587.

26. Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
27. Sermanet, P.; Eigen, D.; Zhang, X.; Mathieu, M.; Fergus, R.; LeCun, Y. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. *arXiv* **2013**, arXiv:1312.6229.
28. Bertalmio, M.; Sapiro, G.; Caselles, V.; Ballester, C. Image inpainting. In Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 23–28 July 2000; pp. 417–424.
29. Efros, A.A.; Freeman, W.T. Image quilting for texture synthesis and transfer. In Proceedings of the 28th annual Conference on Computer Graphics and Interactive Techniques, ACM, New York, NY, USA, 12–17 August 2001; pp. 341–346.
30. Pathak, D.; Krahenbuhl, P.; Donahue, J.; Darrell, T.; Efros, A.A. Context Encoders: Feature Learning by Inpainting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 2536–2544.
31. Iizuka, S.; Simo-Serra, E.; Ishikawa, H. Globally and locally consistent image completion. *ACM Trans. Graph.* **2017**, *36*, 107. [[CrossRef](#)]
32. Yang, C.; Lu, X.; Lin, Z.; Shechtman, E.; Wang, O.; Li, H. High-Resolution Image Inpainting Using Multi-Scale Neural Patch Synthesis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6721–6729.
33. Yu, J.; Lin, Z.; Yang, J.; Shen, X.; Lu, X.; Huang, T.S. Generative image inpainting with contextual attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 5505–5514.
34. Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; Courville, A. Improved Training of Wasserstein GANs. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 1–19.
35. Yang, L.; Luo, P.; Loy, C.C.; Tang, X. A large-scale car dataset for fine-grained categorization and verification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3973–3981.
36. Krause, J.; Stark, M.; Deng, J.; Fei-Fei, L. 3D object representations for fine-grained categorization. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops, Sydney, Australia, 1–8 December 2013; pp. 554–561.
37. Zhao, P.; Zhang, T. Accelerating Minibatch Stochastic Gradient Descent using Stratified Sampling. *arXiv* **2014**, arXiv:1405.3080.
38. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. *arXiv* **2013**, arXiv:1312.6114.
39. Dosovitskiy, A.; Springenberg, J.T.; Brox, T. Learning to Generate Chairs with Convolutional Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1538–1546.
40. Zeiler, M.D.; Fergus, R. Visualizing and understanding convolutional networks. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 818–833.
41. Clevert, D.-A.; Unterthiner, T.; Hochreiter, S. Fast and accurate deep network learning by exponential linear units (elus). *arXiv* **2015**, arXiv:1511.07289.
42. Harrison, P.F. Image Texture Tools: Texture Synthesis, Texture Transfer, and Plausible Restoration. Ph.D. Thesis, Monash University, Melbourne, Australia, 2005.
43. Almansa, A. Echantillonnage, Interpolation et Détection: Applications en Imagerie Satellitaire. Cachan, Ecole Normale Supérieure. Ph.D. Thesis, École Normale Supérieure Paris-Saclay, Cachan, France, 2002.
44. Esedoglu, S.; Shen, J. Digital inpainting based on the Mumford-Shah-Euler image model. *Eur. J. Appl. Math.* **2002**, *13*, 353–370. [[CrossRef](#)]

45. Bertalmio, M. Processing of Flat and Non-Flat Image Information on Arbitrary Manifolds Using Partial Differential Equations. Ph.D. Thesis, University of Minnesota, Minneapolis, MN, USA, March 2001.
46. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

Article

# Camera-Based Blind Spot Detection with a General Purpose Lightweight Neural Network

Yiming Zhao, Lin Bai, Yecheng Lyu and Xinming Huang \*

Department of Electrical and Computer Engineer, Worcester Polytechnic Institute, Worcester, MA 01609, USA; yzhao7@wpi.edu (Y.Z.); lbai2@wpi.edu (L.B.); ylyu@wpi.edu (Y.L.)

\* Correspondence: xhuang@wpi.edu

Received: 1 January 2019; Accepted: 13 February 2019; Published: 19 February 2019

**Abstract:** Blind spot detection is an important feature of Advanced Driver Assistance Systems (ADAS). In this paper, we provide a camera-based deep learning method that accurately detects other vehicles in the blind spot, replacing the traditional higher cost solution using radars. The recent breakthrough of deep learning algorithms shows extraordinary performance when applied to many computer vision tasks. Many new convolutional neural network (CNN) structures have been proposed and most of the networks are very deep in order to achieve the state-of-art performance when evaluated with benchmarks. However, blind spot detection, as a real-time embedded system application, requires high speed processing and low computational complexity. Hereby, we propose a novel method that transfers blind spot detection to an image classification task. Subsequently, a series of experiments are conducted to design an efficient neural network by comparing some of the latest deep learning models. Furthermore, we create a dataset with more than 10,000 labeled images using the blind spot view camera mounted on a test vehicle. Finally, we train the proposed deep learning model and evaluate its performance on the dataset.

**Keywords:** squeeze-and-excitation; residual learning; depthwise separable convolution; blind spot detection

---

## 1. Introduction

In the 2012 ILSVRC competition, deep convolutional neural network designed by Hinton et al. achieved the lowest error rate of 15.3% that is 10.8% better than the runner up [1]. The large and complex classification dataset in ILSVRC with 1000 object categories is widely regarded as a benchmark to evaluate different machine learning models [2]. This milestone achievement attracted many researchers to the field of deep neural networks. In the following years, adaption of new neural network structures continually pushed the error rate lower. In ILSVRC-2014, GoogLeNet achieved 6.67% error rate using inception module [3]. A series of inception modules can find the optimal local construction by concatenating output from convolution kernels in different sizes [4,5]. In the same year, VGG model was published with outstanding performance results, which quickly became one of the most popular structures [6]. In ILSVRC-2015, the invention of residual error in ResNet made it possible to train a very deep network with more than 100 convolution layers [7]. In ILSVRC-2017, researchers from University of Oxford even achieved 2.25% error rate with squeeze-and-excitation module [8].

The success of deep convolutional neural network in image classification stimulates the research interest of solving many other challenging computer vision problems. For object detection tasks, some models like YOLO [9,10] and SSD [11] directly transformed this task to a regression problem with neural network. The R-CNN series model [12–14] replaced the traditional histogram of gradients (HOG) with Selective Search and SVM method [15] using deep neural network. Mask R-CNN [16] can directly generates pixel level semantic segmentation with bounding box. More complex tasks usually

require the extraction of more elaborated and accurate information from images. Other researchers proposed various convolutional kernels, such as deformable convolution [17] which can change the shape of kernels or dilated convolution [18] which can capture a larger range of information with the same kernel size.

For an embedded system with limited computing capacity, the heavy computational cost of very deep neural network is prohibitive. Methods to increase the training and inference speed by reducing the parameters and operations become an important topic lately. Many recent models such as Xception [19] and MobileNet [20] designed the neural networks based on depthwise separable convolutions. This module can dramatically reduce the number of parameters without losing too much accuracy. Furthermore, ShuffleNet [21] utilized less parameters by shuffling and exchanging the output after group convolutions.

However, neural networks in all the aforementioned models still contain lots of layers. The MobileNetV2 [22] released recently has 17 blocks which include 54 convolution layers in total. The question is—do we really need such a large network to solve a specific real-world problem? If the task is not that complex and we ought to avoid the very deep structure, how should we adapt the deep learning models? Do we still need residual learning? In order to answer those questions, we propose a network structure based on AlexNet [1]. After the first convolution layer, there are four identical blocks. For each block, we consider different combinations of the latest deep learning models including residual learning, separable depthwise convolution and squeeze-and-excitation. The final model is chosen by comparing the evaluation accuracy and computing cost.

Finally, we implement our proposed model for camera-based blind spot detection. Blind spot detection is very important to driving safety. However, the radar based system is relatively expensive and has a limited ability for complex situations. There are few works focusing on camera based blind spot detection and the existing publications are largely based on artificial features or traditional signal processing methods [23–26].

There are two main contributions in this paper. One is that we combine depthwise separable convolution, residual learning and the squeeze-and-excitation module together to design a new block. Compared with VGG block, deep neural network composed of the proposed new block can achieve similar performance but with significantly less parameters when evaluated on CIFAR-10 dataset and our own blind spot dataset. More importantly, we present a complete solution to camera-based blind spot detection from experimental setup, data collection and labelling, deep learning model selection, and performance evaluation.

## 2. Related Work

Our goal is to design a deep neural network model that can solve real world problems without involving too many layers. The basic principle to design a lightweight neural network is reducing the model size without losing too much accuracy. So we briefly introduce the latest works on network size reduction. Then, we revisit three deep learning modules considered in this paper.

### 2.1. Existing Work on Reducing Network Size

Most networks are built with 32-bit floating point weights, so an intuitive reduction technique is to quantize the weights in fixed-point representation. By using 16-bit fixed-point representation in stochastic rounding based CNN training, the model memory usage and operations were significantly reduced with little lost on classification accuracy [27]. The extreme case for weight representation is the binary notation. Some researchers directly applied binary weights or activations during the model training process [28,29].

Model compression is another effective approach. Almost 30 years ago, the universal approximation theorem of neural network stated that simple neural networks can represent a wide variety of nonlinear functions when given appropriate parameters [30]. It was reported using a shallow network to mimic the complex, well-engineered, deeper convolutional models [31]. For a complex

network, discarding redundant nodes is another way to reduce the model size. A recent work combined model pruning, weights quantization and Huffman coding to achieve better performance [32].

## 2.2. Deep Learning Module Revisit

This paper is aiming at designing network structure to reduce the computational cost. Separable depthwise convolution can dramatically decrease the number of parameters and operations of the network. Since separable depthwise convolution may result in the loss of accuracy, we consider adding residual learning and squeeze-and-excitation module. These two modules require little additional computing cost, but they are capable of improving accuracy.

### 2.2.1. Separable Depthwise Convolution

Depthwise separable convolution became popular recently for mobile devices [19–22,33]. Although there is difference among several existing works, the core idea is the same. Compared with standard  $3 \times 3$  convolution in VGG, depthwise separable convolution does channel-wise convolutional calculations with  $3 \times 3$  kernels. Then standard  $1 \times 1$  convolutions are applied to integrate information for all channels. Let us assume the number of input channel is  $M$  and the number of output channel is  $N$ . Standard  $3 \times 3$  convolutions require  $M \times 3 \times 3 \times N$  parameters. The depthwise separable convolutions only need  $M \times 3 \times 3 \times 1 + M \times 1 \times 1 \times N$  parameters which are much less than the standard  $3 \times 3$  convolutions.

### 2.2.2. Residual Learning

When researchers made the network deeper and deeper, they encountered an unexpected problem. As the network depth increases, accuracy gets saturated and then degrades rapidly. Residual learning solves this problem with an elegant yet simple solution [7]. In deep residual learning framework, as shown in Figure 1, the original mapping  $F(x)$  is recasted into  $F(x) + x$ , which is the summation of original mapping and the identity mapping of input. This simple solution magically makes it possible to train a very deep neural network with only a small increase of computation. Hence, residual learning quickly became a popular component in the latest deep learning models. DenseNet is also a helpful way to train very deep neural networks [34]. However, the concatenating operation will greatly increase computational cost and parameters. Thus we do not consider DenseNet in this paper.

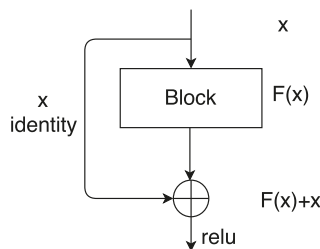


Figure 1. Residual learning on an existing block.

### 2.2.3. Squeeze-and-Excitation

Based on the squeeze-and-excitation (SE) block, SE network won the first place of the classification task in ILSVRC 2017 with the top-5 error 2.25% [8]. SE module can re-weight each feature map by imposing only a small increase in model complexity and computational burden.



In Figure 2, we show how SE module operates with an existing block. Let us assume the output tensor is  $A$  and the shape of  $A$  is  $W \times H \times C$ . Then, the global average operation maps each feature map into one value by calculating the average on each feature map.

$$\hat{A}_k = \frac{1}{W \times H} \sum_{i=1}^W \sum_{j=1}^H A_{ijk}$$

After that, the first fully connected layer performs the squeezing step by  $\frac{C}{r}$  units, and the second fully connected layer does the excitation step by  $C$  units.  $r$  is the reduction ratio in SE module, which can decide the squeeze level of the module and affect the number of parameters. Finally, a sigmoid activation layer transforms it to probability and does the multiplication with original tensor  $A$ .

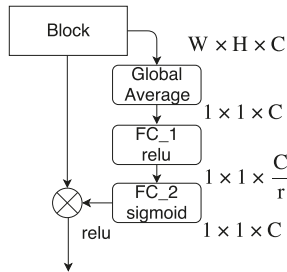


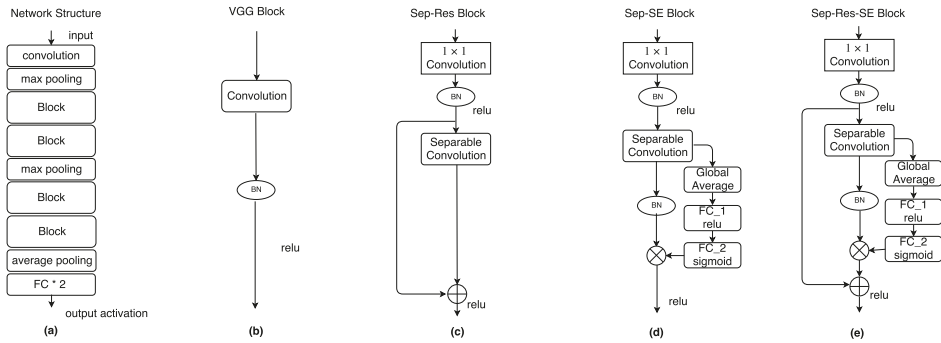
Figure 2. Squeeze-and-excitation on an existing block.

### 3. Investigation of Deep Learning Models

As we mentioned earlier, researchers designed many new network modules to empower the deep learning in recent years. However, an embedded computing platform can only handle the computation load of a model with a few layers. In this case, how should we choose those modules? Here, we hold the same network structure but change the setting of the building block. We propose four different blocks, corresponding to four different neural networks. In Figure 3a, we show the entire network structure. In order to keep it simple and intuitive, we use a VGG-like structure. We begin with one standard convolution layer to extract information from the input data. The kernel size is decided by the shape of the input tensor. For CIFAR-10 dataset, the input shape is  $(32, 32, 3)$ , so we use  $3 \times 3$  kernel with stride 1. For blind spot detection dataset, the input shape is  $(128, 64, 3)$ , so we use  $5 \times 5$  kernel with stride 2. Then, a max pooling is used to compress the information. The main body of the structure consists of four identical blocks. If we choose a VGG block, then there should be four VGG blocks in the main part of the network. The output part contains one average pooling and two fully connected layers. Finally, an activation layer shows output probability of each category. We use sigmoid for two-class problems and softmax for multi-class problems. In Figure 3, we show the details of four different blocks. In Figure 3b, we can see the VGG block has a standard convolution followed by batch normalization and relu function. We set this block as the baseline. Next, we use depthwise separable convolution to replace the standard convolution. This modification reduces the number of parameters significantly. Furthermore, we separately equip the new convolution with residual learning as in Figure 3c to form the Sep-Res block, or with squeeze-and-excitation as in Figure 3d to form Sep-SE block. Finally, we combine those three parts together to ensemble the Sep-Res-SE block, as in Figure 3e. By comparing the results of Sep-Res block and Sep-Res-SE block, we can evaluate the performance of squeeze-and-excitation. By comparing the results of Sep-SE block and Sep-Res-SE block, we can evaluate the performance of residual learning.

Residual learning requires the output tensor in the same size of the input tensor, so we need to keep a constant number of channels in the block. In this paper, we solve this problem by adding one  $1 \times 1$  standard convolution at the beginning of the block as in Figure 3c–e. This bottleneck convolution

can increase the number of channels as needed. When we train squeeze-and-excitation module, we find it is harder to converge. So we add batch normalization before the multiplication to help it converge faster.



**Figure 3.** We show the structure of our neural network in (a). The first convolution layer extracts information from input data and the two fully connected layers at the bottom gradually change the output size to class number. There are four blocks in the main body of the network; we can put different settings in all those four blocks to see how to design network can perform better. (b) is a standard VGG Block, a convolution followed by batch normalization and relu. (c) is a Sep-Res Block, we replace standard convolution with separable depthwise convolution and add residual learning module on it. The first  $1 \times 1$  convolution increase channels to guarantee the output shape is the same as input shape for residual. (d) is a Sep-SE Block, we replace the residual learning module by squeeze-and-excitation module. In (e), we combine all those parts together to form Sep-Res-SE Block.

#### 4. Experiments and Evaluation

##### 4.1. Datasets

###### 4.1.1. CIFAR-10

The CIFAR-10 dataset is a popular dataset for evaluation of machine learning methods. It contains 60,000  $32 \times 32$  color images in 10 different categories. The low resolution, a few categories and sufficient samples in each category make this dataset suitable for evaluating the performance of our proposed models.

###### 4.1.2. Blind Spot Detection

In this subsection, we discuss the procedures to transform the blind spot detection to a machine learning problem. We draw the blind spot region in Figure 4. The region consists with four  $4\text{ m} \times 2\text{ m}$  rectangles. Driver should be alerted if any car enters this region. So we model blind spot detection as a Car or No-Car two-class classification problem. No-Car class indicates there is no vehicle in the blind spot region, so it is safe for lane changing. Car class means there is at least part of a vehicle in the blind spot region, thus driver should not change lanes. Our test vehicle is a Lincoln MKZ equipped with high-resolution Sekonix cameras. We mount the blind spot camera on the side of rooftop with 45 degrees facing backward, and the position of blind spot camera is shown in Figure 5a. In order to capture information better, we create a 3D region with 2 m in height recorded by camera in Figure 5b. Before feeding the training image into models, we preprocess original image by clipping the blind spot region and resize it to  $128 \times 64$ . Figure 5c is the input image of Figure 5b after preprocessing.

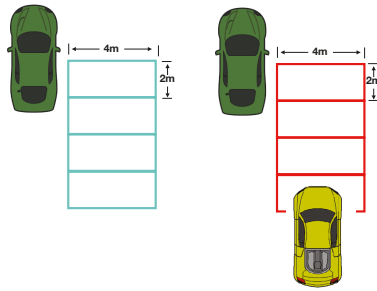


Figure 4. Bird's-eye view of blind spot region.



Figure 5. We show the position of our camera in (a). (b) is the blind spot region in the view of camera. (c) is the training image after preprocessing (b). (d) is an example of No-Car class in training data, (e,f) are examples of Car class in training data. (g–i) are examples of the final output of our blind spot detection system. If the model finds a car, it will alarm the driver by changing the color of the box. Our model is fairly accurate and can exactly account for the blind spot region.

We record several videos when driving on highways and combine them together into one large video. Then, we split the video as training video and test video. For training video, we choose one out of every five frames as the training image. For test video, we use all the frames without sampling as the test dataset. Next, we draw the blind spot region on the training dataset and label them manually. As an example, Figure 5d belongs to No-Car class since the vehicle did not stay in the prescribed 3D region. Figure 5e,f both belong to the Car class since at least part of a vehicle appears in the blind spot region. In total, we obtained 8336 images of the No-Car class and 2184 images belonging to the Car class. The imbalanced dataset would limit the performance of machine learning models and some methods were developed to solve this problem [35,36]. Here we take a simple task-specific policy to balance the dataset. For No-Car class, we discard 3336 similar images which just contain road surface. For Car class, we double the images each with a vehicle occupying two or more rectangles in the blind spot region. Finally, we obtained 5000 images in No-Car class and 3874 images in Car class.

#### 4.2. Experiment Setting

In this part, we discuss the details of model setting. All the models in this paper share the same setting for the first layer, that is a standard convolutional layer with 64 channels. For CIFAR-10 dataset, it has  $3 \times 3$  convolutional kernel with stride 1. For Blind Spot Detection dataset, it has  $5 \times 5$  convolutional kernel with stride 2. The successively repeated four blocks in each model have the same setting for both datasets. All the standard convolutional layers and separable convolutional layers in each of those four blocks use  $3 \times 3$  kernel with 1 stride. There are 128 channels in the first two blocks, and 256 channels in the last two blocks. The squeeze factor  $r$  is 16. All the pooling layers have  $2 \times 2$  kernel with stride 2.

For each dataset, we use the same learning rate and batch size to make a fair comparison among four different neural networks. For CIFAR-10, we set  $learning\_rate = 0.001$ ,  $batch\_size = 64$ , choose Adam [37] as the optimizer and train all four models for 100 epoch. For blind spot dataset, we set  $learning\_rate = 0.0001$ ,  $batch\_size = 64$ , choose Adam as the optimizer and train all the four models for 30 epoch.

#### 4.3. Results

In Table 1, we compare the test accuracy of all four models on two datasets. The Sep-Res-SE model that combines depthwise separable convolution, residual learning and squeeze-and-excitation performs better than the other two models. The VGG Block still has slightly higher accuracy than Sep-Res-SE Block. However, test accuracy is not the only factor that we should consider for the real-time problem. Inference speed and memory cost are also important for an embedded system. We list the inference speed for each model on blind spot dataset with Nvidia Quadro p6000 in Table 1. The model with VGG block requires nearly twice as much time to handle one image when comparing to the other modules. In fact, inference speed is decided by the amount of operations and the memory cost is decided by the number of parameters. In Table 2, we show the number of parameters and operations in the first block of each model. Since four repeated blocks comprise the main body of the model, the parameters and operations in one block can clearly show the computational cost of each model. From Table 2, the VGG model that achieves higher test accuracy requires twice or more parameters and operations. Compared with standard convolution, models equipped with depthwise separable convolution require far fewer parameters and operations. By combining residual learning and squeeze-and-excitation, we can compensate the test accuracy of depthwise separable convolution with slightly more parameters and operations but achieve similar accuracy comparable to the VGG Block.

Therefore, the combining of depthwise separable convolution, residual learning and squeeze-and-excitation is the best tradeoff between accuracy and cost. Subsequently, we apply the trained neural network model to our test vehicle with cameras installed. Figure 5g–i are examples from the real application. It shows that the proposed method can detect the car in the blind spot region effectively with no confusion by the vehicles outside of the region. Our test video posted online shows that the proposed model nearly detected all the cars in the blind spot region. A few mistakes occurred owing to the shadow of the bridge casted on the surface of the road. The model can be further improved by adding more training data.

MobileNet is one of the well-known deep learning structures for mobile devices. For comparison, we also implemented MobileNetV2 which is the recent version of MobileNet. We set the  $learning\_rate = 0.0001$ ,  $batch\_size = 64$ , choose Adam as the optimizer and train it for 100 epoch. We show the comparison results in Table 3. Since the number of model operations are strongly affected by the input tensor size and number of filters, the idea of MobileNetV2 is to downsample the image size quickly and to use less filters in the top layers. However, MobileNetV2 requires many more layers to keep up the performance. As in Table 3, MobileNetV2 has less operations but more parameters comparing with Sep-Res-SE. We test both MobileNetV2 and Sep-Res-SE in the same environment. The MobileNetV2 get 95.35% test accuracy which is lower than the Sep-Res-SE. The inference speed of MobileNetV2 is

also slower than Sep-Res-SE. Although we believe MobileNetV2 may be able to achieve similar or even better results after fine tuning the training process, it is not efficient to use a very deep neural network with large memory cost for the blind spot detection problem.

**Table 1.** This table show the test accuracy on two datasets with four different neural networks and the inference speed on blind spot dataset.

Test Result			
Network Block Type	CIFAR10	Blind Spot	Inference Speed per Image
VGG Block	0.8829	0.9801	0.00259s
Sep-Res Block	0.8554	0.9737	0.00159s
Sep-SE Block	0.8575	0.9701	0.00166s
Sep-Res-SE Block	0.8730	0.9758	0.00169s

**Table 2.** This table shows the number of parameters and operations of the first block in each model. We count both multiplication and add(Multi-Add) as the operation.

	CIFAR10		Blind Spot Detection	
	Params	Multi-Add	Params	Multi-Add
VGG Block	73.7k	37.8M	73.7k	302.5M
Sep-Res Block	25.7k	13.3M	25.7k	106.2M
Sep-SE Block	33.9k	13.4M	33.9k	106.9M
Sep-Res-SE Block	33.9k	17.6M	33.9k	140.5M

**Table 3.** This table shows the comparison with the model proposed in this paper with MobileNet which is a famous neural network structure for mobile device.

Model Comparison on Blind Spot Detection Dataset				
Model	Params	Multi-Add	Accuracy	Inference Speed per Image
Sep-Res-SE	143.4k	420M	0.9758	0.00169s
MobileNetV2	3.4M	48.9M	0.9535	0.00488s

## 5. Conclusion and Discussion

In this paper, we discuss how to design a neural network with only a few layers for real-time embedded applications, such as blind spot detection. Usually higher accuracy requires deeper model and better computational cost. By using depthwise separable convolution, we dramatically reduce the model parameters and operations. Then, we add residual learning and squeeze-and-excitation module to compensate the loss of accuracy with only a small increase of parameters. Compared with VGG block, the Sep-Res-SE block, combining depthwise separable convolution, residual learning and squeeze-and-excitation can achieve similar detection accuracy with far fewer parameters and operations. We recommend this model as the best tradeoff between accuracy and cost.

We also present a complete solution to camera-based blind spot detection. We successfully solve this problem by building a machine learning model from labeling dataset. However, we have not yet considered all the situations with different roads and weather conditions due to the additional workload of gathering and labeling data. Moreover, if geo-information is provided by sensors like Inertial Measurement Unit (IMU), the model can be further improved for sloped road by adjusting the blind spot region.

**Author Contributions:** Conceptualization, Y.Z. and L.B.; methodology, Y.Z.; formal analysis, Y.Z.; data curation, Y.L. and Y.Z.; writing—original draft preparation, Y.Z.; writing—review and editing, X.H.; supervision, X.H.; project administration, X.H.; funding acquisition, X.H.

**Funding:** This research was funded by National Science Foundation grant number 1626236.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Neural Information Processing Systems (NIPS), Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
2. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* 2015, *115*, 211–252. [[CrossRef](#)]
3. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
4. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
5. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017.
6. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* 2014, arXiv:1409.1556.
7. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
8. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 19–21 June 2018; pp. 7132–7141.
9. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
10. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 22–25 July 2017; pp. 6517–6525.
11. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 21–37.
12. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
13. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 13–16 December 2015; pp. 1440–1448.
14. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 91–99.
15. Uijlings, J.R.; Van De Sande, K.E.; Gevers, T.; Smeulders, A.W. Selective search for object recognition. *Int. J. Comput. Vis.* 2013, *104*, 154–171. [[CrossRef](#)]
16. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
17. Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; Zhang, G.; Hu, H.; Wei, Y. Deformable convolutional networks. *arXiv* 2017, arXiv:1703.06211.
18. Yu, F.; Koltun, V. Multi-scale context aggregation by dilated convolutions. *arXiv* 2015, arXiv:1511.07122.
19. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1800–1807.
20. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* 2017, arXiv:1704.04861.

21. Ma, N.; Zhang, X.; Zheng, H.T.; Sun, J. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In *Computer Vision—ECCV 2018*; Springer: Berlin, Germany, 2018; pp. 122–138.
22. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation. *arXiv* 2018, arXiv:1801.04381.
23. Liu, G.; Zhou, M.; Wang, L.; Wang, H.; Guo, X. A blind spot detection and warning system based on millimeter wave radar for driver assistance. *Opt. Int. J. Light Electron Opt.* 2017, *135*, 353–365. [[CrossRef](#)]
24. Van Beeck, K.; Goedemé, T. The automatic blind spot camera: A vision-based active alarm system. In *Proceedings of the European Conference on Computer Vision*, Amsterdam, The Netherlands, 8–16 October 2016; pp.122–135.
25. Hyun, E.; Jin, Y.S.; Lee, J.H. Design and development of automotive blind spot detection radar system based on ROI pre-processing scheme. *Int. J. Automot. Technol.* 2017, *18*, 165–177. [[CrossRef](#)]
26. Baek, J.W.; Lee, E.; Park, M.R.; Seo, D.W. Mono-camera based side vehicle detection for blind spot detection systems. In *Proceedings of the 2015 Seventh International Conference on Ubiquitous and Future Networks (ICUFN)*, Sapporo, Japan, 7–10 July 2015; pp. 147–149.
27. Gupta, S.; Agrawal, A.; Gopalakrishnan, K.; Narayanan, P. Deep learning with limited numerical precision. In *Proceedings of the International Conference on Machine Learning*, Lille, France, 6–11 July 2015; pp. 1737–1746.
28. Courbariaux, M.; Bengio, Y.; David, J.P. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Proceedings of the Advances in Neural Information Processing Systems*, Montreal, Canada, 7–12 December 2015; pp. 3123–3131.
29. Rastegari, M.; Ordonez, V.; Redmon, J.; Farhadi, A. Xnor-net: Imagenet classification using binary convolutional neural networks. In *Proceedings of the European Conference on Computer Vision*. Amsterdam, The Netherlands, 8–16 October 2016; pp. 525–542.
30. Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural Netw.* 1991, *4*, 251–257. [[CrossRef](#)]
31. Ba, J.; Caruana, R. Do deep nets really need to be deep? In *Proceedings of the Advances in Neural Information Processing Systems*, Montreal, QC, Canada, 8–13 December 2014; pp. 2654–2662.
32. Han, S.; Mao, H.; Dally, W.J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv* 2015, arXiv:1510.00149.
33. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated residual transformations for deep neural networks. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, 22–25 July 2017; pp. 5987–5995.
34. Huang, G.; Liu, Z.; Weinberger, K.Q.; van der Maaten, L. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, 22–25 July 2017; pp. 4700–4708.
35. He, H.; Garcia, E.A. Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.* 2009, *21*, 1263–1284.
36. Krawczyk, B. Learning from imbalanced data: Open challenges and future directions. *Progress Artif. Intell.* 2016, *5*, 221–232. [[CrossRef](#)]
37. Kinga, D.; Ba, J. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, San Diego, CA, USA, 7–9 May 2015.



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

Article

# Using Wearable ECG/PPG Sensors for Driver Drowsiness Detection Based on Distinguishable Pattern of Recurrence Plots

Hyeonjeong Lee, Jaewon Lee and Miyoung Shin \*

Bio-Intelligence & Data Mining Laboratory, School of Electronics Engineering, Kyungpook National University, Daegu 41566, Korea; leehj1224k@gmail.com (H.L.); realjaewon94@gmail.com (J.L.)

\* Correspondence: shinmy@knu.ac.kr; Tel.: +82-053-940-8685

Received: 30 December 2018; Accepted: 1 February 2019; Published: 7 February 2019

**Abstract:** This paper aims to investigate the robust and distinguishable pattern of heart rate variability (HRV) signals, acquired from wearable electrocardiogram (ECG) or photoplethysmogram (PPG) sensors, for driver drowsiness detection. As wearable sensors are so vulnerable to slight movement, they often produce more noise in signals. Thus, from noisy HRV signals, we need to find good traits that differentiate well between drowsy and awake states. To this end, we explored three types of recurrence plots (RPs) generated from the R–R intervals (RRIs) of heartbeats: Bin-RP, Cont-RP, and ReLU-RP. Here Bin-RP is a binary recurrence plot, Cont-RP is a continuous recurrence plot, and ReLU-RP is a thresholded recurrence plot obtained by filtering Cont-RP with a modified rectified linear unit (ReLU) function. By utilizing each of these RPs as input features to a convolutional neural network (CNN), we examined their usefulness for drowsy/awake classification. For experiments, we collected RRIs at drowsy and awake conditions with an ECG sensor of the Polar H7 strap and a PPG sensor of the Microsoft (MS) band 2 in a virtual driving environment. The results showed that ReLU-RP is the most distinct and reliable pattern for drowsiness detection, regardless of sensor types (i.e., ECG or PPG). In particular, the ReLU-RP based CNN models showed their superiority to other conventional models, providing approximately 6–17% better accuracy for ECG and 4–14% for PPG in drowsy/awake classification.

**Keywords:** drowsiness detection; smart band; electrocardiogram (ECG); photoplethysmogram (PPG); recurrence plot (RP); convolutional neural network (CNN)

## 1. Introduction

Driver drowsiness or fatigue is one of main causal factors to many road accidents. Accordingly, as a car safety technology to reduce such accidents, the driver drowsiness detection problem is widely examined [1–3], in which various measures are obtainable. The types of measures used in existing studies for driver drowsiness detection include vehicle-based measures, behavioral measures, and physiological measures.

The vehicle-based measures contain wheel position, handle movement, velocity, acceleration, etc. These measures have the advantage of being non-invasive and relatively accurate, but are highly dependent on driver's driving skills, road conditions, and vehicle characteristics. Moreover, they have some potential risks of taking time in detecting the motion of a vehicle to avoid accidents in real driving situations [4–6]. On the other hand, the behavioral measures include driver's eye state, eye blinking rate, yawning, head movement, and so on. Recently, these measures were widely used with deep learning technology [6–8]. These measures are also non-invasive and easy to use, but there are some drawbacks that they are sensitive to camera movement, lighting conditions, and the surrounding environment [1,6,9].



As an alternative or complement to vehicle-based or behavioral measures, physiological measures are currently actively used [2,10], and they contain various biometrical signals (such as heart rate, brain activity, respiration, etc.) acquired from different types of sensors (such as electrocardiogram (ECG), electroencephalogram (EEG), photoplethysmogram (PPG), etc.). In particular, many studies [11–17] were conducted to analyze heart rate variability (HRV) alone, collected with ECG or PPG sensors, which are relatively simple to measure. For example, Vicente et al. [18] implemented HRV-derived features extracted from ECG signals for drowsiness detection. Kim et al. [16] used various features obtained from PPG and respiration sensors to classify driver drowsiness and awake states. Malik et al. [17] applied a series of instantaneous heart rate (IHR) obtained from ECG and PPG sensors to determine whether a subject is awake or asleep.

Conventional HRV-derived features used in many earlier studies usually focused on spectral changes. Such traits are simple and easy to calculate, but are not good enough to capture nonlinear dynamics of complex systems. To handle nonlinear characteristics of physiological signals like ECG that tend to be nonstationary in nature [10,12,19,20], recurrence plots (RPs) were investigated in several studies [21–23]. The RPs were originally introduced by Eckmann et al. [21], and represent the phase-space trajectory of a dynamical system. Since then, they were often used for various studies. Furthermore, recurrence quantification analysis (RQA) measures were derived in References [22,23] to quantify the structures of RPs, and were used for sleep stage classification and obstructive sleep apnea problems [23–25]. The RQA features, however, have limited expressiveness for the characteristics of RP. That is, they do not fully represent various changes over time or trends shown in the RP. In light of these drawbacks, it seems worthwhile to identify robust and reliable patterns from RP that are distinguishable between drowsy and awake states. Moreover, people are generally reluctant to using ECG sensors in real driving situations owing to their intrusiveness (i.e., they disturb drivers by attaching many sensors on the body). On the other hand, PPG sensors in band types represent a small, simple, and low-cost device that can monitor the pulse rate in a non-invasive manner [26,27], even if PPG signals are more susceptible to noise than ECG signals. As many PPG wearable sensors are available that are portable and relatively inexpensive, it will be interesting to use them to detect driver fatigue or drowsiness, if possible, in an actual driving environment.

This study aims to investigate the robust and distinguishable pattern of HRV signals for driver drowsiness detection that can provide reliable results, regardless of which sensors (ECG or PPG) being used. For this, we explored three types of RPs (including Bin-RP, Cont-RP, and ReLU-RP) produced from R–R intervals (RRIs) of ECG (or PPG) sensors. Specifically, we utilized each of these RPs as input features to a convolutional neural network (CNN) for drowsy/awake classification (see Figure 1). For experiments, we collected RRIs at drowsy and awake conditions with an ECG sensor of the Polar H7 strap and a PPG sensor of the Microsoft (MS) band 2, in a virtual driving environment.

The organization of the paper is as follows: Section 2 explains the driver drowsiness dataset used in this study, and the preprocessing process for our analyses. Also, drowsy and awake states are characterized based on three types of RPs, followed by the drowsiness detection model development with CNN and others. Section 3 evaluates the drowsy/awake classification performance of the proposed model and compares it with other existing approaches. Finally, Section 4 discusses the results and concludes our study.

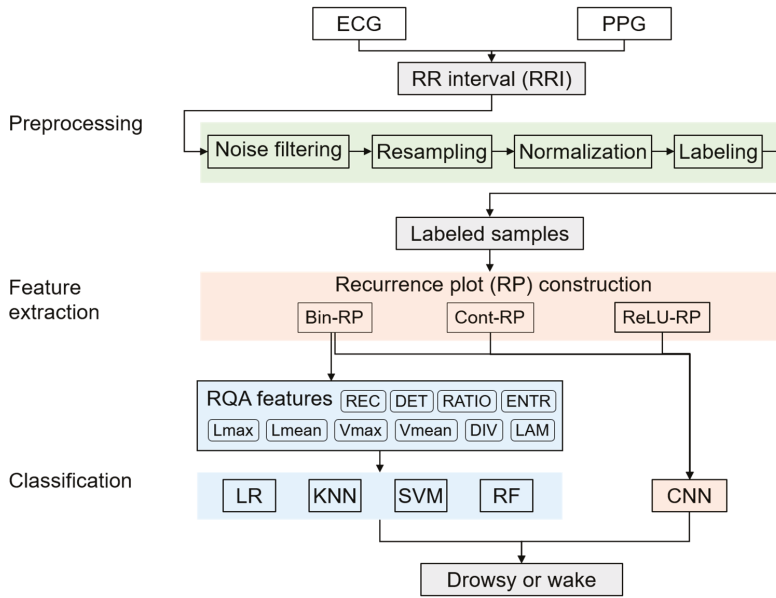


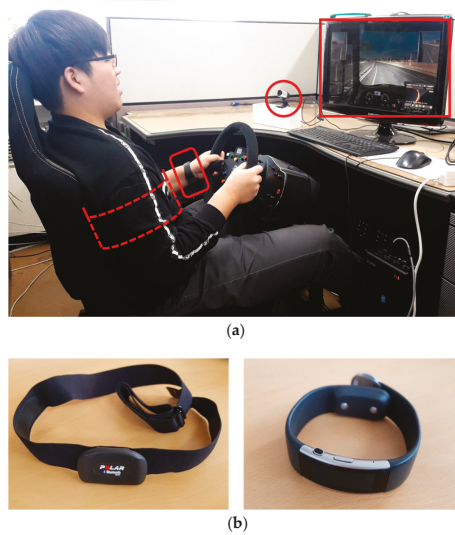
Figure 1. Overall workflow.

## 2. Materials and Methods

### 2.1. Driver Drowsiness Dataset

In this study, we used our in-house driver drowsiness dataset for experiments. This dataset was collected under a virtual driving simulation environment (Figure 2a) where RRIs of each subject were obtained with two body-worn sensors: a Polar H7 strap ECG sensor and an MS Band2 PPG sensor, as shown in Figure 2b. The Polar H7 strap was worn on the chest to measure cardiac data, and MS Band 2 was worn on the left wrist. The strap measured RRIs approximately every second while the band measured RRIs approximately one to two times every second at its own sampling rate. For the measurement, all the subjects received instructions to drive on the virtual road provided by Euro Truck Simulator 2 program [28], allowing a maximum speed of 90 km/h. In addition, the subjects were advised not to drink caffeine beverages within three hours of measurement.

Under this controlled environment, for each subject, we recorded two kinds of RRIs acquired from ECG and PPG sensors, and also two different videos, one for the face recording and the other for driving behavior recording. Here, the two videos would be used later for labeling of the RRI data. Each recording time was approximately one to two hours. Currently, the dataset includes 22 recordings of six subjects (men and women, 20–35 years) in total. Among these, six recordings were collected in the morning when the subject was usually clearly awake, and 16 recordings were recorded after lunch or dinner when the subject was usually feel fatigued or sleepy. To keep better quality of data, in this study, we utilized awake state data taken from morning recordings and drowsy state data taken from after-lunch or after-dinner recordings. Each recording was labeled by assigning a drowsy or awake state to every 1-min length of RRIs with reference to the videos of a subject’s face and driving behaviors.



**Figure 2.** Data collection instruments and environment: (a) Driving simulation environment; (b) physiological devices used for heart rate variability (HRV) collection: POLAR H7 Strap (left) and Microsoft BAND 2 (right).

## 2.2. Data Preparation

For the development of the driver drowsiness detection model, the dataset was preprocessed as follows: firstly, we filtered out noise effects in such a way that if an RRI was not within 20% SD from the mean of its 10 neighbors (including five preceding RRIs and five following RRIs), it was considered to have a noise effect and was replaced with the mean of its neighbors (see our earlier work in Reference [29] for details).

Also, we adjusted the two RRI measurements acquired from the ECG and PPG sensors to the same sampling rate. To do this, for both of the measurements, we interpolated the two adjacent RRIs in each recording with linear functions and resampled them to have a 1-Hz sampling rate. With these resampled RRIs of each recording, we generated 2-min RRI samples without overlapping, and labeled them using only the samples with the same state in a 2-min epoch. That is, we excluded the 2-min RRI samples that had a different state between the first minute and the second minute (e.g., the first minute was drowsy and the second minute was awake). This was intended to have a better classification model with clearly differentiated samples between drowsy and awake states. Then, for further analyses, these 2-min RRI samples were standardized to have a mean of 0 and an SD of 1.

Finally, we obtained the collection of 684 2-min samples, which included 234 drowsy and 450 awake samples, from each type of sensors (i.e., ECG or PPG). Among these, for each sensor, we only used 203 drowsy samples measured after lunch or dinner and 138 awake samples measured in the morning. Thus, for experiments, we eventually used total 341 samples acquired from ECG strap and PPG band, respectively, which include 203 samples at drowsy state and 138 samples at awake state.

## 2.3. Characterization of Drowsy and Awake States Based on Recurrence Plots

For driver drowsiness detection, it is necessary to characterize the difference between drowsy and awake states. To this end, we investigated the RPs produced from RRIs of ECG and PPG. As a plot of visualizing the pattern of recurrence [21], it shows the repetitiveness of state evolution in the phase space of a dynamic system. That is, if the distance between two states in the phase space is very close

(i.e., if the distance is less than a threshold), it is considered that recurrence occurs between two states in the topology space over time. By looking into this plot, it is possible to find the repetitive occurrence of similar sequences of states in the system dynamics.

In this study, we examined three types of RPs: Bin-RP, Cont-RP, and ReLU-RP. The Bin-RP is a binarized recurrence plot where the cells having smaller values than a pre-specified threshold are marked as 1s and others are marked as 0s. The Cont-RP is the non-threshold recurrence plot where each cell indicates the actual distance between two states in the phase space. The ReLU-RP is our newly suggested recurrence plot that can be obtained by filtering Cont-RP with a modified ReLU (rectified linear unit) function. Specifically, each cell  $R_{i,j}$  of the ReLU-RP was determined using the following formula:

$$R_{i,j} = \begin{cases} D & \text{if } D = \text{dist}(s_i, s_j) \geq \varepsilon \\ 0, & \text{otherwise} \end{cases}, \quad (1)$$

where  $D$  is the distance between two embedded states  $s_i$  and  $s_j$  in the phase space, and  $\varepsilon$  is a pre-specified threshold. The choice of the threshold may vary depending on the experimental dataset. If the threshold is too large, most cells in the ReLU-RP will be converted to zero, resulting in the loss of a lot of information from Cont-RP. On the other hand, if the threshold is too small, the resulting ReLU-RP would become similar to Cont-RP. Herein, the threshold was empirically chosen as  $\varepsilon = 0.1$ , which gave the best performance between 0.075 and 0.33.

For the explanation of these RPs, we used an illustrative example to produce the three types of RPs given in Figure 3. Here, the time series data of 25 points were used and the RPs were constructed with an embedding dimension of 2 and a time delay of 1. Figure 3a shows the Bin-RP consisting of only 0s and 1s with thresholding, and Figure 3b shows the Cont-RP in which each cell has actual values of distance without thresholding. Due to the binarization of cell values in the Bin-RP, the two states having a small distance between them are denoted by 1s and the others are denoted by 0s. Thus, as a result, Bin-RP focuses more on the recurrence patterns, mostly represented by diagonal lines in Cont-RP. On the other hand, the ReLU-RP shown in Figure 3c visualizes the de-emphasized recurrence patterns from Cont-RP. By doing so, the ReLU-RP focuses more on non-recurrence patterns, mostly represented by single dots or vertical (or horizontal) lines in Cont-RP.

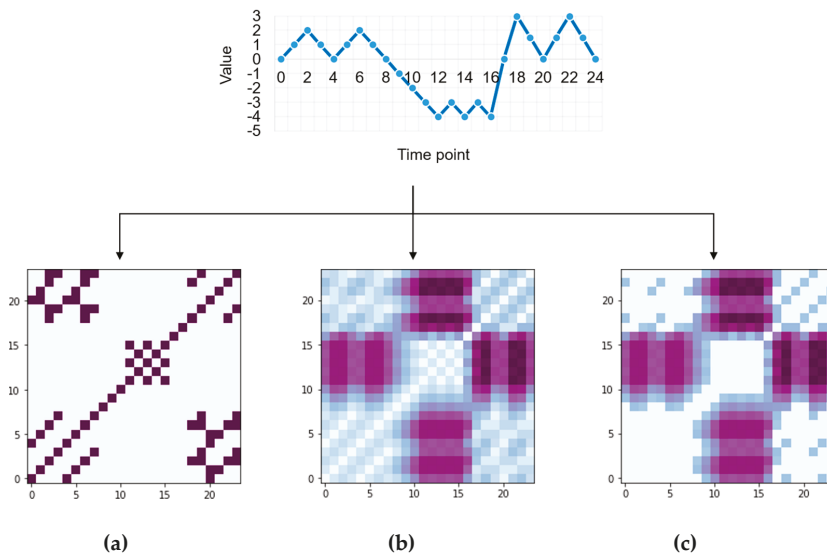
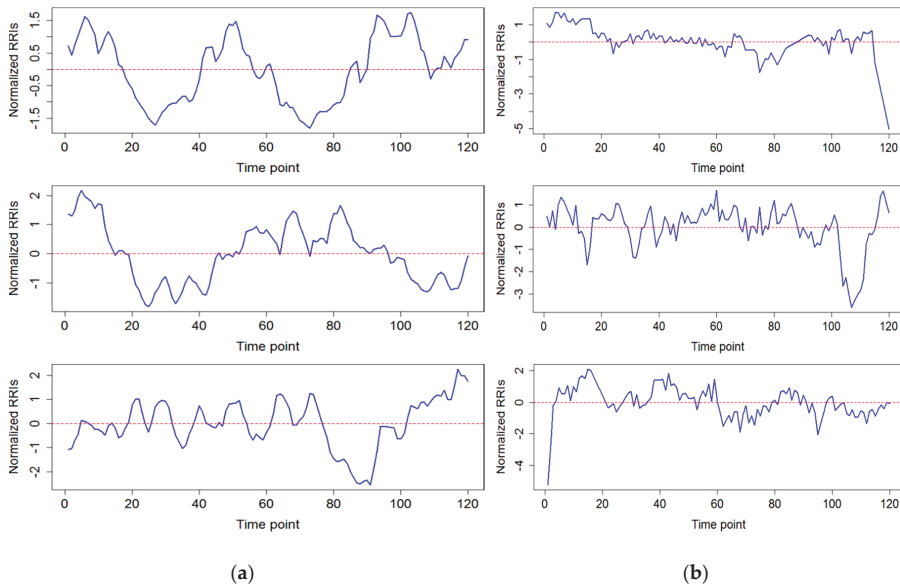


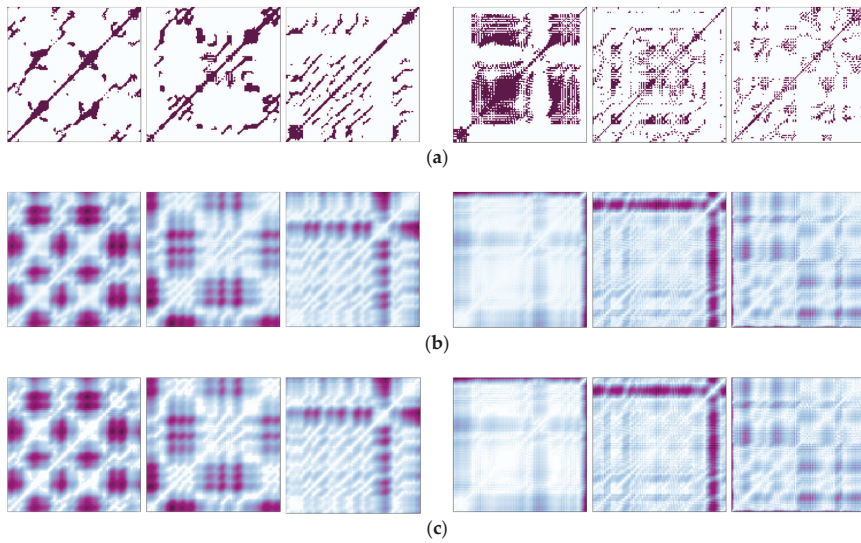
Figure 3. An illustrative example of recurrence patterns (RP): (a) Bin-RP; (b) Cont-RP; (c) ReLU-RP.

According to earlier works [30], a diagonal line shown in RP indicates that the evolution of states in the phase space is similar at different times. On the other hand, a vertical (or horizontal) line in RP indicates that a state does not change or changes very slowly during a certain time. Thus, the Bin-RP focusing on the pattern of diagonal lines results in emphasizing the repetitiveness of a similar state evolution in system dynamics. On the other hand, the ReLU-RP focuses on the pattern of vertical (or horizontal) lines emphasizing the time length in which a state is not significantly changed over time.

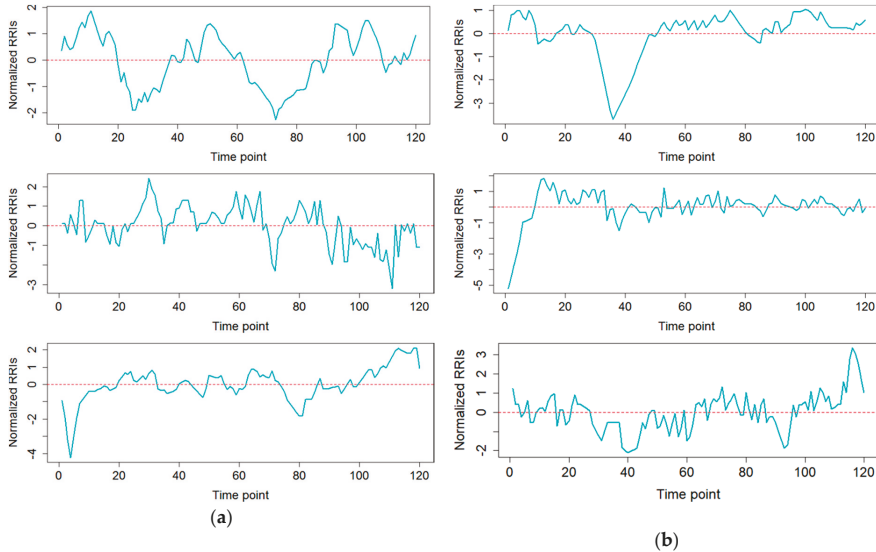
For drowsiness detection, we constructed the three types of RPs from ECG and PPG data, with an embedding dimension of 3 and a time delay of 2. The size of RPs that we produced from the normalized 2-min RRI sample (with 120 intervals) was  $116 \times 116$ . Figure 5 shows the Bin-RP, Cont-RP, and ReLU-RP for drowsy and awake states obtained from ECG data given in Figure 4, while Figure 7 shows the Bin-RP, Cont-RP, and ReLU-RP for drowsy and awake states obtained from PPG data given in Figure 6. From these figures, we can observe that the overall pattern of recurrence presented by diagonal lines shows a clearly distinction between drowsy and awake states in Bin-RPs (Figure 5a and Figure 7a). Moreover, drowsy states seem to have more diagonal lines than awake states, which indicates that the normalized RRIs at drowsy states have a more similar state evolution at different times than those at awake states. Presumably, this is because the normalized RRIs in drowsy samples (Figures 4a and 6a) varied significantly over time with large-scaled repetitiveness, while they rarely changed the overall trend with small-scaled repetitiveness, except for showing a sudden valley at times in awake states (Figures 4b and 6b).



**Figure 4.** Some examples of 2-min R–R interval (RRI) samples obtained from an electrocardiogram (ECG) (strap) sensor: (a) drowsy state; (b) awake state.



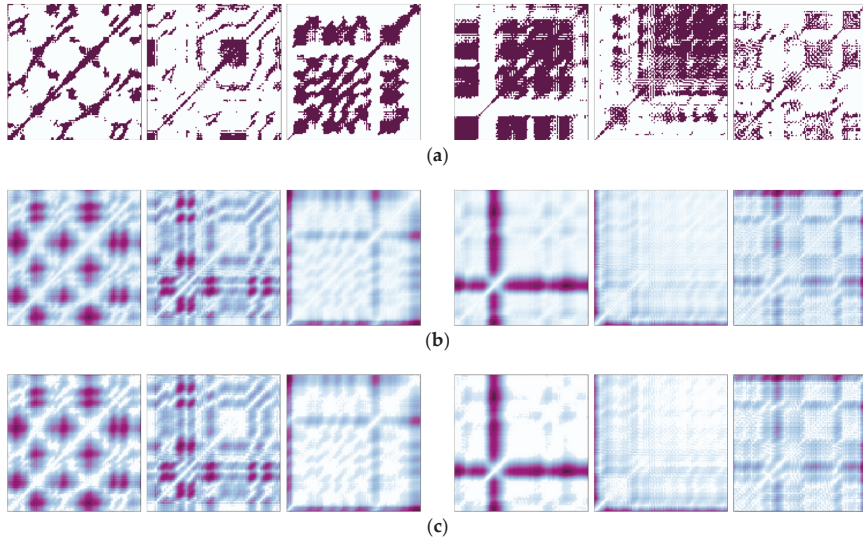
**Figure 5.** Three types of RPs produced from ECG signals (shown in Figure 4) at drowsy (left) and awake (right) states: (a) Bin-RP; (b) Cont-RP; (c) ReLU-RP.



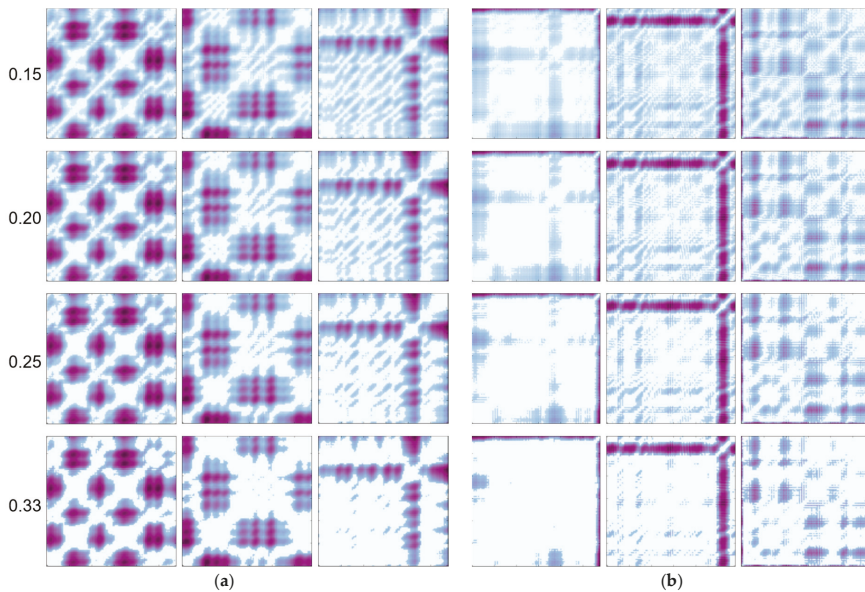
**Figure 6.** Some examples of 2-min RRI samples obtained from a photoplethysmogram (PPG) (band) sensor: (a) drowsy state; (b) awake state.

Such characteristics of drowsy and awake samples were also well captured in the ReLU-RPs, shown in Figures 5c and 7c. The ReLU-RPs emphasize the pattern of vertical (or horizontal) lines from Cont-RP, which indicates that a state does not change or changes very slowly during a certain time. In fact, this indication appears more clearly in the ReLU-RPs of awake samples than in those of drowsy samples, which matches our observation that the normalized RRIs rarely changed the overall trend in awake samples but varied significantly over time in drowsy samples. In Figures 5c and 7c, the ReLU-RPs at drowsy states show a short length of vertical (or horizontal) lines while the ReLU-RPs

at awake states show a long length of vertical (or horizontal) lines. This results in showing better distinguishability between the two states than Bin-RPs. Particularly, as the threshold value of the ReLU function is larger, the effects of diagonal lines are more eliminated from Cont-RP (Figure 8).



**Figure 7.** Three types of RPs produced from PPG signals (shown in Figure 6) at drowsy (left) and awake (right) states: (a) Bin-RP; (b) Cont-RP; (c) ReLU-RP.



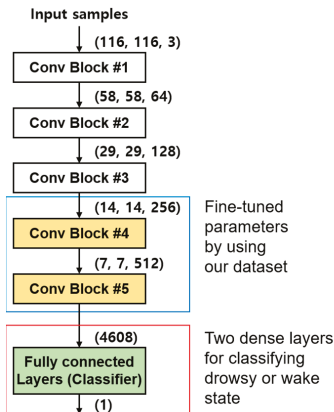
**Figure 8.** A variety of ReLU-RPs from ECG (strap) signals with different thresholds (i.e., 0.15, 0.20, 0.25, and 0.33) at (a) drowsy and (b) awake states.

For completeness, we also explored the Cont-RP that includes all the information about actual distance between any states in the phase space. As seen in Figures 5b and 7b, even if Cont-RPs has more

information than ReLU-RPs, it seems that they are not so helpful in distinguishing between drowsy and awake states. In most cases, Cont-RPs were quite similar to ReLU-RPs, except that, in some cases, Cont-RPs obscured the difference between the two states by keeping more information than ReLU-RPs.

2.4. RP-Based Drowsiness Detection Modeling

For drowsiness detection modeling, each type of RP (i.e., Bin-RP, Cont-RP, and ReLU-RP) was considered as the input to a convolutional neural network (CNN). Unlike conventional approaches using manually extracted and limited features, the CNN allows us to systematically extract distinctive features from the given RP that can help classify between drowsy and awake states. For the CNN model, we fine-tuned the parameters of the VGG16 model [31] using Python. Specifically, the parameters of the initial three convolution blocks were frozen to use the pre-trained filters of the VGG16. The last two convolution blocks were trained to fit our drowsiness dataset. The feature maps learned through convolution layers were then used in two fully connected layers to classify the input into drowsy or awake state (see Figure 9 for details).



(a)

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 116, 116, 3)	0
block1_conv1 (Conv2D)	(None, 116, 116, 64)	1792
block1_conv2 (Conv2D)	(None, 116, 116, 64)	36928
block1_pool (MaxPooling2D)	(None, 58, 58, 64)	0
block2_conv1 (Conv2D)	(None, 58, 58, 128)	73856
block2_conv2 (Conv2D)	(None, 58, 58, 128)	147584
block2_pool (MaxPooling2D)	(None, 29, 29, 128)	0
block3_conv1 (Conv2D)	(None, 29, 29, 256)	295168
block3_conv2 (Conv2D)	(None, 29, 29, 256)	590080
block3_conv3 (Conv2D)	(None, 29, 29, 256)	590080
block3_pool (MaxPooling2D)	(None, 14, 14, 256)	0
block4_conv1 (Conv2D)	(None, 14, 14, 512)	1180160
block4_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block4_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block4_pool (MaxPooling2D)	(None, 7, 7, 512)	0
block5_conv1 (Conv2D)	(None, 7, 7, 512)	2359808
block5_conv2 (Conv2D)	(None, 7, 7, 512)	2359808
block5_conv3 (Conv2D)	(None, 7, 7, 512)	2359808
block5_pool (MaxPooling2D)	(None, 3, 3, 512)	0
flatten_1 (Flatten)	(None, 4608)	0
dense_1 (Dense)	(None, 256)	1179904
dense_2 (Dense)	(None, 1)	257

(b)

Figure 9. Architecture of our fine-tuned VGG16 convolutional neural network (CNN). Here, we used five convolution blocks and two fully connected layers for classification.

2.5. Other Model Development for Performance Comparison

For a comparative study of our suggested model using RPs, we built four different classification models using recurrence quantification analysis (RQA) features. Here, the RQA features included recurrence (REC), determinism (DET), ratio, longest diagonal size (Lmax), average diagonal length (Lmean), divergence (DIV), maximum vertical line length (Vmax), average vertical line length (Vmean),



laminarity (LAM), and entropy (ENTR), which were commonly used in many related studies. Each of these features was statically analyzed to find whether there was a significant difference between drowsy and awake states (Tables 1 and 2). Here, each feature is given with its sample mean and SD, along with a  $p$ -value from the Student's  $t$ -test. As for the four classification models, they include logistic regression (LR), K-nearest neighbor (KNN), support vector machine (SVM), and random forest (RF). All these models were implemented with Python, and the parameters in each model were set with default values. For model evaluation, we conducted 10-fold cross-validation and then analyzed the classification performance with the averaged prediction accuracy, precision, recall, and F-measure.

**Table 1.** Statistical significance of recurrence quantification analysis (RQA) features for awake/drowsy differentiation from electrocardiogram (ECG) signals. Recurrence (REC), determinism (DET), ratio, longest diagonal size (Lmax), average diagonal length (Lmean), divergence (DIV), maximum vertical line length (Vmax), average vertical line length (Vmean), laminarity (LAM), and entropy (ENTR) features were measured.

RQA Feature	Drowsy	Awake	$p$ -Value
REC	0.14 ± 0.04	0.14 ± 0.08	0.0473
DET	0.58 ± 0.16	0.53 ± 0.18	0.0075
Ratio	4.41 ± 0.90	4.05 ± 0.81	0.0002
Lmax	24.69 ± 21.03	23.08 ± 25.11	0.5337
Lmean	5.06 ± 0.90	6.21 ± 5.77	0.0201
DIV	0.06 ± 0.04	0.07 ± 0.04	0.013
Vmax	3.4 ± 7.51	4.88 ± 15.32	0.0305
Vmean	14.45 ± 1.10	15.24 ± 7.92	0.5727
LAM	1.91 ± 0.18	1.92 ± 0.20	0.9066
ENTR	0.7 ± 0.32	0.62 ± 0.49	0.0002

**Table 2.** Statistical significance of RQA features for awake/drowsy differentiation from photoplethysmogram (PPG) signals.

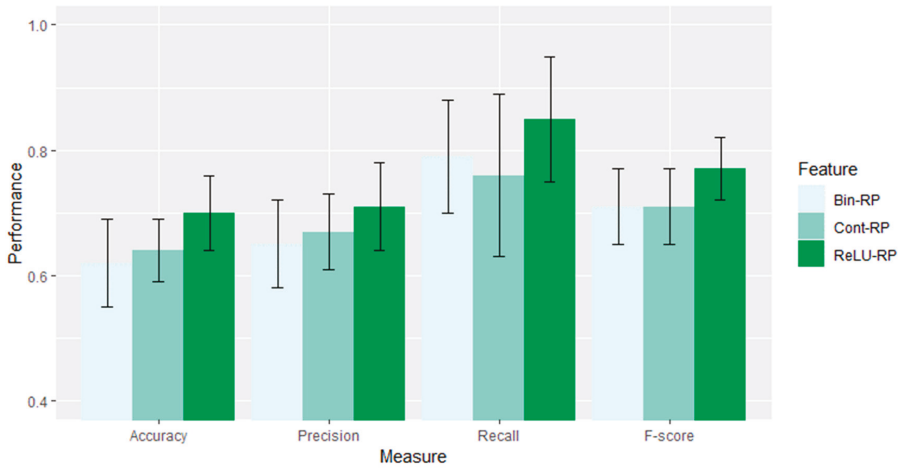
RQA Feature	Drowsy	Awake	$p$ -Value
REC	0.27 ± 0.07	0.26 ± 0.07	0.4597
DET	0.69 ± 0.15	0.65 ± 0.13	0.0127
Ratio	2.57 ± 0.38	2.5 ± 0.34	0.0663
Lmax	42.57 ± 30.08	35.43 ± 25.45	0.0189
Lmean	5.94 ± 1.81	5.67 ± 1.89	0.1965
DIV	0.04 ± 0.02	0.04 ± 0.02	0.0234
Vmax	24.22 ± 10.35	23.61 ± 12.25	0.6299
Vmean	5.04 ± 2.18	4.67 ± 2.18	0.1276
LAM	0.82 ± 0.11	0.8 ± 0.1	0.0709
ENTR	2.26 ± 0.34	2.2 ± 0.34	0.0974

### 3. Results and Discussion

#### 3.1. Evaluation of RP-Based CNN Models for Drowsiness Detection

To evaluate the usefulness of each type of RP for drowsiness detection, we developed three different CNN models, which employed Bin-RP, Cont-RP, and ReLU-RP as input features. Also, for comparison, we developed four classification models (i.e., LR, KNN, SVM, RF) using six significant RQA features (with a  $p$ -value <0.05), and four classification models (i.e., LR, KNN, SVM, RF) using all RQA features.

Figure 10 shows the drowsy/awake classification performance of Bin-RP, Cont-RP, and ReLU-RP used for the CNN-based models using ECG data. Here, it was found that ReLU-RP has better classification ability than Bin-RP and Cont-RP in differentiating between drowsy and awake states. Specifically, it has the best performance in all aspects of prediction accuracy, precision and recall, and F-measure.



**Figure 10.** Comparison of the drowsiness detection performance of Bin-RP vs. Cont-RP vs. ReLU-RP using ECG.

In addition, as shown in Table 3, the CNN models using Bin-RP, Cont-RP, and ReLU-RP showed much higher performance with 62% to 70% accuracy than other existing models using RQA features with 53% to 63% accuracy. Particularly, among the three types of RPs, the proposed ReLU-RP-based CNN model is 70% accurate and superior to other RPs with 62% to 64% accuracy.

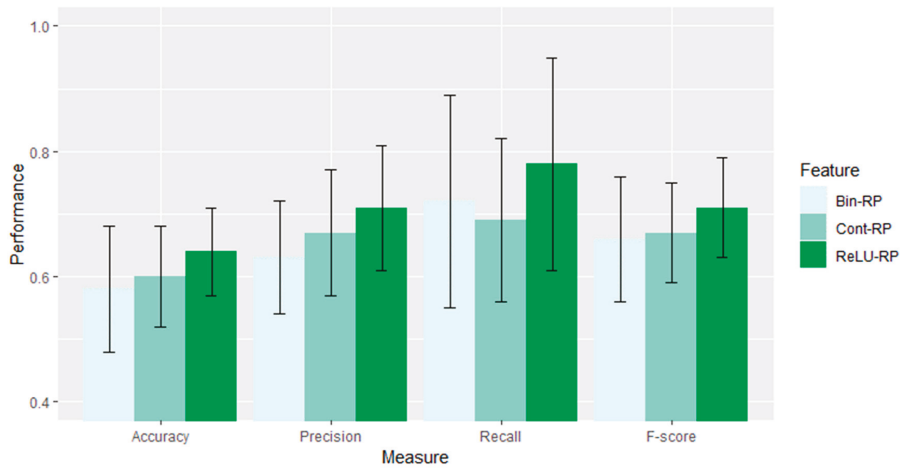
**Table 3.** Drowsy/awake classification performance using ECG strap signals; performance measures were calculated based on 10-fold cross-validation. LR—logistic regression; KNN—K-nearest neighbor; SVM—support vector machine; and RF—random forest.

Feature	Model	Accuracy	Precision	Recall	F-Score	
RQA features	6 significant features	LR	0.62 ± 0.07	0.64 ± 0.07	0.85 ± 0.07	0.73 ± 0.06
		KNN	0.61 ± 0.08	0.66 ± 0.10	0.71 ± 0.10	0.68 ± 0.08
		SVM	0.57 ± 0.05	0.67 ± 0.10	0.53 ± 0.12	0.59 ± 0.09
		RF	0.61 ± 0.07	0.65 ± 0.07	0.77 ± 0.12	0.70 ± 0.07
	All 10 features	LR	0.63 ± 0.10	0.65 ± 0.10	0.81 ± 0.09	0.72 ± 0.09
		KNN	0.54 ± 0.05	0.61 ± 0.08	0.68 ± 0.06	0.64 ± 0.05
		SVM	0.53 ± 0.04	0.62 ± 0.08	0.52 ± 0.11	0.56 ± 0.08
		RF	0.60 ± 0.07	0.63 ± 0.08	0.78 ± 0.09	0.69 ± 0.06
Recurrence plot (RP)	Bin-RP	0.62 ± 0.07	0.65 ± 0.07	0.79 ± 0.09	0.71 ± 0.06	
	Cont-RP	0.64 ± 0.05	0.67 ± 0.06	0.76 ± 0.13	0.71 ± 0.06	
	<b>ReLU-RP</b>	<b>0.70 ± 0.06</b>	<b>0.71 ± 0.07</b>	<b>0.85 ± 0.10</b>	<b>0.77 ± 0.05</b>	

### 3.2. Comparative Analysis of ECG and PPG Sensors for Drowsiness Detection

Previously, we looked primarily at ECG-related results. This section compares them with PPG-related results for drowsiness detection in various aspects.

Firstly, we compared the classification performance between Bin-RP, Cont-RP, and ReLU-RP using CNN models for PPG data (see Figure 11). As in the ECG, ReLU-RP was generally superior to Bin-RP or Cont-RP overall in classifying drowsy and awake states. In addition, the overall classification performance was slightly lower and the standard deviation was greater than that of the ECG, but still maintained 58% to 64% accuracy. In particular, in PPG, ReLU-RP was 64% accurate, demonstrating much higher performance than Bin-RP and Cont-RP with 58% to 60% accuracy. Similar results were also observed with other measures such as precision and recall, and F-measure.



**Figure 11.** Comparison of the drowsiness detection performance of Bin-RP vs. Cont-RP vs. ReLU-RP in PPG.

Also, as shown in Table 4, the CNN models using Bin-RP, Cont-RP, and ReLU-RP performed better with 58% to 64% accuracy than previous models using RQA features with 50% to 59% accuracy. Overall, the PPG band data were usually far more sensitive and unstable than the ECG strap data, and overall model performance was degraded. However, if the RP (especially, ReLU-RP) was used with the CNN model, the result seemed to be reasonably good, and was comparable to that of ECG using RQA features. Above all, models using ReLU-RP were still the best in terms of prediction accuracy, precision and recall, and F-measure.

**Table 4.** Drowsy/awake classification performance using PPG band signals; performance measures were calculated based on 10-fold cross-validation.

Feature	Model	Accuracy	Precision	Recall	F-Score	
RQA features	3 significant features	LR	0.54 ± 0.07	0.67 ± 0.10	0.46 ± 0.10	0.54 ± 0.09
		KNN	0.59 ± 0.07	0.64 ± 0.08	0.72 ± 0.09	0.67 ± 0.08
		SVM	0.57 ± 0.06	0.68 ± 0.10	0.54 ± 0.11	0.59 ± 0.08
	All 10 features	RF	0.56 ± 0.10	0.62 ± 0.10	0.67 ± 0.10	0.64 ± 0.09
		LR	0.58 ± 0.07	0.66 ± 0.07	0.61 ± 0.09	0.63 ± 0.07
		KNN	0.50 ± 0.10	0.56 ± 0.09	0.67 ± 0.12	0.61 ± 0.10
Recurrence plot (RP)	Bin-RP	SVM	0.56 ± 0.06	0.67 ± 0.09	0.55 ± 0.08	0.60 ± 0.06
		RF	0.55 ± 0.07	0.61 ± 0.07	0.69 ± 0.07	0.65 ± 0.06
		CNN	0.58 ± 0.10	0.63 ± 0.09	0.72 ± 0.17	0.66 ± 0.10
	Cont-RP	CNN	0.60 ± 0.08	0.67 ± 0.10	0.69 ± 0.13	0.67 ± 0.08
		ReLU-RP	0.64 ± 0.07	0.71 ± 0.10	0.78 ± 0.17	0.71 ± 0.08

In fact, the significance of our proposed method is in employing wearable ECG (or PPG) sensors (i.e., strap-type ECG sensor or band-type PPG sensor), which are easy to wear while driving, for HRV measurement. As these wearable sensors are convenient to use but very susceptible to noise, it is hard to expect that we can achieve as good a performance as with other specialized equipment. Moreover, a considerable number of samples, acquired from wearable ECG/PPG sensors, had mixed characteristics between drowsy and awake states. Despite such difficulties, our suggested ReLU-RP-based CNN model showed good and stable performance in both cases (70% classification accuracy for ECG and 64% for PPG). In particular, it showed superiority to other conventional models, providing approximately 6–17% better accuracy for ECG and 4–14% for PPG in drowsy/awake classification. From a practical point of view, band-type PPG sensors are much more attractive to use

because of easiness to wear and less intrusiveness. Taking into consideration the proposed method, PPG might be a good alternative to ECG for various applications.

#### 4. Conclusions

In this study, we suggested a robust driver drowsiness detection method that employs HRV measurements acquired from wearable ECG/PPG sensors. Particularly, we examined three types of RPs including Bin-RP, Cont-RP, and ReLU-RP. When ReLU-RP was used as the input to CNN, it could distinguish very well between drowsy and awake states by extracting and learning drowsiness characteristics with the pattern of vertical (or horizontal) lines from RRIs of heartbeats.

To assess the usefulness of the proposed models for realistic drowsiness detection, we did experiments with our in-house dataset that was collected with two body-worn sensors of ECG and PPG in a virtual driving simulation environment. Although this dataset is very limited and controlled, the results of the proposed methods seem to be quite promising. Furthermore, the CNN model using ReLU-RP was superior to previous models for both of ECG and PPG data. Thus, this method is expected to be highly utilizable for detecting driver sleepiness in actual driving situations. Furthermore, our model might be used to classify 2-min RRI samples that include epochs with different states between the first minute and second minute (i.e., awake–drowsy, drowsy–awake samples), by considering them as drowsy states, because they are not in a good condition for safe driving.

In future studies, we plan to investigate other types of physiological measures available in smart bands, such as skin temperature and galvanic skin response, for more robustness of drowsiness detection models. It would also be interesting to use the models with other vehicle-based or behavior-based measures.

**Author Contributions:** Conceptualization, H.L., J.L., and M.S.; methodology, H.L. and J.L.; software, H.L. and J.L.; validation, H.L.; writing, H.L., J.L., and M.S.; visualization, H.L.; supervision, M.S.

**Funding:** This research was supported by the MSIT (Ministry of Science, ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2018-2016-0-00313) supervised by the IITP (Institute for Information & communications Technology Promotion).

**Acknowledgments:** We would like to thank colleagues from the Bio-Intelligence and Data Mining Laboratory of Kyungpook National University, Republic of Korea for assistance in accumulating experimental data. We also would like to thank the reviewers for their valuable comments on the manuscript.

**Conflicts of Interest:** The authors declare no conflicts of interest.

#### Abbreviations

The following abbreviations are used in this manuscript:

HRV	Heart rate variability
RRI	R–R interval
ECG	Electrocardiogram
PPG	Photoplethysmogram
RP	Recurrence plot
RQA	Recurrence quantification analysis
ReLU	Rectified linear unit
CNN	Convolutional neural network

#### References

1. Sigari, M.H.; Fathy, M.; Soryani, M. A driver face monitoring system for fatigue and distraction detection. *Int. J. Veh. Technol.* **2013**, *2013*, 1–11. [[CrossRef](#)]
2. Sun, Y.; Yu, X.B. An innovative nonintrusive driver assistance system for vital signal monitoring. *IEEE J. Biomed. Health Inform.* **2014**, *18*, 1932–1939. [[CrossRef](#)]
3. Murata, A. Proposal of a method to predict subjective rating on drowsiness using physiological and behavioral measures. *IIE Trans. Occup.* **2016**, *4*, 128–140. [[CrossRef](#)]

4. Li, Z.; Li, S.E.; Li, R.; Cheng, B.; Shi, J. Online detection of driver fatigue using steering wheel angles for real driving conditions. *Sensors* **2017**, *17*, 495. [[CrossRef](#)] [[PubMed](#)]
5. Lim, S.; Yang, J.H. Driver state estimation by convolutional neural network using multimodal sensor data. *Electron. Lett.* **2016**, *52*, 1495–1497. [[CrossRef](#)]
6. Reddy, B.; Kim, Y.H.; Yun, S.; Seo, C.; Jang, J. Real-time Driver Drowsiness Detection for Embedded System Using Model Compression of Deep Neural Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017; pp. 438–445. [[CrossRef](#)]
7. Park, S.; Pan, F.; Kang, S.; Yoo, C.D. Driver drowsiness detection system based on feature representation learning using various deep networks. In *Asian Conference on Computer Vision*; Springer: Cham, Switzerland, 2016; pp. 154–164, ISBN 978-3-319-54525-7.
8. Lyu, J.; Yuan, Z.; Chen, D. Long-term multi-granularity deep framework for driver drowsiness detection. *arXiv* **2018**, arXiv:1801.02325.
9. Ngxande, M.; Tapamo, J.R.; Burke, M. Driver drowsiness detection using behavioral measures and machine learning techniques: A review of state-of-art techniques. In Proceedings of the 2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech), Bloemfontein, South Africa, 30 November–1 December 2017; pp. 156–161. [[CrossRef](#)]
10. Shabani, H.; Mikaili, M.; Noori, S.M.R. Assessment of recurrence quantification analysis (RQA) of EEG for development of a novel drowsiness detection system. *Biomed. Eng. Lett.* **2016**, *6*, 196–204. [[CrossRef](#)]
11. Li, G.; Chung, W.Y. Detection of driver drowsiness using wavelet analysis of heart rate variability and a support vector machine classifier. *Sensors* **2013**, *13*, 16494–16511. [[CrossRef](#)] [[PubMed](#)]
12. Nayak, S.K.; Bit, A.; Dey, A.; Mohapatra, B.; Pal, K. A Review on the Nonlinear Dynamical System Analysis of Electrocardiogram Signal. *J. Healthc. Eng.* **2018**, *2018*, 1–19. [[CrossRef](#)]
13. Patel, M.; Lal, K.L.; Kavanagh, D.; Rossiter, P. Applying neural network analysis on heart rate variability data to assess driver fatigue. *Expert Syst. Appl.* **2011**, *38*, 7235–7242. [[CrossRef](#)]
14. Jung, S.J.; Shin, H.S.; Chung, W.Y. Driver fatigue and drowsiness monitoring system with embedded electrocardiogram sensor on steering wheel. *IET Intell. Transp. Syst.* **2014**, *8*, 43–50. [[CrossRef](#)]
15. Chui, K.T.; Tsang, K.F.; Chi, H.R.; Wu, C.K.; Ling, B.W.K. Electrocardiogram based classifier for driver drowsiness detection. In Proceedings of the 2015 IEEE International Conference in Industrial Informatics (INDIN), Cambridge, UK, 22–24 July 2015; Institute of Electrical and Electronics Engineers Inc.: Piscataway, NJ, USA, 2015; pp. 600–603. [[CrossRef](#)]
16. Kim, S.; Choi, B.; Cho, T.; Lee, Y.; Koo, H.; Kim, D. Development of a Classification Model for Driver's Drowsiness and Waking Status Using Heart Rate Variability and Respiratory Features. *J. Ergon. Soc. Korea* **2016**, *35*, 371–381. [[CrossRef](#)]
17. Malik, J.; Lo, Y.L.; Wu, H.T. Sleep-wake classification via quantifying heart rate variability by convolutional neural network. *Physiol. Meas.* **2018**, *39*, 085004. [[CrossRef](#)] [[PubMed](#)]
18. Vicente, J.; Laguna, P.; Bartra, A.; Bailón, R. Drowsiness detection using heart rate variability. *Med. Biol. Eng. Comput.* **2016**, *54*, 927–937. [[CrossRef](#)] [[PubMed](#)]
19. Subramaniam, N.P.; Hyttinen, J. Analysis of nonlinear dynamics of healthy and epileptic EEG signals using recurrence based complex network approach. In Proceedings of the 2013 International IEEE/EMBS Conference on Neural Engineering (NER), San Diego, CA, USA, 6–8 November 2013; pp. 605–608. [[CrossRef](#)]
20. Al-Fahoum, A.S.; Qasaimeh, A.M. A practical reconstructed phase space approach for ECG arrhythmias classification. *J. Med. Eng. Technol.* **2013**, *37*, 401–408. [[CrossRef](#)] [[PubMed](#)]
21. Eckmann, J.P.; Kamphorst, S.O.; Ruelle, D. Recurrence plots of dynamical systems. *Europhys. Lett.* **1987**, *4*, 973–977. [[CrossRef](#)]
22. Marwan, N.; Wessel, N.; Meyerfeldt, U.; Schirdewan, A.; Kurths, J. Recurrence-plot-based measures of complexity and their application to heart-rate-variability data. *Phys. Rev. E* **2002**, *66*, 026702. [[CrossRef](#)] [[PubMed](#)]
23. Nguyen, H.D.; Wilkins, B.A.; Cheng, Q.; Benjamin, B.A. An online sleep apnea detection method based on recurrence quantification analysis. *IEEE J. Biomed. Health Inform.* **2014**, *18*, 1285–1293. [[CrossRef](#)]
24. Smietanowski, M.; Szelenberger, W.; Trzebski, A. Nonlinear dynamics of the cardiovascular parameters in sleep and sleep apnea. In memory of Alberto Malliani (1935–2006)—A brave heart and beautiful mind. *J. Physiol. Pharmacol.* **2006**, *57*, 55–68.

25. Acharya, U.R.; Bhat, S.; Faust, O.; Adeli, H.; Chua, E.C.P.; Lim, W.J.E.; Koh, J.E.W. Nonlinear dynamics measures for automated EEG-based sleep stage detection. *Eur. Neurol.* **2015**, *74*, 268–287. [CrossRef]
26. McDuff, D.; Gontarek, S.; Picard, R.W. Improvements in remote cardiopulmonary measurement using a five band digital camera. *IEEE Trans. Biomed. Eng.* **2014**, *61*, 2593–2601. [CrossRef]
27. Tamura, T.; Maeda, Y.; Sekine, M.; Yoshida, M. Wearable photoplethysmographic sensors—Past and present. *Electronics* **2014**, *3*, 282–302. [CrossRef]
28. Euro Truck Simulator 2. Available online: <https://steamspy.com/app/227300> (accessed on 30 December 2018).
29. Lee, J.; Kim, J.; Shin, M. Correlation Analysis between Electrocardiography (ECG) and Photoplethysmogram (PPG) Data for Driver's Drowsiness Detection Using Noise Replacement Method. *Procedia Comput. Sci.* **2017**, *116*, 421–426. [CrossRef]
30. RECURRENCE PLOTS. Available online: <http://www.recurrence-plot.tk/> (accessed on 30 December 2018).
31. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

Article

# Multi-Object Detection in Traffic Scenes Based on Improved SSD

Xinqing Wang, Xia Hua \*, Feng Xiao, Yuyang Li, Xiaodong Hu and Pengyu Sun

College of Field Engineering, PLA Army Engineering University, Nanjing 210007, China; wwwxxxqqq@126.com (X.W.); xiaofeng199377@163.com (F.X.); lyychqs@163.com (Y.L.); hxd3281008@163.com (X.H.); zzc91292@163.com (P.S.)

\* Correspondence: huaxia120888@163.com; Tel.: +86-176-2603-9818

Received: 9 September 2018; Accepted: 2 November 2018; Published: 6 November 2018

**Abstract:** In order to solve the problem that, in complex and wide traffic scenes, the accuracy and speed of multi-object detection can hardly be balanced by the existing object detection algorithms that are based on deep learning and big data, we improve the object detection framework SSD (Single Shot Multi-box Detector) and propose a new detection framework AP-SSD (Adaptive Perceive). We design a feature extraction convolution kernel library composed of multi-shape Gabor and color Gabor and then we train and screen the optimal feature extraction convolution kernel to replace the low-level convolution kernel of the original network to improve the detection accuracy. After that, we combine the single image detection framework with convolution long-term and short-term memory networks and by using the Bottle Neck-LSTM memory layer to refine and propagate the feature mapping between frames, we realize the temporal association of network frame-level information, reduce the calculation cost, succeed in tracking and identifying the targets affected by strong interference in video and reduce the missed alarm rate and false alarm rate by adding an adaptive threshold strategy. Moreover, we design a dynamic region amplification network framework to improve the detection and recognition accuracy of low-resolution small objects. Therefore, experiments on the improved AP-SSD show that this new algorithm can achieve better detection results when small objects, multiple objects, cluttered background and large-area occlusion are involved, thus ensuring this algorithm a good engineering application prospect.

**Keywords:** machine vision; biological vision; deep learning; convolutional neural network; Gabor convolution kernel; recurrent neural network; enhanced learning

---

## 1. Introduction

Pedestrian and vehicle object detection and recognition in traffic scenes is not only an important branch of object detection technology but also the core technology in the research fields of automatic driving, robot and intelligent video surveillance, both of which highlights its significance in research [1].

The object detection algorithm based on deep learning can be applied to a variety of detection scenarios [2], mainly because of its strong comprehensiveness, activeness and capability of detecting and identifying multiple types of objects simultaneously [3–6]. Among various types of artificial neural network structures, deep convolutional networks, with powerful feature extraction capabilities, have achieved satisfactory results in visual tasks such as image recognition, image segmentation, object detection and scene classification [7].

Faster R-CNN (where R corresponds to “Region”) [8] is the best method based on deep learning R-CNN series object detection. By using VOC2007 + 2012 training set, the VOC2007 test set tests mAP to 73.2% and the object detection speed can reach 5 frames per second. Technically, the RPN [8] network and the Fast R-CNN network are combined and the proposal acquired by the RPN is directly connected

to the ROI (Region of interest) pooling layer, which is a framework for implementing end-to-end object detection in the CNN network.

YOLO (You Only Look Once) [9] is a new object detection method, which is characterized by fast detection and high accuracy. Its author considers the object detection task as a regression problem for object area prediction and category prediction. This method uses a single neural network to directly predict item boundaries and class probabilities, thus rendering end-to-end item detection possible. With its detection speed fast enough, YOLO's basic version can achieve real-time detection of 45 frames/s and the Fast-YOLO can reach 155 frames/s. In comparison with other object detection systems that exist currently, the YOLO object area has a larger positioning error but its false positive of the background prediction is better than the currently existing ones of other systems.

SSD, the abbreviation for Single Shot Multi-box Detector [7], is an object detection algorithm proposed by Wei Liu on ECCV 2016 and is one of the most-often used detection frameworks so far. In comparison with Faster-RCNN [8], it is much faster; and in comparison with YOLO [9], it has a more satisfactory accuracy mean (MAP). Generally speaking, SSD has the following main characteristics:

- (1) It incorporates YOLO's innovative idea of transforming detection into regression, thus making it possible to complete network training at one time;
- (2) Based on the Anchor in Faster RCNN, it proposes a similar Prior box;
- (3) It incorporates a detection method based on the Pyramid Feature Hierarchy, with similar ideas to those of FPN.

Although SSD has achieved higher accuracy and better real-time performance on specific data sets, the training process of the model is not only time-consuming but also heavily dependent on the quality and quantity of training samples. The object is detected by the color and edge information of the image, which undermines the detection effects of those objects that do not have enough image information, particularly when small and weak objects and large-area occlusion of objects are involved. The detection efficiency of the algorithm still needs to be improved to meet the real-time requirements of equipment operation.

According to the characteristics and requirements of pedestrian and vehicle object detection tasks in complex traffic scenes, the following four improvements have been made to the traditional SSD algorithm:

(1) Inspired by the shape of the primary feature convolution kernel which is trained by the deep neural network, we take into account human visual characteristics so as to design and construct a Gabor feature convolution kernel library which is composed of multi-scale Gabor, multi-form Gabor and color Gabor. Aiming at the multi-object self-characteristics to be detected in traffic scenes, we get the optimal feature extraction Gabor convolution kernel set through training and screening and this set is to replace the low-level convolution kernel set which is used by the original feature extraction network model VGG-NET (Visual Geometry Group) for regional basic feature extraction, thus we obtain a new feature extraction network Gabor-VGGNET, which greatly enhances SSD's ability to distinguish multi-object;

(2) In order to solve the problem that SSD has difficulty in detecting small and weak objects with low resolution in large traffic scenes, a dynamic region zoom-in network (DRZN) is proposed after we use enhanced learning and sequential search methods and consider those characteristics and requirements of object detection tasks in large traffic scenes. The network framework greatly reduces the amount of computation by down-sampling images and maintains the detection accuracy of objects with different sizes in high resolution images through dynamic region zoom-in, thus improving significantly the detection and identification accuracy of small and weak objects with low resolution and reducing the missed-alarm rate;

(3) The traditional SSD has the defect that the fixed confidence threshold is not flexible enough. Therefore, the fuzzy threshold method is used here to employ the adaptive threshold strategy so as to reduce the missed-alarm rate and the false alarm rate;



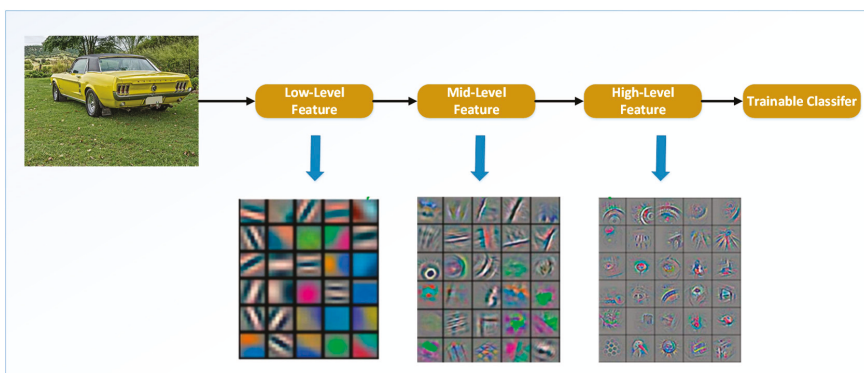
(4) In order to realize real-time video object detection on low-power mobile and embedded devices, a single-image multi-object detection framework is combined with a Long Short Term Memory (LSTM) network to form an interleaved circular convolution which realizes the temporal correlation of network frame level information by using an efficient Bottleneck-LSTM layer to refine and propagate the feature mapping between frames and greatly reduces the network computing cost. Using the timing correlation feature of LSTM and the dynamic Kalman filter algorithm, we can track and recognize the object affected by strong interference such as illumination variation and large-area occlusion.

## 2. Improved Feature Extraction Network Gabor-VGGnet

### 2.1. Gabor Convolution Kernel Design by Simulating Photoreceptive Cells

Convolutional neural network [10] is a special deep neural network model. Its particularity is embodied in two aspects. On the one hand, its connections between neurons are not fully connected and on the other hand, some neurons are in the same layer which means that the weights of the connections are shared (i.e., the same). Its network structure of non-full connection and weight-sharing makes it more similar to biological neural networks, which reduces the complexity of the network model and reduces the number of weights [10].

Deep convolution neural network can combine feature extraction with recognition and can be optimized continuously through back propagation, thus making feature extraction a self-learning process that avoids the limitations caused by manual feature selection. Training a certain convolution layer of the deep convolution neural network is actually training a series of filters to activate these filters' high sensitivity to specific objects and in so doing we can recognize and detect the deep convolution neural network. The filter bank of the first convolution layer of the convolution neural network is used to detect low-order features. With the increase of convolution layer, the features detected by the corresponding filters are more complex. At the beginning of the training, the filters of the convolution layer are completely random and they will not activate any features, that is, they will not be able to detect any features [11]. Then, through the deep convolution neural network visualization toolbox "Yo shin ski/Deep-Visualization-Toolbox" [12], we can obtain the feature convolution kernels of each level by visualizing the CNN model, all of which can be shown in the following Figure 1:



**Figure 1.** Convolution kernel of different rank features extracted by convolution neural network (CNN) model.

The Gabor wavelet is similar to the visual stimulus response of simple cells in the human visual system. It has good characteristics in extracting the local spatial and frequency domain feature information of the object and has strong robustness to the change of the brightness and contrast of the image as well as to the change of the object attitude. Moreover, it demonstrates the most useful local features for object recognition, which is mainly why it is widely used in computer vision and texture

analysis. Compared with other methods, Gabor wavelet transform can deal with fewer data to meet the real-time requirements of the system; on the other hand, wavelet transform is insensitive to light changes and can tolerate a certain degree of image rotation and deformation, both contribute to the improved robustness of the system [13].

In the airspace, a 2-dimensional Gabor filter is the product of a sinusoidal plane wave and a Gaussian kernel function. Gabor filters are self-similar, that is, all Gabor filters can be generated from expansion and rotation of a mother wavelet. Traditional Gabor filters extract relevant features in different directions in different frequency domains. However, when it is practically applied, we found that the two-dimensional Gabor function can enhance the edge and the underlying image features such as peak, valley and ridge contours. The mathematical expression of the two-dimensional Gabor function is shown in Equation (1), Equation (2) is the real part of the function and Equation (3) is the imaginary part of the function.

$$g(x, y, \lambda_1, \theta_1, \psi_1, \sigma_2, \gamma_1) = \exp\left(-\frac{x'^2 + \gamma_1^2 y'^2}{2\sigma_2^2}\right) \exp\left(i\left(2\pi \frac{x'}{\lambda_1} + \psi_1\right)\right) \quad (1)$$

$$g(x, y, \lambda_1, \theta_1, \psi_1, \sigma_2, \gamma_1) = \exp\left(-\frac{x'^2 + \gamma_1^2 y'^2}{2\sigma_2^2}\right) \cos\left(2\pi \frac{x'}{\lambda_1} + \psi_1\right) \quad (2)$$

$$g(x, y, \lambda_1, \theta_1, \psi_1, \sigma_2, \gamma_1) = \exp\left(-\frac{x'^2 + \gamma_1^2 y'^2}{2\sigma_2^2}\right) \sin\left(2\pi \frac{x'}{\lambda_1} + \psi_1\right) \quad (3)$$

In the equation,  $x$  and  $y$  represent the abscissa and ordinate of this pixel,  $x' = x\cos\theta + y\sin\theta$ ,  $y' = -x\sin\theta + y\cos\theta$ ,  $\lambda_1$  represents the sine function wavelength;  $\theta_1$  represents the direction of the kernel function;  $\psi_1$  represents the phase shift;  $\sigma_2$  represents the standard deviation of the Gaussian function;  $\gamma_1$  represents the aspect ratio of the space. The real part can smooth the image and the imaginary part can be used for edge detection. The schematic diagram of the real part of the filter is shown in Figure 2, where the left side is the real component and the right side is the imaginary component.

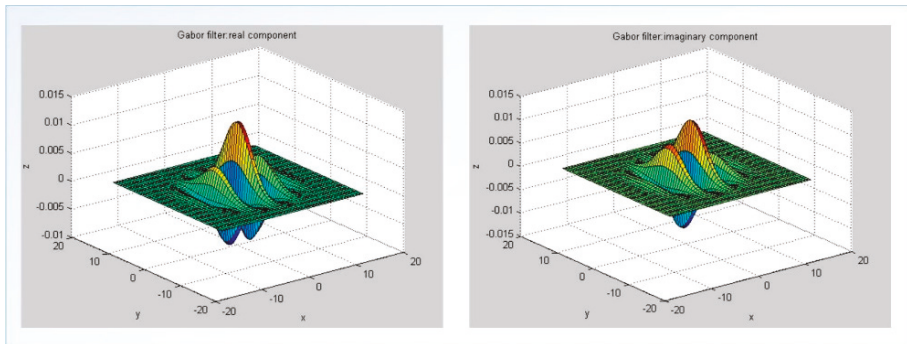


Figure 2. 3D schematic diagram of Gabor filter.

Traditional convolution kernels are generally rectangular or square in shape but our experiments generate the finding that the shape of the Gabor filter convolution kernel has a decisive influence on the edge enhancement effect of the Gabor filter. Each of those different-structured Gabor filters can form an optimal response to the image content that is consistent with its scale, direction, center position, phase and structure type. In order to enable the Gabor filter to extract more complex and rich edge

and texture feature information, we introduce parameters  $k_1, k_2, k_3, k_4$  and  $k_5$  to adjust the Gabor convolution verification part, as shown in Equation (4).

$$g(x, y, \lambda_1, \theta_1, \psi_1, \sigma_2, \gamma_1, k_1, k_2 \dots, k_5) = \exp\left(-\frac{x'^2 + \gamma_1^2 y'^2}{2\sigma^2}\right) \cos\left(2\pi \frac{(k_1 * x'^{k_2} + y'^{k_3} + k_4)^{k_5}}{\lambda_1} + \psi_1\right) \quad (4)$$

Figure 3 shows the partial two-dimensional Gabor filter convolution kernel constructed by Equation (4). The parameters  $k_2, k_3$  and  $k_5$  determine the structure type of the convolution kernel. The parameters  $k_1$  and  $k_4$  determine the direction of the Gabor filter convolution kernel and thereby extract more complex and rich information on edge and texture feature

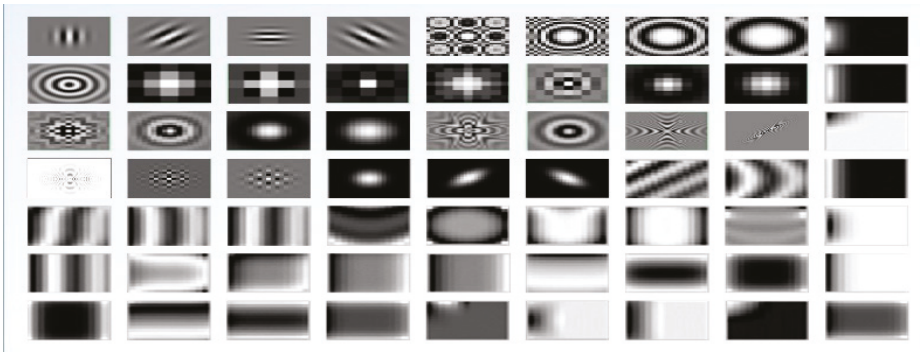


Figure 3. Convolution kernel of two dimensional Gabor filters with various shapes.

Compared with other visual features, color features require less computation and depend less on the viewing angle, direction and size of the image, thereby they have better robustness and lower complexity. RGB space is currently the most commonly used way to express color information. It uses the brightness of the three primary colors of red, green and blue to quantitatively express color and uses R (red), G (green) and B (blue) three-color lights to superimpose each other to realize color mixing. If the proportion of the three colors is different, the colors obtained will be different. The traditional Gabor filter is only used to extract edge, texture and other features from gray images, thus ignoring the color information that plays an important role in image object detection. Inspired by the color convolution kernel trained by the deep convolution neural network, we use the color convolution kernel trained by the neural network as a reference to construct a three-dimensional color Gabor filter through reconstruction to activate the color features of color images.

There are three different kinds of cone cells in the eye, which are sensitive to light of red, green and blue wavelengths respectively. When light waves of different wavelengths enter the eye and are projected on the retina, the brain perceives the color of the scene by analyzing the information input by each cone cell [14]. Imitating the visual mechanism of human eyes, we take a two-dimensional Gabor filter to filter the color feature detection of one color component in a three-dimensional color space, then we construct the relationship among the three color components according to the color characteristics of the object to be extracted and finally we obtain Gabor filters of the other two color components respectively. By combining these three two-dimensional filters [15,16], we can obtain a color Gabor filter too extract the specified object color features.

$$gb_C = gb_R + gb_G + gb_B \quad (5)$$

In Equation (5),  $gb_R$  represents a two-dimensional Gabor filter of color Gabor on the R color channel and the shape of the  $gb_R$  convolution kernel is determined by the above formula.  $gb_G$  and  $gb_B$  are two-dimensional Gabor filters of color Gabor on G and B color channels, respectively.

In the acknowledged receptive field, there are four components: red, green, blue and yellow, with four receptive fields [17]. In order to imitate the perception of color by human visual cells, we use the color convolution kernel trained by neural network as a reference and summarize the mathematical relationship between each color channel by imitation and reconstruction, as is shown in Equations (6) and (7):

$$g^{b_G} = \begin{cases} 255 - g^{b_R, R\&G \text{ or } Y\&B} \\ g^{b_R, R\&G\&B\&Y \text{ or } R\&B} \end{cases} \quad (6)$$

$$g^{b_B} = \begin{cases} 255 - g^{b_R, R\&G\&B\&Y} \\ g^{b_R, R\&G} \\ 255 - g^{b_G, Y\&B} \\ g^{b_G, G\&B} \end{cases} \quad (7)$$

The constraint behind Equation (7) is the color Gabor-sensitive object color. For example, R&G indicates that the object primary color is red and green and Y represents yellow. Figure 4 shows a partial color Gabor convolution kernel:

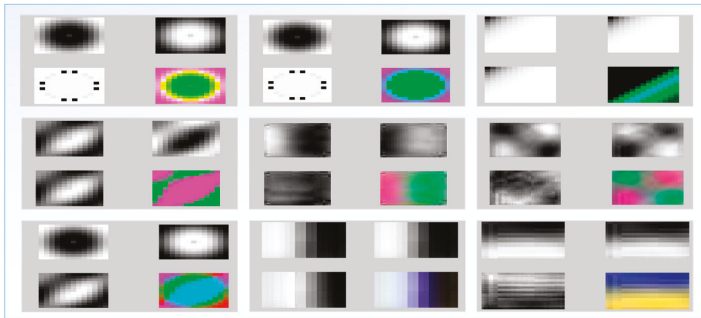


Figure 4. Three-dimensional color Gabor filter convolution kernel.

## 2.2. Intelligent Screening Process for Optimal Gabor Convolution Kernel Set

In the experiment, we use vehicle objects datasets to train the SSD512 [7] model and then assess the convolution kernels' influence on the object recognition rate when the number of convolution kernels vary. After analyzing the experimental results, we set the depth of the convolution layer to be 128 → 256 → 384 → 384 → 384 respectively in order to ensure the highest possible detection accuracy. In the human retina, there are about 6 million to 8 million cone cells and the total number of rod cells is more than 100 million. The ratio of the two is about 10:1 [14]. Therefore, we set the number of two-dimensional Gabor convolution kernels in the first layer of Gabor convolution core group to be 110 and the number of color Gabor convolution kernels to be 18. Then we use the vehicle object dataset in the KITTI dataset [15] to train several different convolution kernels and test their influence on the recognition rate. Through the experiment, we find that the two-dimensional Gabor filter convolution kernel is 3 × 3 in size, the color Gabor filter convolution kernel is 5 × 5 and when the 1 × 1 convolution kernel is added to the Inception structure of the network for dimensionality reduction, the combined filter bank can obtain better object feature sensitivity. In order to effectively improve the detection accuracy of the algorithm as a whole, it is important to construct a reasonable Gabor feature extraction convolution kernel group to extract multiple features with different degrees of discrimination. The screening process of the optimal Gabor convolution kernel group is shown in Figure 5.

First, a two-dimensional Gabor library containing various forms is constructed by means of transforming parameters from Equations (1) and (4); a color Gabor library of the same size can be constructed from Equations (6) and (7); and then a small-scale test image set containing only

“people,” “riders” and “vehicles” respectively (20 for each of the three objects, 60 in total) is constructed. Convolution kernels are randomly and non-repeatedly extracted from two Gabor libraries to form convolution kernel groups; each convolution kernel group rolls up the images of the test set one by one and then corresponding feature maps can be obtained through non-maximum suppression. Then we convert the feature maps into feature vectors through pooling and input the soft max classifier, a traditional SSD detection framework that has trained with small sample data. Thus we can obtain the detection confidence of the test image object and with it we take the average of all confidence of the test set as the evaluation score for the feature extraction effectiveness of the convolution kernel group, so that we obtain the optimal convolution kernel group which is corresponding to the highest evaluation score.

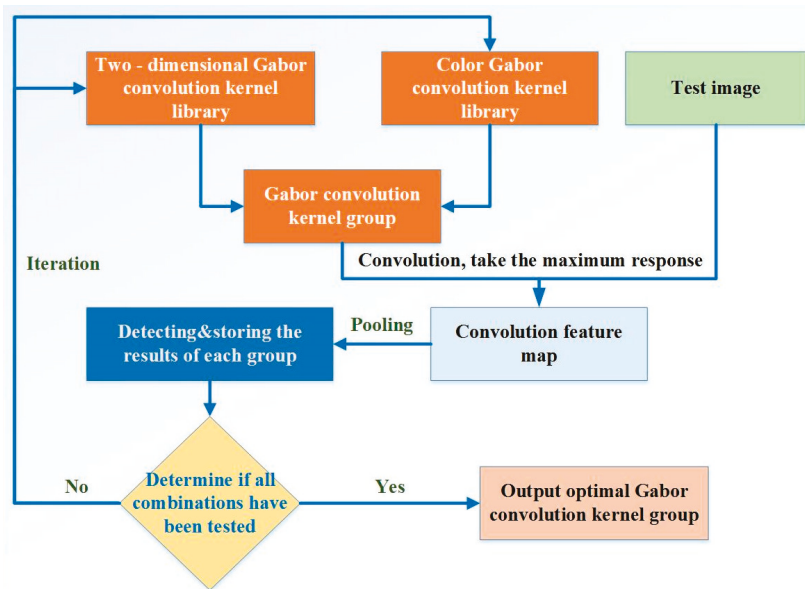


Figure 5. Intelligent screening process for optimal Gabor convolution kernel set.

The scale of Gabor library is reasonably determined by the actual number of convolution kernels in the convolution kernel group. If the scale of Gabor library is too large, then the calculation amount is too large and if it is too small, then it is not representative and the information is not comprehensive enough. The combination of  $m_1$  convolution kernels from the library with a total number  $n_1$  is shown in Equation (8)

$$C_1 = C_{n_1}^{m_1} = \frac{n_1!}{m_1!(n_1 - m_1)!} \quad (8)$$

In order to avoid the data explosion caused by too many combinations and the incompleteness of feature extraction caused by the small-sized database, we randomly divide two-dimensional Gabor convolution kernels into 2 groups each with 10 convolution kernel inside and then divide color Gabor convolution kernels into 2 groups each with 18 convolution kernel inside. The scale of the constructed Gabor library is 180 convolution kernels

### 3. Improvement of Small and Low-Resolution Object Detection Problem

The SSD adopts the feature pyramid structure for detection and has good detection accuracy for small and weak objects. However, the detection effect for low-resolution and weak objects in complex and large traffic scenarios is still not ideal [7].

In order to solve the problem that the existing SSD is difficult to detect small and weak objects with low resolution in complex large scenes, this paper proposes a dynamic region zoom-in network (DRZN), which reduces the calculation of object detection by down-sampling the images of high resolution large scenes while maintaining the detection accuracy of small and weak objects with low resolution in high resolution images through dynamic region zoom and the effect of improving the detection and recognition accuracy is obvious. The detection is performed in a coarse-to-fine manner. First, the down-sampled version of the image is detected and then the areas identified as likely to improve the detection accuracy are sequentially enlarged to higher resolution versions and then detected. The method is based on enhanced learning and consists of an amplification precision gain regression network [16] (R-net) and a Zoom-in region selection algorithm. The former learns the correlation between coarse detection and fine detection and predicts the precision gain after region amplification and the latter performs learning and predicts the result before it dynamically selects regions to be amplified.

First, the down-sampled version of the image is roughly detected in order to reduce the amount of computation and to improve the operation efficiency. Then, the regions where low-resolution small objects may exist are sequentially selected for amplification and for analysis to ensure the recognition accuracy of the low-resolution small objects. We use the reinforcement learning method to model the amplification reward in terms of detection accuracy and calculation efficiency and then we dynamically select a series of regions to be amplified to high resolution for analysis. Reinforcement learning (RL) is a branch of machine learning. Compared with the typical problems generated from supervised-learning and unsupervised-learning of machine learning, the biggest feature of reinforcement learning is learning from interaction. In the interaction between agent and environment, agent learns knowledge continuously as it is motivated by the reward or punishment obtained and adapts to the environment more. The RL learning paradigm is very similar to our human learning process and therefore RL is regarded as an important way to achieve universal AI.

RL is a popular mechanism for learning sequential search policies, as it allows models to consider the effect of a sequence of actions rather than individual ones. The current machine learning algorithms can be divided into three types: supervised learning, unsupervised learning and reinforcement learning.

In many other machine-learning algorithms, the learner is just learning how to do, while RL can learn which action to choose so as to get the maximum reward in a specific situation. In many scenarios, the current action will not only affect the current rewards but also affect the subsequent status and a series of rewards.

The three most important characteristics of RL are:

- (1) It is usually a closed-loop form;
- (2) It does not directly indicate which actions to choose;
- (3) A series of actions and reward signals will affect the action that follows. Existing works have proposed methods to apply RL in cost sensitive settings. We follow the approach and treat the reward function as a linear combination of accuracy and cost.

The overall framework of the algorithm is shown in Figure 6.

The algorithm mainly consists of two mechanisms: (1) the first one is to learn the statistical relationship between the coarse detector and the fine detector, so as to predict which areas need to be amplified with the given output of the coarse detector; (2) The second mechanism is to analyze the sequence of regions at high resolution when the coarse detector output and the region that needs to be analyzed by the fine detector are already given.

The strategy proposed and used in this paper can be expressed as a Markov decision process [17]. At each step, the system first observes the current state, estimates the potential cost-perceived rewards for different actions and selects actions with the greatest long-term cost-perceived rewards. The elements include: action, state set, cost-aware reward.

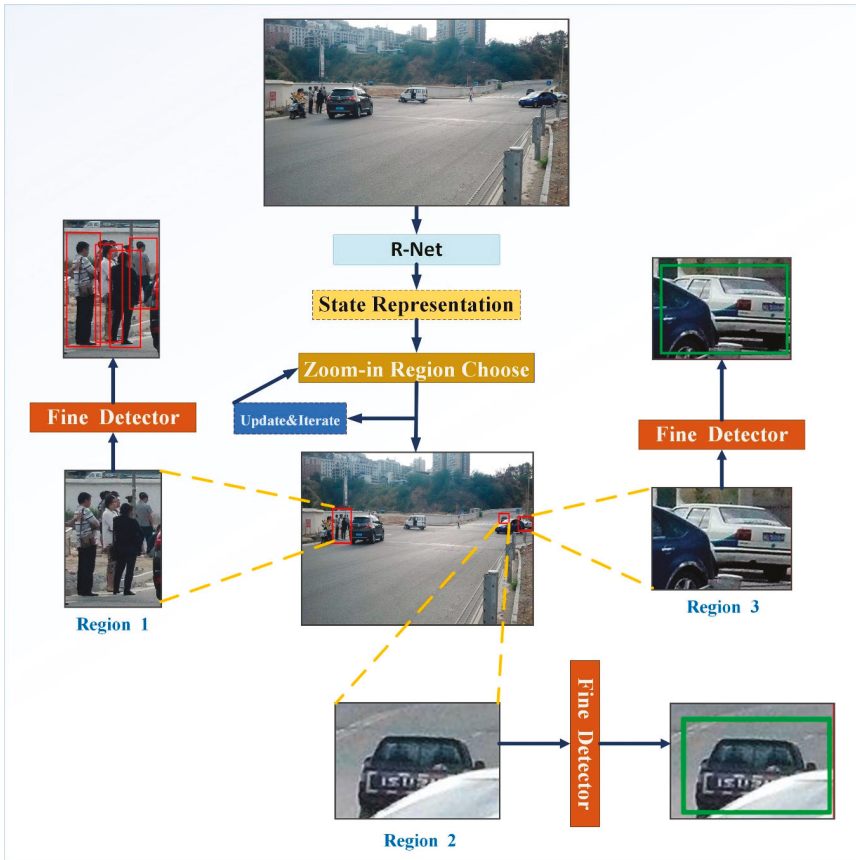


Figure 6. Network Architecture of Dynamic Region Enlargement Algorithm.

Action. The algorithm analyzes regions with high magnification returns in high resolution. Here, the action refers specifically to the selection of an area to be analyzed with high resolution. Each action can be represented by a vector  $(x, y, w, h)$ , among which  $(x, y)$  represents the location of the specified area and  $(w, h)$  indicates the size of the specified area. In each step, the algorithm make assessment of a set of potential actions (list of rectangular regions) based on potential long-term rewards.

State set. It represents the encoding of two types of information: the prediction accuracy gain of the area to be analyzed and the history of the area that has been analyzed with high resolution (the same area should not be amplified by multiple times). We design an amplification precision gain regression network (R-net) to learn the information accuracy gain map (AG map) as a representation of the state. The AG map has the same width and height as the input image. The value of each pixel in the AG map is an estimation of how much detection accuracy can be improved by including that pixel in the input image. Therefore, the AG map provides a detection accuracy gain for selecting different actions. After the action is taken, the value corresponding to the selected region in the AG map is correspondingly reduced, so the AG map can dynamically record the action history.

Cost-aware reward. The state encodes the prediction accuracy gain when the amplification of every image sub-region is involved. In order to maintain high precision with a limited amount of computation, we define a loss-return function, as Equation (9) shows. Given the state and

action, the loss-reward function scores each action (zoom area) by considering cost increments and precision improvements.

$$R(s_{states}, a_{actions}) = \sum_{k \in a_{actions}} |g_k - p_k^l| - |g_k - p_k^h| - \lambda_2 \frac{b_1}{B} \quad (9)$$

Here,  $k$  in the action indicates that the object  $k$  is included in the region selected by the action,  $p_k^l$  and  $p_k^h$  indicates the object detection score for the same object coarse detector and fine detector and  $g_k$  is the corresponding object real label. The variable  $b_1$  represents the total number of pixels in the selected area, representing the total number of pixels in the input image. The first term in the formula indicates an increase in detection accuracy. The second term indicates the cost of amplification. The balance between accuracy and calculation is controlled by  $\lambda_2$  parameters.

The Amplified Precision Gain Regression Network (R-Net) predicts the precision gain of amplification over a particular region based on the coarse detection results. R-Net trains on the coarse and fine test data pairs so that it can observe how they relate to each other in order to learn the appropriate precision gain relationship [7].

The two SSDs are trained respectively on a high resolution fine image training set and on a low resolution coarse image training set and then it is used as coarse and fine detectors respectively. We apply two pre-trained detectors to a set of training images and obtain two sets of image detection results: low resolution detection  $\{(d_i^l, p_i^l, f_i^l)\}$  in down-sampled images and high resolution detection  $\{(d_j^h, p_j^h)\}$  in high resolution versions of each image. Here,  $d$  is the detection bounding box,  $p$  is the probability of being the object and  $f$  is the corresponding detected feature vector. We use the superscripts  $h$  (High) and  $l$  (Low) to represent high resolution and low resolution (down-sampled) images.

In order to enable the model to differentiate whether or not the high-resolution detection can improve the overall detection result, we introduce a matching layer to correlate the detection results produced by the two detectors. In this layer, if we find that the possible object in the down-sampled image and the possible object in the high resolution image have a sufficiently large intersection  $IoU(d_i^l, d_j^h)$  ( $IoU > 0.5$ ), then the definitions of  $i$  and  $j$  correspond with each other. We match the rough detection scheme and the fine detection scheme according to the rules and thus generate a set of correspondence between them [7].

Given a set of correspondences  $\{(d_k^l, p_k^l, p_k^h, f_k^l)\}$ , we can estimate the amplification accuracy gain of the coarse detection. The detector can only handle objects within a certain range, so applying the detector to a high resolution image does not necessarily produce the best accuracy. For example, if the detector is primarily trained on a small object dataset, the detection accuracy of the detector for larger objects is not high. Therefore, we use  $|g_k - p_k^l| - |g_k - p_k^h|$  to measure which test result (rough or fine) is closer to the fact and in the equation  $g_k \in \{0, 1\}$  is a measure of the real tag. When the high resolution score is closer to the basic fact than the low resolution score, the function indicates that the object is worth zooming in. Otherwise, applying a coarse detector on the down-sampled image may result in higher precision, so we should avoid zooming in on the object. We use the correlation regression (CR) layer to estimate the amplification accuracy gain of the object  $K$ , such as Equation (10).

$$\min_{W_1} \left( |g_k - p_k^l| - |g_k - p_k^h| - \phi(W_1, f_k^l) \right)^2 \quad (10)$$

Here,  $\phi$  represents the regression function and  $W_1$  represents the parameter set. The output of this layer is the estimated accuracy gain. The CR layer consists of two fully connected layers, with 4096 cells in the first layer and only 1 output cell in the second layer.



An AG map (Accuracy Gain map) can be generated based on the learning accuracy gain of each object. We assume that each pixel within the candidate frame has an equal contribution to its accuracy gain. Therefore, the AG map generated is:

$$AG(x,y) = \begin{cases} \alpha \frac{\phi(\hat{W}, f_k^l)}{b_k} & \text{if } (x,y) \text{ in } d_k^l \\ 0 & \text{otherwise} \end{cases} \quad (11)$$

In Equation (11),  $(x,y)$  indicates that the point  $(x,y)$  is within the bounding box  $d_k^l$ ,  $b_k$  indicates the number of pixels contained in  $d_k^l$ ,  $\alpha$  is a constant and  $\hat{W}$  indicates the estimated parameters of the CR layer. The AG map is used as a state representation, which naturally contains information about the quality of the rough detection. After zooming in and detecting the area, all values in the area are set to be 0 so as to prevent future scaling in the same area. The R-net structure of the amplification precision gain regression network is shown in Figure 7.

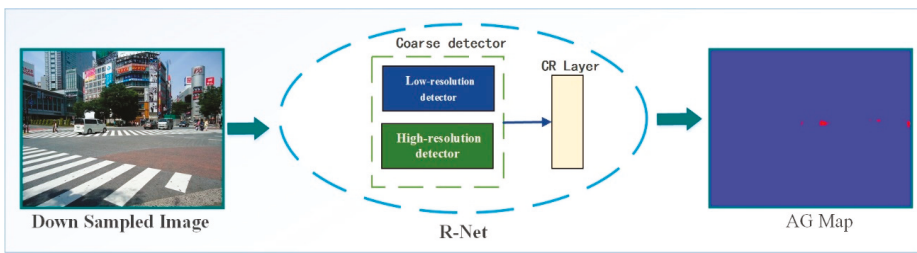


Figure 7. R-Net network framework.

Through R-net we obtain the AG map and the value of each pixel in the AG map is an estimate of how much detection accuracy can be improved by including that pixel in the input image. Therefore, the AG map provides a detection accuracy gain for the selection of different actions. After the action is taken, the value corresponding to the selected region in the AG map is correspondingly reduced, so the AG map can dynamically record the action history. According to AG map, we propose a dynamic zooming region selection algorithm. The specific algorithm flow is shown in Figure 8.

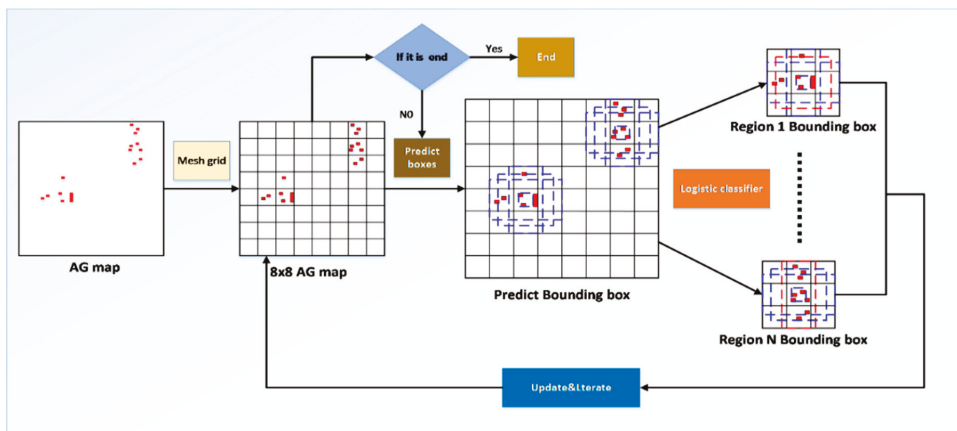


Figure 8. Dynamic zoom area selection process.

Firstly, we divide the AG map into equal-area rectangular regions according to the  $8 \times 8$  grid, count the sum of the pixel values in each rectangle, set the threshold and select the regional center

block. The  $3 \times 3$  rectangles that center on the center block of each region constitute the enlarged screening area. If an enlarged screening area has several rectangular regions that satisfy the threshold value of the pixel value, then the one with the largest pixel value is taken as the center of the region. If the center of the region is taken on the side of the large square, the  $3 \times 3$  is formed by adding blank squares of the same size. In the enlargement and screening area, we take the central point of enlarged screening area as the center. Then four prediction bound boxes with different ratios of length, width and breadth are constructed and the best enlargement area bounding box can be selected after we compare the structural indicators (pixel values, ratios) of each prediction bounding box.

The total pixel value  $s_{ump}x_i$  in the rectangular area  $rtg_i$  in the grid-divided AG map is shown in Equation (12).

$$s_{ump}x_i = \sum_{j \in rtg_i} px_j \tag{12}$$

Here, the  $px_j$  represents the pixel value of the  $j$ th pixel in the  $rtg_i$  region. The larger the  $s_{ump}x_i$  value, the larger the amplification gain of the rectangular region  $rtg_i$  and the region with high magnification gain is taken as the center to make it correspond to the human eye's processing of the block domain. We adaptively select the pixel value threshold by the second-order difference method to complete the initial screening of the regional center block. The second-order difference can represent the magnitude of the change in the discrete array and can be used to determine the threshold in a set of pixel values. By detecting an AG map, 64 candidate regions are obtained by default. Finally, each candidate region obtains an overall pixel value  $s_{ump}x_i$  for indicating the amplification gain. Therefore, a total of  $64 \times 1$  arrays can be obtained and elements less than 0.1 are discarded. To have no object, get an  $n \times 1$  array C. Let the function of estimating the slope of the  $s_{ump}x_i$  from decreasing by  $f(g)$ , see Equation (13).

$$f(C_k) = \frac{(C_{k+1} - C_k) - (C_k - C_{k-1})}{C_k}, k = 2, 3, \dots, n - 1 \tag{13}$$

Then, take the  $C_k$  as the  $s_{ump}x_i$  threshold of the AG map image and this  $C_k$  is obtained when  $f(C_k)$  takes the maximum value.

In order to reduce the calculation amount of area enlargement and fine detection, to effectively improve the efficiency and real-time of the algorithm and to ensure the better tolerance of the selected area, we form an enlarged screening area with  $3 \times 3$  rectangles centering on each regional central block. If the same zoom-in filter area has multiple rectangular areas that satisfy the pixel value threshold condition, the one with the largest pixel value is taken as the center of the area.

We use the center point of the enlarged screening area as the center position to predict the prediction bounding boxes of six fixed sizes according to different aspect ratios. The area of the enlarged screening area is  $S_Z$ . The area of each predicted bounding box is shown in Equation (14).

$$s_k = s_{min} + \frac{s_{max} - s_{min}}{m - 1}(k - 1), k = 1, 2, \dots, 5 \tag{14}$$

Here,  $s_{min} = 0.1 \times S_Z$ ,  $s_{max} = 0.7 \times S_Z$ ,  $m = 5$ . We give different aspect ratios for different prediction bounding boxes, such as Equation (15).

$$a_r = \frac{W}{H}, a_r \in \left\{ 1, 2, 3, \frac{1}{2}, \frac{1}{3} \right\} \tag{15}$$

In the equation,  $W$  and  $H$  respectively indicate the width and length of the bounding box. Then, the width and length of the bounding box are predicted to be  $H_k = \sqrt{s_k/a_r}$ ,  $W_k = \sqrt{a_r \cdot s_k}$  respectively. When  $a_r = 1$ , there is also a prediction bounding box with a scale of 6, that is,  $s'_k = \sqrt{s_k \cdot s_{k+1}}$ , so there

are a total number of 6 prediction bounding boxes. For any bounding box,  $b_l$ , we calculate the total pixel value in the box  $s_{ump}x_i$ , such as Equation (16).

$$s_{ump}x(b_l) = \sum_{i \in b_l} px_i, \quad l = 1, 2, 3, 4 \tag{16}$$

$$S(b_l) = W \times L \tag{17}$$

$W$  and  $L$  represent the width and length of the box respectively. The proportion  $P$  of pixels with high amplification income in the region  $b_l$  is shown in Equation (18):

$$P(b_l) = \frac{pn_1}{pn_2} \tag{18}$$

$Pn_1$  represents the total number of pixels in the  $b_l$  region with pixel gains (pixel values with pixel values greater than 0.1) and  $Pn_2$  represents the total number of pixel points in the  $b_l$  region. That is, for each prediction bounding box,  $b_l$  has a feature vector  $(x, y, s_{ump}x_i, W, L, P)$  and  $x$  and  $y$  respectively represent the abscissa and ordinate of the center point of  $b_l$ .

We use a manually calibrated training sample to train a Logistic classifier [18] to evaluate the frame selection effect of each prediction bounding box. We classify the evaluation results into two categories: a prediction bounding box that satisfies the amplification requirements and a prediction bounding box that does not meet the amplification requirements.

For the input prediction bounding box,  $b_l(x, y, s_{ump}x_i, W, L, P)$ , Logistic classifier introduces weight parameter  $\theta = (\theta_1, \theta_2, \dots, \theta_6)$ , then weights the attributes in  $b_l$  to obtain  $\theta^T b_l$  and then it introduces a logistic function (sigmoid function) to get the function  $h_\theta(b_l)$ , as is shown in Equation (19):

$$h_\theta(b_l) = \frac{1}{1 + e^{-\theta^T b_l}} \tag{19}$$

Thus, we get the estimation function  $P(y | b_l; \theta)$ , as is shown in Equation (20):

$$P(y|b_l; \theta) = \begin{cases} h_\theta(b_l); & y = 1 \\ 1 - h_\theta(b_l); & y = 0 \end{cases} \tag{20}$$

It means the probability that the label is  $y$  when the test sample  $b_l$  and the parameter  $\theta$  are determined.

After evaluating the frame selection effect of each prediction bounding box by Logistic classifier, we can obtain a corresponding frame selection evaluation score for each prediction bounding box and then perform a non-maximum value suppression to obtain the final prediction as the final Enlarged bounding box.

After completing the selection of the enlarging bounding box, we set all the pixel values in the enlarged screening area to 0 so as to avoid the inefficiency caused by repeated selection. Then we update the corresponding area of the AG map and detect whether the AG map has been highly enlarged profit area for detection (whether the total value of the AG map pixel is 0) has been detected; if yes, then we complete the detection, if no, then we continue to repeat the detection process.

Before the original image of the obtained fine detection candidate region is sent to the fine detector for detection, the bilinear interpolation is first performed to be enlarged to the minimum size of the candidate detection region of fine detector detection (This paper sets the minimum candidate region to be  $10 \times 10$ ).

#### 4. Confidence Determination by Adaptive Threshold

In the final stage of SSD classification, we use Soft-max to classify the candidate region which will obtain the confidence level (i.e., the probability of belonging to each category) belonging to each

category. When the confidence level belonging to a certain class is higher than the set threshold, the candidate region is used. It is judged as the object of this kind and if the same candidate area has multiple categories whose confidence level is higher than the threshold, the highest one is taken [19]. Aiming at the insufficiency of SSD to detect the fixed confidence threshold, the fuzzy adaptive threshold method [20] is used to make adjustment to the adaptive threshold strategy so as to reduce the missed alarm rate and false alarm rate.

The fuzzy degree is determined by the fuzzy rate function. When the fuzzy rate is the lowest, the segmentation effect is the best. Among them, the fuzzy rate is related to the membership function and the basic idea of fuzzy mathematics is the idea of membership degree [18]. By default,  $N$  candidate regions obtained by detecting an image are sent to SSD and finally  $M$  confidence levels are obtained for each candidate region to represent  $M$  categories. So  $N$  arrays of  $M \times 1$  size can be obtained altogether. The maximum value in each array is taken out and sorted from large to small and the values less than 0.1 among them are discarded (if all the  $n$  values are less than 0.1, it is determined that there is no object), resulting in an array  $C$  of  $n \times 1$ .  $\mu(x)$  is the membership function and  $\mu(C_k)$  is the membership of the region in the array  $C$  where the confidence level is  $C_k$ . The ambiguity rate  $\gamma(C)$  of array  $C$  is the ambiguity measure parameter for array  $C$ . Here  $h(C_k)$  represents the number of elements in the array  $C$  whose confidence is  $C_k$  and the ambiguity rate  $\gamma(C)$  of the array  $C$  is defined in Equation (21).

$$\gamma(C) = \frac{2}{n} \sum_{k=0}^{n-1} T(C_k)h(C_k) \tag{21}$$

In this equation, the ambiguity rate  $\gamma(C)$  of the array  $C$  depends on the membership function  $\mu(x)$ .

$$\mu(x) = \begin{cases} 0, & 0 \leq x \leq q - \Delta q \\ 2 \left[ \frac{(x-q+\Delta q)}{2\Delta q} \right]^2, & q - \Delta q \leq x \leq q \\ 1 - 2 \left[ \frac{(x-q+\Delta q)}{2\Delta q} \right]^2, & q < x \leq q + \Delta q \\ 1, & q + \Delta q < x \leq C_n \end{cases} \tag{22}$$

Then  $\mu(x)$  is determined by the window width  $c = 2\Delta q$  and the parameter  $q$ . Once the window width is selected,  $\gamma(C)$  is only related to the parameter  $q$ . The solution process of the fuzzy threshold method is to preset the window width and the coefficient is often set to be 0.3. By changing  $q$  to make the membership function  $\mu(x)$  slide over the confidence interval  $[C_0, C_{n-1}]$ , the blur rate curve is obtained by calculating the blur rate  $\gamma_q(C)$ , the valley point of the curve is  $q$  that makes  $\gamma_q(C)$  get a minimum value, which is the adaptive threshold value.

### 5. Video Multi-Object Detection Technology Based on Recurrent Neural Network

This section examines strategies for building video detection models by increasing time perception while the operating speed and low computational resource consumption are both ensured. Video image data contains a variety of time cues that can be expanded to achieve more accurate and stable object detection than a single image. Therefore, the detection result information from the earlier frame can be used to refine the prediction result at the current frame. Since the network can detect objects in different states across frames, the network prediction results will also have higher confidence as the training time progresses, thus effectively reducing the instability in single image object detection.

In order to realize real-time video object detection on low-power mobile and embedded devices, a single-image multi-object detection framework is combined with a Long Short Term Memory (LSTM) network to form an interleaved circular volume. The product structure, by using an efficient Bottleneck-LSTM layer [21] to refine and propagate the feature mapping between frames, achieves the temporal correlation of network frame-level information and greatly reduces the network computing cost. By using the timing correlation feature of LSTM and the dynamic Kalman filter algorithm [22],

we can track and recognize the object affected by strong illumination and large-area occlusion in the video is realized.

The Recurrent Neural Network (RNN) solves this problem better. They are networks with loops that allow information to persist. One of the advantages of RNN is that they can relate previous information to current tasks, such as using previous video frames to help detect current video frame [23]. Long-Short Term Memory (LSTM) is a special RNN designed to avoid long-term dependencies. A method of combining convolutional LSTMs into a single image detection framework is proposed as a means of propagating frame level information across time. However, the simple integration of LSTMs leads to a large amount of computation and prevents the network from running in real time. To solve this problem, a Bottleneck-LSTM was introduced, which uses the features of deep separable convolution and Bottleneck design principles to reduce computational costs. Figure 9 shows the network structure of the LSTM-SSD. Multiple convolutional LSTM layers are inserted into the network, each of which propagates and refines the feature map according to a certain proportion.

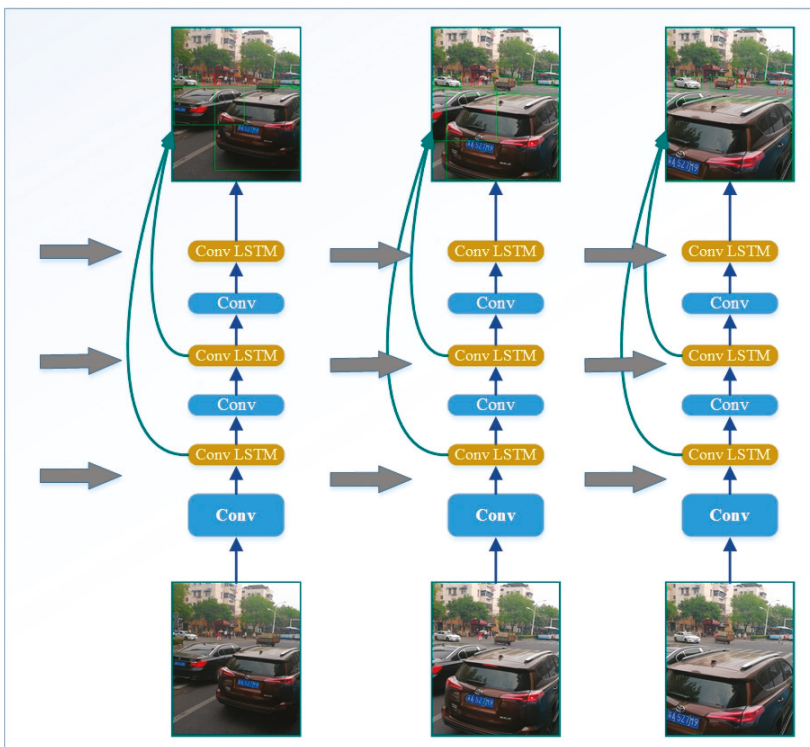


Figure 9. Mobile video object detection framework based on time-aware feature mapping.

A layer of Conv-LSTM in the network receives the feature map of its previous Conv-LSTM and then we form a new feature map by combining the current map obtained in the above process with the one transmitted from the previous frame and then we predicts the detection result and transmits the feature map to the following Conv-LSTM and convolution layer. The output of Conv-LSTM will replace all the previous feature maps in all subsequent calculations and continue the detection task. However, the simple integration of LSTMs can lead to a large amount of computation and thus prevents the network from running in real time. To solve this problem, we introduce a Bottleneck-LSTM that takes advantage of its deep separable convolution as well as of the Bottleneck design principles to reduce computational costs.

The video data is regarded as a sequence  $V$  composed of multiple frames of images,  $V = \{I_0, I_1, \dots, I_n\}$ . The task of the algorithm is to obtain the frame-level detection result  $D$ ,  $D = \{D_0, D_1, \dots, D_n\}$ ,  $D_k$  represents the detection result of the image frame  $I_k$ , which includes the position of a series of detection frames and the recognition confidence of each object. We consider constructing an online learning mechanism so that the detection result  $D_k$  can be predicted and corrected by the image frame  $I_{k-1}$ , such as Equation (23)

$$F(I_t, s_{t-1}, AG_{t-1}) = (D_t, s_t, AG_t) \tag{23}$$

Here  $s_k = \{s_k^0, s_k^1, s_k^2, \dots, s_k^{m-1}\}$  refers to the vector of feature maps which describes the image of the  $k$ th frame of the video;  $AG_k = \{AG_k^0, AG_k^1, AG_k^2, \dots, AG_k^{m-1}\}$  represents an AG map describing the image of the  $k$ th frame of the video. We can construct a neural network with  $m$  layer LSTM convolution layer to approximately realize this function. This neural network takes each feature map and amplification precision enhancement map  $AG_{t-1}$  in the feature map vector  $s_{t-1}$  as the input of LSTM convolution layer and thus we can obtain the corresponding feature map vector  $s_t$  and amplification precision enhancement map  $AG_t$ . If we want to get the detection result of the whole video, we only need to run each frame of image sequentially through the network.

When applied to video sequences, the LSTM state can be understood as a feature representing timing. LSTM can then refine its input using timing features at each timing step and meanwhile it can extract additional time information from the input and updating its status. This refinement pattern can be applied by placing LSTM convolution layers immediately on any intermediate feature map. The feature map is used as the input to LSTM and the output of LSTM will replace the previous feature map in all subsequent calculations. The single frame image object detector can be defined by a function  $G(I_t) = D_t$ , which will be used to construct a composite network with  $m$  LSTM layers. These LSTM convolution layers can be considered as dividing the layer of function  $G$  into  $m + 1$  suitable sub-networks  $\{g_0, g_1, \dots, g_m\}$ , then such as Equation (24)

$$G(I_t) = (g_m \circ \dots \circ g_1 \circ g_0)(I_t) \tag{24}$$

The “ $\circ$ ” represents the Hadamard product. We also define any layer of LSTM convolutional layer into a function.

$$L_k(M, s_{t-1}^k, AG_{t-1}) = (M_+, s_t^k, AG_t) \tag{25}$$

In Equation (25),  $M$  and  $M_+$  are feature maps of the same dimension. Then we calculate the formula according to the timing as Equation (26):

$$\begin{aligned} (M_+^0, s_t^0, AG_t^0) &= L_0(g_0(I_t), s_{t-1}^0, AG_{t-1}^0) \\ (M_+^1, s_t^1, AG_t^1) &= L_1(g_1(M_+^0), s_{t-1}^1, AG_{t-1}^1) \\ &\vdots \\ (M_+^{m-1}, s_t^{m-1}, AG_t^{m-1}) &= L_{m-1}(g_{m-1}(M_+^{m-2}), s_{t-1}^{m-1}, AG_{t-1}^{m-1}) \\ D_t &= g_m(M_+^{m-1}) \\ AG_t &= g_m(AG_{t-1}^{m-1}) \end{aligned} \tag{26}$$

Figure 10 depicts the input and output of the entire model when video is processed.

Because multiple gates need to be computed in a single forward channel, LSTMs have high requirements for computing resources, which greatly affects the overall efficiency of the network. To solve this problem, a series of changes have been introduced to make LSTMs compatible with the purpose of real-time moving object detection.

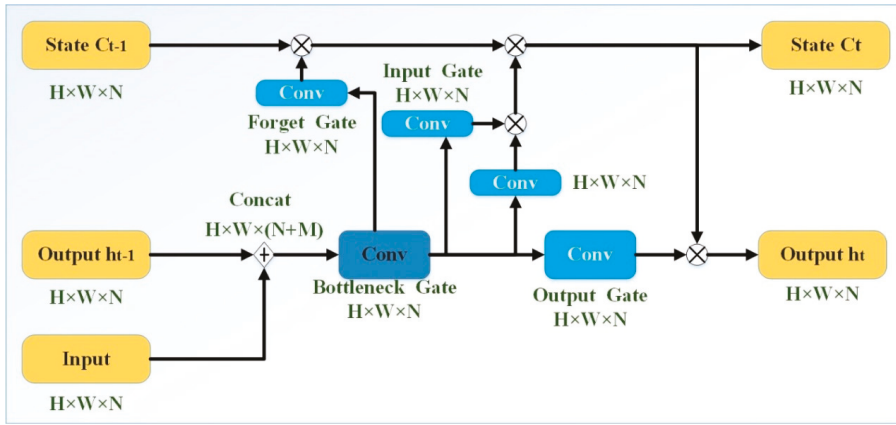


Figure 10. Schematic diagram of the model processing video input and output.

First, we need to consider adjusting the dimensions of LSTM. By extending the channel width multiplier  $\alpha_\delta$  defined in Reference [22], we can have better control over the network structure. The original width multiplier is a super parameter used to scale the channel size of each layer rather than uniformly applying this multiplier to all layers. Three new parameters  $\alpha_{base}$ ,  $\alpha_{ssd}$  and  $\alpha_{lstm}$  are introduced to control the channel sizes of different parts of the network. Any given layer in a basic mobile network with  $N$  output channels is modified to have  $N_{abase}$  basic output channels, while  $\alpha_{ssd}$  is applied to all SSD feature maps and  $\alpha_{lstm}$  is applied to LSTM layers. Here we set  $\alpha_{base} = \alpha$ ,  $\alpha_{ssd} = 0.5\alpha$ ,  $\alpha_{lstm} = 0.25\alpha$  and then the output of each LSTM is a quarter of the input size, which greatly reduces the calculation required.

At the same time, the efficiency of traditional LSTM is greatly improved by adopting a new Bottleneck-LSTM [22], such as Equation (27).

$$b_t = \phi\left({}^{M+N}W_b^N \times [x_t, h_{t-1}]\right) \quad (27)$$

Here,  $x_t$  and  $h_{t-1}$  are the input feature maps;  $\phi(x) = \text{ReLU}(x)$  and ReLU indicates the Rectified Linear Unit activation.  ${}^jW^k \times X$  represents a deep separable convolution with weight  $W$  which come into being after the input channels of  $X$  and  $j$ , as well as the output channels of  $k$  are all input. The benefits of this modification are twofold: first, the use of bottleneck feature mapping reduces the computation within the gate and is thereby superior to standard LSTMs in all practical scenarios; second, the Bottleneck-LSTM is deeper than the standard LSTM and the deeper model is better than the wider and shallower one.

Strong interference phenomena such as occlusion, illumination and shadow in complex traffic scenes may cause loss of object appearance information, which may cause object omissions in the detection process. The well-trained convolutional neural network can cope with a certain degree of interference but it cannot cope with the strong interference of large-area occlusion and the object image information is seriously missing. In this paper, a spatiotemporal context strategy is proposed to obtain useful a priori information from previous detection results to reasonably predict a small number of candidate regions and increase the probability of the object being detected.

This paper chooses Kalman filter [24] as a tool to transfer object information between the previous frame and the current frame and combines the object detection task to design the Kalman filter model.  $D_k = \{X_k^0, X_k^1, \dots, X_k^m\}$  indicates the detection result of the image frame  $I_k$  using the detector without filtering;  $X_k^t = [x_k^t, y_k^t, a_k^t, b_k^t, c_k^t, d_k^t]$  indicates the detection result of the image frame using the detector without filtering;  $x, y, a, b$  and  $d$  are the coordinates and width of the upper left corner of the circumscribed rectangle of an object  $t$  of the  $k$ th frame respectively,  $c$  is the object confidence and  $d$  is

the category to which the object belongs. The predicted value  $D_{k+1}^{\wedge}'$  of the detection result  $D_{k+1}$  of the  $(k + 1)$ th frame of the video can be obtained through LSTM. However, there are errors caused by noise and other factors in the prediction process and so if the prediction result is not corrected, the error will be infinitely amplified during the video detection process due to the iterative process. In order to avoid that, the prediction value  $D_{k+1}^{\wedge}'$  of LSTM is corrected by taking the initial detection result  $Z_{k+1}$  of the video frame  $k + 1$  as the measurement value, that is to say, the estimation value  $D_{k+1}^{\wedge}$  of the detection result  $D_{k+1}$  of the video frame  $k + 1$  is obtained by means of "prediction + measurement feedback." The estimated value filtering equation of the system is Equation (28):

$$X_{k+1}^{\wedge} = A_k \hat{X}_k^t + K_{k+1} \left( Z_{k+1}^t - H_{k+1} A_k \hat{X}_k^t \right) \tag{28}$$

The measurement equation of the system is such as Equation (29):

$$Z_{k+1}^t = H X_{k+1}^t + v_{k+1} \tag{29}$$

The prediction error covariance matrix equation is such as Equation (30):

$$K_{k+1} = P_{k+1/k} H^T \left( H P_{k+1/k} H^T + V_{k+1} \right)^{-1} \tag{30}$$

The Kalman gain equation is such as Equation (31):

$$P_{k+1/k} = A P_k A^T + W_k \tag{31}$$

The modified error covariance matrix equation is such as Equation (32):

$$P_{k+1} = (I - K_{k+1} H) P_{k+1/k} \tag{32}$$

$A$  is a state transition matrix,  $H_1$  is an observation matrix and  $w_k$  is a state noise,  $v_k$  is an observation noise, both of the two kinds of noise are Gaussian white noise. Both state noise  $w_k$  and observed noise  $v_k$  are Gaussian white noise.

The initial values of  $P_{k+1/k}$  and  $X_k$  are  $P_{k=1} = W$  and  $X_1^t = \hat{X}_1^t$  respectively.  $\hat{X}_1^t$  is the state vector of the detection result of the first frame in which the object  $t$  appears, which is passed to the second frame as the estimated value of the first frame for filtering, with the five change values initialized to 0. Starting from the second frame when the object  $t$  appears, the predicted value  $\hat{X}_1^t$  and the estimated value  $\hat{X}_k^t$  of the current frame are taken as the two candidate regions of the frame image and pooling features are extracted along with the candidate regions extracted by SSD. When the frame detection is finished, the result is sent to the next frame for filtering as the filter value of the frame. When there are multiple objects, they are filtered separately and when the number of objects increases, the corresponding number of filters is added. In addition, this paper sets to cancel the filter [10] when the candidate region that corresponds to the ten continuous frames of a certain object is not used as the detection result.

The improved overall detection algorithm framework process is shown in Figure 11, which consists of three major network structures: Dynamic Region Zoom-in Network (yellow border mark), LSTM & Dynamic Kalman filter (green border mark), Adaptive Gabor SSD Detector (red border mark).

- (1) Input a single frame image of the video to be detected, down-sample the image to obtain a low resolution version and reduce the amount of calculation;
- (2) The predicted AG map transmitted by DRZN combined with the LSTM network is used to sequentially select and amplify the object region that needs to be detected at high resolution and the Adaptive Gabor SSD Detector is combined with the predicted Feature map transmitted by the LSTM network for object detection and recognition. Thus we obtain the result  $R_1$  and the resolution of detected areas is set to be 0 in the original image with low resolution;



- (3) Input the remaining low-resolution image into the Adaptive Gabor SSD Detector and combine the predicted feature maps transmitted by the LSTM network to perform object detection and identification and thereby we obtain the result  $R_2$ ;
- (4) Merging the  $R_1$  and  $R_2$  detection results to obtain the initial detection result  $R_3$ ;
- (5) Obtaining the prediction detection result  $R_4$  of the current frame through the LSTM network and combining the initial detection result  $R_3$  and the prediction detection result  $R_4$  by Dynamic Klamn filter to obtain the final detection recognition result  $R_5$ ;
- (6) Input the AG map generated in the current frame detection process, the Feature maps of each layer and the detection result  $R_5$  into the LSTM network to guide the detection result of the next frame.

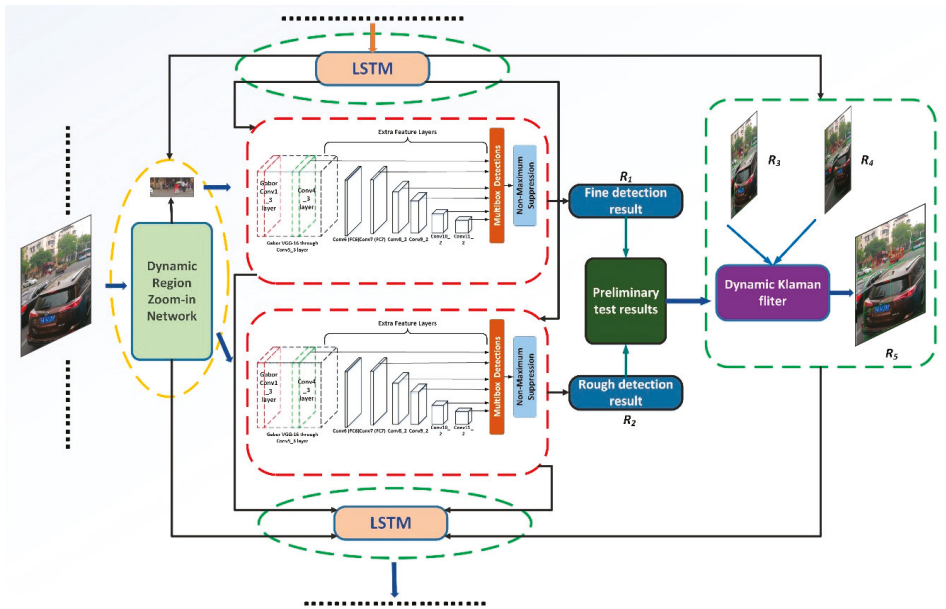


Figure 11. Overall framework of the improved detection algorithm.

## 6. Experimental Analysis

### 6.1. Experimental Basic Conditions and Data Sets

This article uses the DELL Precision R7910 (AWR7910) graphics workstation with Intel Xeon E5-2603 v2 (1.8 GHz/10 M) and NVIDIA Quadro K620 GPU accelerated computing. The SSD is based on the deep learning framework Caffe. Caffe supports parallel computing between CPU and GPU, enabling computationally intensive deep learning to be completed in the short term.

We conducted experiments on the traffic scene dataset [25] (Web dataset) and KITTI dataset collected by YFCC100M. The KITTI dataset, co-founded by the Karlsruhe Institute of Technology in Germany and the Toyota Institute of Technology in the United States, is the largest data collection for computer vision algorithms in the world’s largest autopilot scenario. It is used to evaluate the performance of computer vision technology such as vehicle (motor vehicle, non-motor vehicle, pedestrian, etc.) detection, object tracking and road segmentation in the vehicle environment. KITTI contains real-world image data from scenes such as urban, rural and highways, with up to 15 vehicles and 30 pedestrians per image, with varying degrees of occlusion. In the test set,

100 low-resolution small objects (images with a small object size smaller than  $10 \times 10$ ) were selected to form a low-resolution small object test set of the KITTI data set.

The YFCC100M dataset contains nearly 100 million images along with abstracts, titles and tags. To better demonstrate our approach, we collected 1000 higher resolution test images from the YFCC100M dataset. Images are collected by searching for the keywords “pedestrians,” “roads” and “vehicles.” For this dataset, we annotate all objects with at least 16 pixel width and less than 50% occlusion. The image is rescaled to 2000 pixels on the longer side to fit our GPU memory.

Usually the object detection data set has only one rectangular edge frame for representing the position of the object. In order to detect the object of large-area partial occlusion and to learn from the idea of deformable component model, this paper proposes a local labeling strategy which is to mark some parts of the object with a rectangular frame. Since the local occlusion of the object is generally short during the object motion, the local annotation should not be used too much. Otherwise, the normal object without occlusion will have a higher local detection score and the overall object is not greatly suppressed due to the lower detection score. Exclusion situation. Half of the images from each category in the image are selected as test set 1 and the remaining half of the images are used as training proof sets (where the ratio of the training set to the verification set is 4:1). The proportion of the local annotation of the training set is about 5% and the test set does not use local annotation. In the test set, 100 low-resolution small objects (images with a small object size less than  $10 \times 10$ ) were selected to form a low-resolution small object test set of the WD data set. In the experiment we normalized all image sizes to  $320 \times 320$ .

### 6.2. Experimental Parameter Settings

This paper selects SSD512 [26] in the SSD series to make improvement. SSD512 provides deep convolutional neural network models of large, medium and small scales. We select the medium-sized VGG\_CNN\_M\_1024 model as the basic model and changes the parameters related to the number of object categories. (The original model needs to identify 20 categories of objects and this article has only 3 categories).

The selection of hyperparameters in convolutional neural networks is a key factor affecting the recognition rate. In order to select appropriate hyperparameters, most researchers rely on empirical tests on all data sets to determine the value of hyperparameters based on the recognition results. The method is time-consuming and laborious in the case of a complex data model with a large amount of data and the efficiency is extremely low. Therefore, in order to optimize the tuning process and quickly select the optimal value of adaptive pooled error correction, a small sample data set (200 images) was produced, which greatly saved time and improved the efficiency of parameter adjusting the value selecting. The parameter selection process is shown in Figure 12.

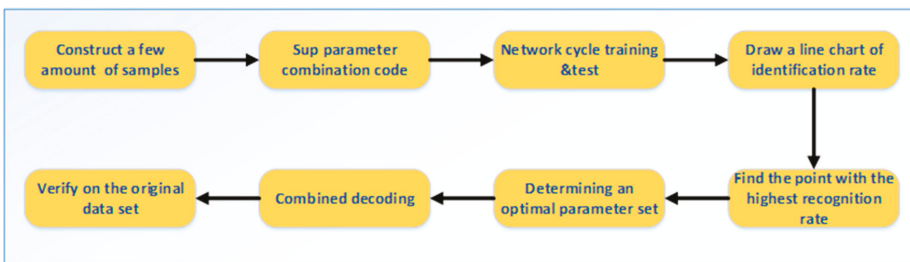


Figure 12. Small sample tuning parameters flow chart.

In the extraction of small samples, both the number of categories of the population and the proportion of each category in the population should be considered and the stratified sampling in the probability sampling method can well take care of these two points. Therefore, according to the

extraction rule, the small sample data set can represent the original data set to a certain extent and the optimal hyperparameter obtained by the small sample data set training can adapt to the original data set to a certain extent. With small sample tuning, the threshold is set to 0.1 (the default setting is 0.7) when the adaptive threshold is not used; the number of candidate regions left by non-maximum suppression in all experiments is set to 100 (the default setting is 300). Other settings remain the same as default and all subsequent experiments are based on the above settings. For LSTM, we expand the LSTM to 10 steps and train in a 10-frame sequence with a channel width multiplier and a model learning rate of 0.003.

### 6.3. Evaluation Indicators

Object detection needs to achieve both object localization and object recognition. By comparing the Intersection over Union (IOU) and the size of the threshold, the accuracy of the object positioning is determined. The correctness of object recognition is determined by comparison of confidence score and threshold value. The above two steps comprehensively determine whether the object detection is correct and finally transform the detection problem of multi-category objects into the binary problem: "For a certain kind of object, the detection is correct or wrong," so that the confusion matrix can be constructed and the accuracy of the model can be evaluated by using a series of indicators of object classification [22].

In the discrimination of multi-object classifier, the number of classes of objects is set as  $n$ . The discrimination of single object still follows four possibilities that each hypothesis has two results, namely, suppose  $D_i^j (j = 1, 2, \dots, n)$  represents an object  $j$  and select hypothesis  $H_i^j$  is true. In any experimental problem of binary hypothesis, four possibilities should be considered when making judgment:

(a)  $H_0^j$  is assumed to be true and evaluated to be  $D_0^j$ ; (b)  $H_0^j$  is assumed to be true and discriminated as  $D_1^j$ ; (c)  $H_1^j$  is assumed to be true and discriminated as  $D_0^j$ ; (d)  $H_1^j$  is assumed to be true and discriminated as  $D_1^j$ .

(a) and (d) select the object  $j$  correctly; (b) is named the first type of error, also called the false alarms (alarmed when there is no such object); (c) is called a type ii error and is called misreporting (where there is an object and misjudgment is no object). In addition, in the multi-object recognition, object  $D_i^j$  is identified as the error discrimination of object  $D_i^k (k = 1, 2, \dots, n, k \neq j)$ .

If the probability density functions of the object  $Z^j$  in the discrimination domain  $Z_0^j$  and  $Z_1^j$  are  $f(z|H_0)$  and  $f(z^j|H_1^j)$  respectively, then there are:

False alarm rate such as Equation (33):

$$P_f = \sum_{j=1}^n P(D_1^j | H_0^j) = \sum_{j=1}^n \int_{Z_1^j} f(z^j | H_0^j) dz \tag{33}$$

Leakage alarm rate such as Equation (34):

$$P_m = \sum_{j=1}^n P(D_0^j | H_1^j) = \sum_{j=1}^n \int_{Z_0^j} f(z^j | H_1^j) dz \tag{34}$$

Detection rate such as Equation (35):

$$P_d = \sum_{j=1}^n P(D_1^j | H_1^j) = \sum_{j=1}^n \int_{Z_1^j} f(z^j | H_1^j) dz \tag{35}$$

Check error rate such as Equation (36):

$$P_e = \sum_{j=1}^n \sum_{k=1, j \neq k}^n P(D_1^j | H_1^j) = \sum_{j=1}^n \sum_{k=1, j \neq k}^n \int_{Z_1^k} f(z^j | H_1^j) dz \quad (36)$$

In multi-object classification, we are concerned with the recognition effect of the existing objects and the recognition rate generally refers to the detection rate. From the definition, we can clearly know that the sum of false alarm rate, detection rate, missed alarm rate and error detection rate is 1. In the actual calculation, the recognition rate is calculated first and then the false alarm rate and false alarm rate are calculated. For multi-object recognition, the false alarm rate accumulated in a certain period of time should be calculated. For the data set, we use the averaging method to calculate the overall false alarm rate, missing alarm rate, detection rate and error detection rate.

Deep learning adjusts the weight of the neural network through the back propagation of the error to achieve the purpose of modeling. The number of back-propagation iterations is gradually increased from tens of thousands of times to hundreds of thousands of times, until the training error tends to converge. Finally, the model is evaluated by the average accuracy of the computational model (average precision, AP) and the average accuracy of all categories (mean AP, mAP). The AP measures the accuracy of the detection algorithm from both the recall rate and the accuracy rate. The AP is the most intuitive standard for evaluating the accuracy of a depth detection model and can be used to analyze the detection of a single category, mAP is the average of APs of each category and the higher the mAP, the higher the overall performance of the model in all categories [19].

#### 6.4. Experimental Design

First, each strategy is combined with SSD512 separately and corresponding comparison experiments are carried out to show the function of each strategy. Then we combine all the strategies with SSD512 to make an overall assessment of the final improved algorithm.

First, we train the original SSD 512 with the training set and record this model as M0. Then a new feature extraction network Gabor-VGGnet strategy is added to the M0 to generate the model M1. An adaptive threshold strategy is added to M0 so as to generate model M2. After that, a dynamic local area enlargement strategy is employed on the basis of M0 to generate a model M3. Based on M0, a moving video object detection improvement strategy based on time-aware feature mapping is added to generate model M4. Finally, M0 is combined with all strategies to generate model M5. Finally, all of the M0, M1, M2, M4 and M5 are tested and compared by using the two database test sets. Moreover, to highlight the low-resolution small object detection, M0 and M3 are tested and compared by using the small object test set.

In addition, this paper selects Faster R-CNN, SSD series and YOLO series detection framework as the deep learning comparison algorithm and compares the detection effect on Web Dataset and KITTI dataset with M5. The Faster R-CNN, SSD Series and YOLO Series detection frameworks use the default parameter settings in the official code published by the author and perform training in the same training set of M5. What's more, we make test with the common test set in the Web Dataset and KITTI datasets.

#### 6.5. Validation of Each Improvement Strategy

The experimental results are shown in Table 1 and the detection results of the common test sets of the models M0, M1, M2, M4 and M5 on the KITTI and WD data sets are compared as follows:

**Table 1.** Comparison of each model identification and detection effect.

Model	Dataset	AP (%)			mAP (%)	$P_f$ (%)	$P_m$ (%)	$P_d$ (%)	$P_e$ (%)
		Person	Car	Cyclist					
M0	KITTI	73.36	71.53	65.32	70.07	20.21	19.34	41.32	19.13
	WD	71.59	69.63	62.75	67.99	19.25	21.38	38.83	20.54
M1	KITTI	87.53	82.16	78.28	82.66	16.48	17.91	57.38	8.23
	WD	85.64	80.59	74.34	80.19	18.95	19.28	51.42	10.35
M2	KITTI	77.18	72.35	68.69	72.74	12.31	13.29	57.84	16.56
	WD	73.52	70.45	64.83	69.61	15.17	14.49	52.45	17.89
M4	KITTI	88.42	81.73	74.38	81.51	9.53	11.69	64.25	14.53
	WD	74.92	72.34	65.63	70.96	16.24	15.19	51.16	17.41
M5	KITTI	92.42	92.23	90.85	91.83	5.19	7.13	81.47	6.21
	WD	88.46	87.38	83.24	86.36	8.26	11.27	71.05	9.42

In the KITTI data set, the AP of various object detections increases by 19~25%, the mAP increases by about 21.76%, the false alarm rate decreases by 15.02% and the detection rate increases by 40.15%, just as what we can see from the M0 and M5 test results through comparison. The missed alarm rate decreases by 12.21% and the false detection rate decreases by 12.92%. In the WD dataset, the AP of various object detections increases by 21~23%, the mAP increases by 18.37% and the false alarm rate decreases by 11.99%. The rate increases by 32.22%, the missed alarm rate decreases by 8.07% and the false detection rate decreases by 11.12%. The improvement of various indicators is obvious, indicating that the overall strategy of this paper is effective in making up for the defects of SSD512.

We compare the M0 and M1 test results in the table so as to find that, in the KITTI data set, the AP of all kinds of object detection increases by 11~14%, the mAP increases by 12.59%, the false alarm rate decreases by 3.73%, the detection rate increases by 16.06%, the missing alarm rate decreases by 1.43% and the false detection rate decreases by 10.9%. In the data set of WD, AP of all kinds of object detection increases by 10~13%, mAP increases by about 12.2%, false alarm rate decreases by 0.3%, detection rate increases by 12.59%, missing alarm rate decreases by 2.1% and false detection rate decreases by 10.19%. Based on M0, M1 model is obtained by adding a new feature extraction network Gabor-VGGnet. By comparing the test results of the two databases with M0, we can find that, compared with M0, M1's object recognition accuracy has been improved greatly and its multi-object error-detection rate reduces significantly, suggesting that a new feature extraction network Gabor-VGGnet, in comparison with the original one, can have more differentiation in terms of the object feature after training.

We compare the M0 and M2 test results in the table so as to find that, in the KITTI data set, the AP of various object tests increases by 1~4%, the mAP increases by about 2.67%, the false alarm rate decreases by 7.90%, the detection rate increases by 16.52%, the missing alarm rate decreases by 6.05% and the error detection rate decreases by 2.57%. In the WD data set, the AP of various object detection increases by 1~3%, the mAP increases by about 1.62%, the false alarm rate decreases by 4.08%, the detection rate increases by 13.62%, the missing alarm rate decreases by 6.89% and the false detection rate decreases by 2.65%. M2 model is based on M0 after the adaptive threshold strategy is trained. Through the comparison of the two databases with M0, we can find that the multiple objective detection rate has been improved, the multiple object detection false alarm rate and missed-alarm rate decreases significantly, showing that the adaptive threshold policy plays a role to differ the real objective with low confidence level from the false objective with high confidence level and thereby it can effectively reduce the missed-alarm rate false alarm rate of SSD512 when multi-object detection is involved.

We compare the M0 and M4 test results in the table so as to find that, in the KITTI data set, the AP of all kinds of object detection increased by 9~15%, the mAP increased by about 11.44%, the false alarm rate decreased by 10.68%, the detection rate increased by 22.93%, the missing alarm rate decreased

by 7.65% and the false detection rate decreased by 4.6%. In the WD data set, AP of all kinds of object detection increased by 1~3%, mAP increased by about 2.97%, false alarm rate decreased by 3.01%, detection rate increased by 12.33%, missing alarm rate decreased by 6.19% and error detection rate decreased by 3.13%. M4 model is based on M0 to join the mobile video object detection based on time perception feature mapping improvement strategy training, through the test results on two databases and M0 comparison we can find that the M4 compared with M0, multiple objective to improve the detection rate of larger, more object detection false alarm rate and missing alarm rate decreased significantly, the recognition of the object average recognition accuracy and precision also won a larger increase. Moreover, since WD dataset is a static image dataset, the spatial-temporal context policy cannot be effective and the improvement effect is not as significant as that in the video dataset KITTI. It is shown that the improved strategy of moving video object detection based on time perception feature map can effectively reduce the leakage and false alarm rate of SSD512 for multi-object detection in video and greatly improve the accuracy of object recognition.

In order to further verify that the M4 model has learned the temporal continuity of the video and is robust in terms of occlusion and other interference, we create artificial occlusion on the single frame image in the KITTI video dataset for testing. For the true detection frame of each object in the image, we design artificial occlusion according to the object occlusion rate  $p_z \in (0, 1]$ . For the object real detection frame of size  $H \times W$ , a region of size  $p_z \cdot H \times p_z \cdot W$  is randomly selected in the detection frame and all pixel values in the region are taken as 0, thus forming artificial occlusion. The normal test set in the KITTI video data set is randomly selected for every 50 frames so as to construct the artificial occlusion and then the anti-occlusion robustness test set is constructed. M0 and M4 are tested on this test set and the object occlusion rate is  $P_z = 0.25, P_z = 0.5, P_z = 0.75, P_z = 0.1$ . The test results are shown in Table 2:

Table 2. M4 anti-occlusion interference verification.

Model	Evaluation Metric	$P_z = 0.25$	$P_z = 0.5$	$P_z = 0.75$	$P_z = 0.1$
M0	mAP (%)	53.36	41.24	22.15	12.89
	$P_d$ (%)	33.58	21.56	12.33	4.25
M4	mAP (%)	74.28	66.82	59.79	51.58
	$P_d$ (%)	60.35	55.62	51.16	42.39

Based on the table above, we compare the different mAP and detection rate  $P_d$  of M0 and M4 when different object occlusion rates are involved and thus find out that our method is superior to the single-frame SSD method when it comes to the occlusion of noise data, indicating that our network has learned the video time continuity and that it can use time clues to achieve robustness in face of occlusion noise.

Table 3 compares the detection effects of the models M0 and M3 on the KITTI and WD data sets on the low-resolution small object test set.

Table 3. Low-resolution small object detection effect verification.

Model	Dataset	AP (%)			mAP (%)	$P_f$ (%)	$P_m$ (%)	$P_d$ (%)	$P_e$ (%)
		Person	Car	Cyclist					
M0	KITTI	13.63	19.38	9.73	14.25	33.12	29.43	10.14	27.31
	WD	8.59	16.33	8.53	11.15	34.15	30.48	6.45	28.92
M3	KITTI	77.45	80.19	58.68	72.11	10.82	10.17	60.48	18.53
	WD	65.62	70.49	52.37	62.83	11.91	14.85	52.03	21.21

We compare the M0 and M3 test results in the table so as to find that, in the KITTI data set, AP for various object detection increases by 49–64%, MAP increases by about 57.86%, false alarm

rate decreases by 22.3%, detection rate increases by 50.34%, missed alarm rate decreases by 19.26% and false alarm rate decreases by 8.78%. In WD data set, AP of various object detection increases by 44–57%, MAP increases by 51.68%, false alarm rate decreases by 22.24%, detection rate increases by 45.58%, missed alarm rate decreases by 15.63% and false alarm rate decreases by 6.71%. The M3 model is based on M0 and it is obtained after we add dynamic local area amplification strategy. By comparing the test results of low-resolution small object test sets on two databases, we can find that, M3, compared with M0, has greatly improved the recognition accuracy and detection rate of those objects with multiple-objective, low-resolution and small size. Thus its false detection rate, false alarm rate and missed alarm rate have significantly decreased, indicating the effectiveness of dynamic local area amplification strategy for the detection and recognition of multiple-objective, low-resolution and small-size objects. Because it is difficult to identify the category of low-resolution weak objects, the false detection rate of M3 is mostly caused by wrong classifications. However, the high false detection rate of M0 is usually caused when multi-object is involved, thus indicating that while SSD 512 deep convolutional network is extracting features layer by layer and it causes serious information loss for low-resolution weak objects.

Figure 13 verifies the effectiveness of R-NET gain effect evaluation in M3 model. The blue-font number in the first line indicates the confidence that the red box is the object. C represents the detection result of the coarse detector and F represents the detection result of the fine detector. The red font number represents the precision gain of R-NET. Positive and negative values are normalized to [0, 1] and [−1, 0]. By comparison, we can find that r-net gives a lower precision gain score for areas where coarse detection is good enough or better than fine detection (column 1 and column 2) and it gives a higher precision gain score for areas where fine detection is much better than coarse detection (column 3).



Figure 13. R-net amplification precision gain effect.

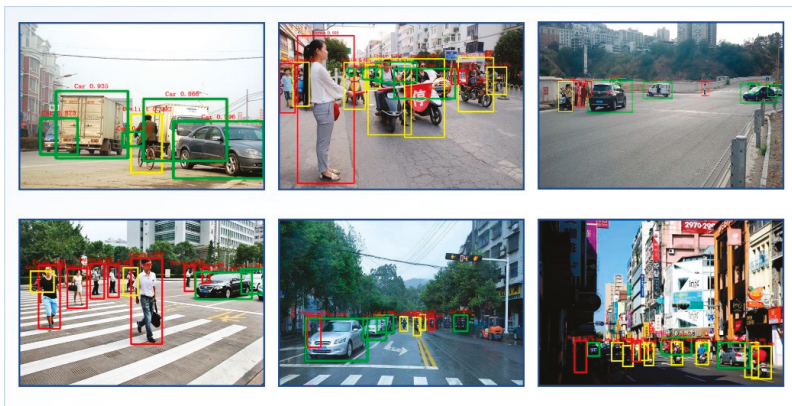
### 6.6. Compare Experiments with Other Detection Algorithms

In addition, this paper selected the detection framework of Faster R-CNN, DSOD300 (Deeply Supervised Object Detector) [27], YOLOv2 544 [28] in the YOLO series detection framework and the improved SSD model DSSD (Deconvolutional Single Shot Detector) [29] as the comparison algorithm of deep learning, so that we can compare their effects with those of M5 on the Web Dataset and KITTI Dataset. The parameter setting used here are the default one that has been published by the author in the official code. We do the training in the same training set as M5, then do the test by using the normal test set of the Web Dataset and KITTI datasets. The detection and recognition effects are shown in Table 4, where  $F_{PS}$  represents the speed and frame rate of the algorithm.

**Table 4.** Comparison of detection and recognition effects of other algorithms.

Method	Dataset	AP (%)			$mAP$ (%)	$P_d$ (%)	$F_{PS}$
		Person	Car	Cyclist			
Faster R-CNN	KITTI	83.26	74.13	75.42	77.61	45.22	13.15
	WD	81.49	71.33	68.65	73.82	36.63	11.64
DSOD300	KITTI	77.43	72.26	68.38	72.69	58.68	58.23
	WD	70.73	69.39	67.04	69.05	52.32	50.35
DSSD513	KITTI	75.46	69.53	68.34	71.11	59.42	46.34
	WD	72.19	68.83	66.45	69.16	49.79	39.38
YOLOv2 544	KITTI	79.43	71.25	67.32	72.66	60.82	56.74
	WD	73.29	69.63	68.85	70.59	54.86	49.28
M5	KITTI	92.42	92.23	90.85	91.83	81.47	31.86
	WD	88.46	87.38	83.24	86.36	71.05	19.83

Comparing with the detection results of M5 and other deep learning comparison algorithms in the above table, we find that, in the KITTI data set, the AP of various object recognition increases by 9~16%, while the  $mAP$  increases by about 14~21% and the detection rate increases by 21~36%. In WD data set, AP of various object recognition increases by 7~11%,  $mAP$  increases by about 13~16% and detection rate increases by 11~35%. Although the detection and recognition rate is not as good as DSOD300, DSSD513, YOLOv2 544 and other detection algorithms,  $F_{PS}$  can also reach 32 frames/s and basically meet the real-time requirements. The detection effects of M5 model are shown in Figure 14.



**Figure 14.** M5 model test results example.



In summary, the M5 model is not only higher than other algorithms in terms of detection accuracy and recognition accuracy but also achieves a detection rate of 32 frames/s. It proves that the algorithm can achieve accuracy and real-time balance, which is fast and good. The performance is obviously superior to other deep learning comparison algorithms and thus has a strong application prospect.

## 7. Conclusions

In order to solve the problem that in complex large traffic scenes, we can hardly balance between the accuracy and real-time performance when we use existing object detection algorithms based on big data and depth learning, this paper improves the object detection framework SSD based on depth learning and then proposes a new multi-object detection framework AP-SSD, which is dedicated to multi-object detection in complex large traffic scenes.

Through testing on the specified data set, we find that, compared with other object detection frameworks based on depth learning, this new multi-object detection framework can enable the average accuracy (AP) of various object recognition to increase by 9–16%, the average accuracy (mAP) to increase by about 14–21%, the multi-object detection rate to increase by 21–36% and the detection and recognition rate to reach 32 frames/s, which basically meets the real-time requirements and is far more robust than other object detection algorithms. Thus it achieves the balance between the accuracy of the algorithm and the running rate and thus providing examples and new ideas for the application of deep learning in specific object detection. In addition, experiments also show that the improved AP-SSD can obtain better detection results when weak objects, multi-object, messy backgrounds, illumination changes, blurs, large-area occlusion and so forth, are involved. Therefore, it makes the video multi-object detection to be of high engineering application value.

**Author Contributions:** X.W.: Conceptualization; Data curation; Project administration; Writing—original draft; Methodology; Validation; Visualization. X.H. (Xia Hua): Conceptualization; Formal analysis; Investigation; Methodology; Project administration; Software; Validation; Visualization; Writing—original draft. F.X.: Formal analysis; Investigation; Methodology; Validation; Visualization; Writing—original draft. Y.L.: Writing—original draft; Software; Investigation. X.H. (Xiaodong Hu): Writing—original draft; Software; Investigation. P.S.: Investigation.

**Funding:** This work was supported in part by the China National Key Research and Development Program (No. 2016YFC0802904), National Natural Science Foundation of China (61671470), Natural Science Foundation of Jiangsu Province (BK20161470), 62nd batch of funded projects of China Postdoctoral Science Foundation (No. 2017M623423).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Ye, T.; Wang, B.; Song, P.; Li, J. Automatic Railway Traffic Object Detection System Using Feature Fusion Refine Neural Network under Shunting Mode. *Sensors* **2018**, *18*, 1916. [[CrossRef](#)] [[PubMed](#)]
2. Lecun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436. [[CrossRef](#)] [[PubMed](#)]
3. Han, J.; Zhang, D.; Cheng, G.; Liu, N.; Xu, D. Advanced Deep-Learning Techniques for Salient and Category-Specific Object Detection: A Survey. *IEEE Signal Process. Mag.* **2018**, *35*, 84–100. [[CrossRef](#)]
4. Xu, X.; Li, Y.; Wu, G.; Luo, J. Multi-modal Deep Feature Learning for RGB-D Object Detection. *Pattern Recognit.* **2017**, *72*, 300–313. [[CrossRef](#)]
5. Ranjan, R.; Sankaranarayanan, S.; Bansal, A.; Bodla, N.; Chen, J.C.; Patel, V.M.; Castillo, C.D.; Chellappa, R. Deep Learning for Understanding Faces: Machines May Be Just as Good, or Better, than Humans. *IEEE Signal Process. Mag.* **2018**, *35*, 66–83. [[CrossRef](#)]
6. Chin, T.W.; Yu, C.L.; Halpern, M.; Genc, H.; Tsao, S.L.; Reddi, V.J. Domain-Specific Approximation for Object Detection. *IEEE Micro* **2018**, *38*, 31–40. [[CrossRef](#)]
7. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2016; pp. 21–37.
8. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [[CrossRef](#)] [[PubMed](#)]

9. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conferences on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
10. Moeskops, P.; Viergever, M.A.; Mendrik, A.M.; de Vries, L.S.; Benders, M.J.; Išgum, I. Automatic segmentation of MR brain images with a convolutional neural network. *IEEE Trans. Med. Imaging* **2017**, *35*, 1252–1261. [[CrossRef](#)] [[PubMed](#)]
11. Jin, K.H.; McCann, M.T.; Froustey, E.; Unser, M. Deep Convolutional Neural Network for Inverse Problems in Imaging. *IEEE Trans. Image Process.* **2017**, *26*, 4509–4522. [[CrossRef](#)] [[PubMed](#)]
12. Zeiler, M.D.; Fergus, R. Visualizing and Understanding Convolutional Networks. In *Computer Vision—ECCV 2014*; Springer: New York, NY, USA, 2014; pp. 818–833.
13. Luan, S.; Chen, C.; Zhang, B.; Han, J.; Liu, J. Gabor Convolutional Networks. *IEEE Trans. Image Process.* **2018**, *27*, 3457–4366. [[CrossRef](#)] [[PubMed](#)]
14. Keil, A.; Stolarova, M.; Moratti, S.; Ray, W.J. Adaptation in human visual cortex as a mechanism for rapid discrimination of aversive stimuli. *Neuroimage* **2007**, *36*, 472–479. [[CrossRef](#)] [[PubMed](#)]
15. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [[CrossRef](#)]
16. Gao, M.; Yu, R.; Li, A.; Morariu, V.I.; Davis, L.S. Dynamic Zoom-in Network for Fast Object Detection in Large Images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
17. Chen, D.; Trivedi, K.S. Optimization for condition-based maintenance with semi-Markov decision process. *Reliab. Eng. Syst. Saf.* **2005**, *90*, 25–29. [[CrossRef](#)]
18. AndrewCucchiara. Applied Logistic Regression. *Technometrics* **2013**, *34*, 358–359.
19. Feng, X.Y.; Mei, W.; Hu, D.S. Aerial Object Detection Based on Improved Faster R-CNN. *Acta Opt. Sin.* **2018**, *38*, 0615004. [[CrossRef](#)]
20. Barhoumi, W.; Bakkay, M.C.; Zargouba, E. Automated photo-consistency test for voxel colouring based on fuzzy adaptive hysteresis thresholding. *IET Image Process.* **2013**, *7*, 713–724. [[CrossRef](#)]
21. Hanson, J.; Yang, Y.; Paliwal, K.; Zhou, Y. Improving protein disorder prediction by deep bidirectional long short-term memory recurrent neural networks. *Bioinformatics* **2017**, *33*, 685–692. [[CrossRef](#)] [[PubMed](#)]
22. Liu, M.; Zhu, M. Mobile Video Object Detection with Temporally-Aware Feature Maps. *arXiv*, 2017; arXiv:1711.06368.
23. Su, B.; Lu, S. Accurate Recognition of Words in Scenes without Character Segmentation using Recurrent Neural Network. *Pattern Recognit.* **2017**, *63*, 397–405. [[CrossRef](#)]
24. Zorzi, M. Robust Kalman Filtering under Model Perturbations. *IEEE Trans. Autom. Control* **2017**, *62*, 2902–2907. [[CrossRef](#)]
25. Thomee, B.; Shamma, D.A.; Friedland, G.; Elizalde, B.; Ni, K.; Poland, D.; Borth, D.; Li, L.J. YFCC100M: The new data in multimedia research. *Commun. ACM* **2016**, *59*, 64–73. [[CrossRef](#)]
26. Wang, Y.; Wang, C.; Zhang, H.; Zhang, C.; Fu, Q. Combing Single Shot Multibox Detector with transfer learning for ship detection using Chinese Gaofen-3 images. In Proceedings of the IEEE Progress in Electromagnetics Research Symposium-Fall, Singapore, 19–22 November 2017; pp. 712–716.
27. Shen, Z.; Liu, Z.; Li, J.; Jiang, Y.G.; Chen, Y.; Xue, X. DSOD: Learning Deeply Supervised Object Detectors from Scratch. *IEEE Comput. Soc.* **2017**, *3*, 1937–1945.
28. Zhang, J.; Huang, M.; Jin, X.; Li, X. A Real-Time Chinese Traffic Sign Detection Algorithm Based on Modified YOLOv2. *Algorithms* **2017**, *10*, 127. [[CrossRef](#)]
29. Fu, C.Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional Single Shot Detector. *arXiv*, 2017; arXiv:1701.06659.



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

Article

# Real-Time Road Lane Detection in Urban Areas Using LiDAR Data

Jiyoung Jung <sup>1</sup> and Sung-Ho Bae <sup>2,\*</sup>

<sup>1</sup> Department of Software Convergence, Kyung Hee University, Yongin 17104, Korea; jiyong.jung@khu.ac.kr

<sup>2</sup> Department of Computer Science and Engineering, Kyung Hee University, Yongin 17104, Korea

\* Correspondence: shbae@khu.ac.kr; Tel.: +82-31-201-2593

Received: 6 September 2018; Accepted: 24 October 2018 ; Published: 26 October 2018

**Abstract:** The generation of digital maps with lane-level resolution is rapidly becoming a necessity, as semi- or fully-autonomous driving vehicles are now commercially available. In this paper, we present a practical real-time working prototype for road lane detection using LiDAR data, which can be further extended to automatic lane-level map generation. Conventional lane detection methods are limited to simple road conditions and are not suitable for complex urban roads with various road signs on the ground. Given a 3D point cloud scanned by a 3D LiDAR sensor, we categorized the points of the drivable region and distinguished the points of the road signs on the ground. Then, we developed an expectation-maximization method to detect parallel lines and update the 3D line parameters in real time, as the probe vehicle equipped with the LiDAR sensor moved forward. The detected and recorded line parameters were integrated to build a lane-level digital map with the help of a GPS/INS sensor. The proposed system was tested to generate accurate lane-level maps of two complex urban routes. The experimental results showed that the proposed system was fast and practical in terms of effectively detecting road lines and generating lane-level maps.

**Keywords:** road lane detection; map generation; driving assistance; autonomous driving

---

## 1. Introduction

Autonomous driving vehicles with various levels of automation from semi-autonomous driving technologies such as adaptive cruise control (ACC) and lane-keeping assist systems (LKAS) to fully-autonomous driving vehicles are now commercially available on the market. While conventional digital maps for car navigation are made for human drivers and have road-level resolution, map providers are now focusing on generating digital maps with a relatively high resolution. The presence of lane-level digital maps reduces the burden of capacity and eventually the cost of each autonomous driving vehicle.

Using conventional digital maps with road-level resolution, an individual autonomous driving vehicle carries excessive burden to fully understand its surroundings to make a decision. For example, if the vehicle has to turn right at the next junction, it needs to figure out the total number of lanes on the road and the lane that the vehicle is currently in in order to move to the rightmost lane safely before reaching the junction. However, if a lane-level map were provided to the vehicle, the path planning process would become considerably simpler and safer so that the individual vehicle would be less obligated to be equipped with very expensive sensors and processors.

Unlike the road-level digital map generation process, a large part of which is automated, the process of generating a lane-level digital map usually requires manual work at many stages. Lane detection on an urban road is more difficult because of multiple lanes, diverse road signs on the ground and complex lane variations at the junctions.

In the case of road-level map generation, it is important to reflect the direction and the curvature of the road accurately, as well as the existence of ramps and overpasses. Lane-level map generation

requires more details; therefore, the individual lanes must be correctly acquired via accurate road line detection followed by appropriate parameterization. If the road is simple and has only a couple of lanes with no road signs on the ground other than the road lines, the generation of a road-level map and a lane-level map will not be very different. However, a complex boulevard in an urban area has multiple lanes and various road signs mixed with road lines. In this case, fast and accurate road line detection becomes an important issue in lane-level map generation.

Many of the conventional road line detection methods using LiDAR data about a decade ago focused on detecting only two lines on each side of the vehicle, in order to reduce the unintended lane departures of an autonomous driving vehicle [1,2]. Kammel and Pitzer [3] proposed a LiDAR-based lane marker detection method, but the purpose of the algorithm was the robust estimation and correction of an offset between the provided map and the real lane on which the autonomous vehicle was traveling. As the road was rather simple, lane markers were assumed to be either the painted road lines or the curbs with some height. Jordan et al. [4] compared LiDAR-based and camera-based lane detection methods and summarized that the LiDAR-based methods detect an increase in the reflectivity of the lane markings when compared to the road surface reflectivity.

More recently, Hata and Wolf [5] detected road markings and curbs by using a 32-channel LiDAR, but the method was tested on a simple two-lane ring track, while road markings other than lines were not distinguished because the purpose of the algorithm was the localization of the vehicle. Yan et al. [6] proposed a mobile mapping system using LiDAR. All road markings including the road lines were precisely extracted to generate a high-definition point-cloud map; however, the markings were not categorized, and the lines were not parameterized for use as a lane-level digital map.

In this paper, we present a practical real-time working prototype for road lane detection in an urban area using LiDAR data. The proposed method is a fully-automated process to detect road lanes on complex urban roads with multiple lanes and diverse road signs on the ground such as arrows with different directions and stop signs. The following are the main contributions of this paper:

- Multiple lanes are detected simultaneously including the lane on which the vehicle is currently.
- Road lines are distinguished from other road markings on complex urban roads.
- Road lines are represented as uniformly-distributed 3D points to be easily used for further applications, such as road curvature calculation and lane-level map generation.
- Lane-level digital map generation by accumulating the detected road lanes is presented as an application of the proposed method.

The rest of this paper is organized as follows: Related works are introduced in Section 2. The proposed method is detailed in Section 3. The experimental results are shown in Section 4, and Section 5 discusses the limitations and the future work of the proposed method and concludes the paper.

## 2. Related Work

Several methods for road map generation have been developed thus far. One of the traditional methods is to use satellite images or aerial images [7–9]. One satellite/aerial image can cover a large area, and the road map can be generated automatically by using adequate image processing. However, up-to-date satellite images are not always available to the general public, and aerial images are expensive to update. Even though the camera resolution is rapidly increasing, it is still too limited to facilitate the generation of a lane-level map from satellite/aerial images. Moreover, as the images are two-dimensional, it is not possible to build a 3D map out of satellite/aerial images.

To build a road map without access to the satellite data or an airplane, in many studies, researchers have installed a GPS sensor on a probe vehicle and recorded its positioning information as the vehicle moves on the road [10–14]. This is an intuitive method of building a road map, and the accuracy of the resulting road map will depend on the accuracy of the GPS sensor. In contrast, as the GPS sensor records only the position of the vehicle, the vehicle has to pass multiple lanes one by one to generate a

lane-level map. The generation of a lane-level map for a route takes a considerable amount of time and effort if the route contains many lanes. Moreover, it is difficult for a human driver to drive the vehicle on the exact center of the lane as the GPS sensor records its positions.

To resolve these shortcomings, map providers nowadays install various perception sensors on the map-making vehicles. The most popular sensor is a camera [15,16]. The use of cameras can provide the information of multiple lanes from a single frontal or an all-around view of the vehicle. The vehicle is no longer forced to be driven on the center of the lane if the GPS sensor and the cameras are calibrated. A 2D or a 3D LiDAR sensor provides the 3D information of its surroundings in real time. Although 3D information can be obtained using multiple cameras, the use of a 3D LiDAR sensor is increasing with a significant decrease in the cost of the sensor [16–20].

After processing the acquired data, various methods can be used to represent the generated lane-level road maps. Several researchers have studied how to represent road maps efficiently while maintaining their high usability in practical applications such as autonomous driving. Several studies have been conducted on the representation of a lane-level road map using polygons [21–24], clothoids [14,25], splines [26–29] and piecewise polynomials [30].

Instead of a particular representation, we simply represent the detected lines by using densely- and uniformly-distributed 3D points along the detected road lines. As lane-level maps used for autonomous driving do not have a global standard to represent the road lines, there are pros and cons to choosing any of the representations mentioned above, and it is inevitable to need to transform a certain representation to another to calculate road characteristics such as the curvature or the tangent of the road. Therefore, because of the computational advantage with little burden for storage, we represent the detected road lines using densely- and uniformly-distributed 3D points along the line, which can easily be processed to any other representations for further application.

Finally, it is worth mentioning a few more studies on road detection using LiDAR in recent years. Clode et al. [31] presented a method for automatic road detection from airborne laser scanner (ALS) data. The test data were collected from Fairfield in Sydney, Australia, with an approximate point density of one point per  $1.3\text{ m}^2$  in the area of  $2\text{ km} \times 2\text{ km}$ . The method classified road or non-road using height and intensity information of ALS data. Later, Clode et al. [32] proposed a more mature method for road/non-road classification using the same dataset, including road vectorization with appropriate parameters such as centerline, orientation and width.

Zhang [33] presented road and road edge (curb) detection method using a 2D LiDAR sensor, which was demonstrated in a prototype vehicle in the DARPA Urban Challenge 2007. A forward down-looking 2D LiDAR sensor was equipped on the front-top of the vehicle, and the elevation information of the range data was used to detect the lowest smooth surface as the drivable region and curbs as road edges. Han et al. [34] presented a road boundary and obstacle detection method for the 2010 Autonomous Vehicle Competition in Korea using a downward-looking 2D LiDAR. The method detected drivable region and obstacles on the road boundary such as curbs, bushes, traffic cones and vehicles. Both methods [33,34] were successfully demonstrated through competition participation; however, the road condition in the test fields was limited to two lanes, and the road lanes were not detected. Yuan et al. [35] presented a road detection and corner extraction method using a 3D LiDAR for robot navigation on the pavement. The test fields were narrow sidewalks surrounded by bushes or buildings, which were distinguished by corner extraction from range data.

Fernandes et al. [36] proposed a road detection method using 3D LiDAR data. The authors projected 3D LiDAR points on a 2D reference image plane and upsampled the points to generate dense height maps. The method detected road surface, which is the drivable region, from the dense height maps surrounded by road boundaries with elevation such as curbs or parked vehicles. Caltagirone et al. [37] proposed a 3D LiDAR-based road detection method using fully-convolutional neural networks (FCN). The FCN was designed for a pixel-wise semantic segmentation task in the point cloud top-view images. The proposed system carried out road segmentation in real time. While both

methods [36,37] were successfully tested using the KITTI road benchmark [38], only the drivable region was detected, and road lanes were not distinguished.

### 3. Proposed System

The proposed method detects the drivable region and road lines from LiDAR data in stages. The LiDAR data are assumed to be scanned from a spinning multi-channel LiDAR sensor installed on top of a vehicle while driving on the urban road.

#### 3.1. Overview

A spinning multi-channel LiDAR sensor spins 5–20 times in a second to densely scan its surroundings with the same number of scanlines as the number of channels. Usually, it scans up to 100 m, which is sufficiently far for a moving vehicle to detect any object in advance and take appropriate action in time. However, a sensor with less than 16 channels may have limited vertical resolution to recognize an object located more than 30 m away.

As we aimed to detect road lines on the ground rather than vertical obstacles by using LiDAR data, we were less obligated to process LiDAR points far away from the sensor. Therefore, it was reasonable to exclude the LiDAR points scanned far away from the sensor with limited resolution. The proposed strategy processed the LiDAR points inside a certain perimeter with an appropriate vertical resolution to be accurately categorized. The detected road lines were later accumulated with the help of a GPS/INS sensor for digital map generation.

We processed the three stages of LiDAR point categorization for road lane detection as illustrated in Figure 1. First, we differentiated the LiDAR points of the drivable region on the ground from the points scanned inside a certain perimeter from the sensor. We assumed that the vehicle was currently on the road and examined the vertical slope between the LiDAR points of the adjacent channels in the radial direction. If the slope was smaller than a certain threshold, the point was categorized as the drivable region until an obstacle was faced. Then, among the points categorized as the drivable region, we distinguished the points of road marks by their intensity. A LiDAR point carried an intensity value, as well as its 3D location, which depended on the reflexivity of the scanned surface. The paint used for the road marks usually had higher reflexivity than the asphalt or cement on the road. Therefore, the points of the road marks could easily be categorized from the points of the drivable region by using the intensity.



Figure 1. LiDAR point categorization.

The final stage of distinguishing road lines from the points categorized as road marks was a difficult problem. Previous works circumvented the issue by detecting the road lines of the simple roads having at most two lanes with few road signs on the ground. In this case, road line detection became as simple as categorizing the LiDAR points with a high intensity from the points of the drivable region. However, diverse road marks such as arrows, stop signs and the names of different destinations of individual lanes frequently appeared on the urban roads. It was difficult to distinguish them from the road lines because road marks and road lines usually had the same reflexivity to the LiDAR scan.

#### 3.2. LiDAR Point Categorization

A multi-channel LiDAR sensor carries multiple laser scanners, as many as the number of its channels. They are usually installed vertically inside the sensor. Each laser scanner emits and receives

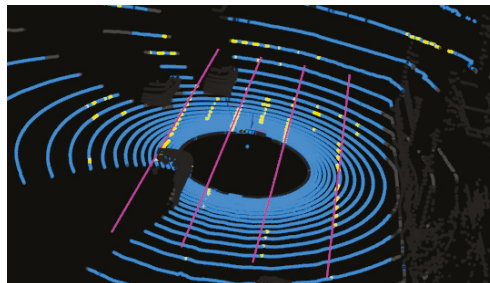
light to measure the distance for each channel. In order to scan the entire surroundings of the sensor, a mirror inside the sensor spins rapidly to scan 360°.

We categorized the LiDAR points inside a certain perimeter from the sensor as the drivable region by calculating the slope between the points in the radial direction from the sensor. When a spinning multi-channel LiDAR horizontally scanned 360° with multiple laser emitters vertically installed on the sensor, the points of multiple channels scanned at the same instance formed a ray from the sensor. For example, Velodyne HDL-32E horizontally scanned 360° in 0.1 s with 32 lasers. Each channel scanned approximately 2100 points per spin; therefore, approximately six points were densely positioned in each degree angle. Furthermore, 32 points scanned at the same time formed a ray from the sensor. If we considered the point nearest to the sensor as Channel Number 0 and the point furthest from the sensor as Channel Number 31, we could calculate the slope between two points of the adjacent channels from Channels 0–31 as follows:

$$s = \frac{\Delta z}{\sqrt{\Delta x^2 + \Delta y^2}} = \frac{z_b - z_a}{\sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}} \quad (1)$$

where  $a$  and  $b$  indicate the adjacent channels and  $x$ ,  $y$  and  $z$  indicate the 3D coordinates of each LiDAR point in the right, forward and up directions from the sensor, respectively. When the slope  $s$  of the ground became greater than a certain threshold (i.e., we empirically set the value to 0.15), we assumed that there was a vertical obstacle on the ground and categorized the points on the same ray beyond this channel as not drivable.

This method effectively detected sudden vertical slope changes on the road such as the curb between the road and the sidewalk or vertical obstacles such as the median strip, beacons or other vehicles. The method successfully categorized speed bumps or small potholes as the drivable regions. An example of drivable region detection is shown in Figure 2. Gray points indicate the given 3D LiDAR points, whereas blue points indicate the points categorized as the drivable region.



**Figure 2.** Example of LiDAR point categorization and the road line detection result. Blue points indicate the drivable region, yellow points the road marks including road lines and the magenta lines the detected road lines.

Once the points of the drivable region were discriminated, we distinguished the points of road marks by using the LiDAR point intensity. The signs and lines on the road were usually painted with conspicuous colors such as white or yellow. The intensity of the points scanned at the road marks was easily distinguishable from the intensity of the points scanned at the asphalt or cement on the ground. The LiDAR point intensity of the road marks was usually considerably higher. In Figure 2, the yellow points indicate the detected road marks on the ground.

### 3.3. Road Line Detection

To detect the road lines out of various road marks having similar LiDAR intensities, we searched for a set of parallel lines separated by the interval of the lane width. The lane width was initially set as 3.4 m, but the value was updated during the detection process.

Given a set of LiDAR points categorized as road marks, we aimed to estimate  $L$  sets of line parameters. We modeled each road line in 3D as:

$$[x, y, z]^T = t\mathbf{u} + \mathbf{v}_0 = t[u_x, u_y, u_z]^T + [x_0, y_0, z_0]^T, \quad (2)$$

where  $\mathbf{u}$  is a vector parallel to the line,  $\mathbf{v}_0$  is a point on the line and  $t$  is a real number.

For initialization, we initialized  $K \geq 2L - 1$  sets of lines with the interval of half of the initial lane width to separate the road marks from the road lines effectively, where  $K$  is the number of candidates to detect  $L$  lines. Assuming that the location of the sensor was the origin and the sensor was facing front, all the line parameters were initialized as follows:

$$u_x = 0, \quad u_y = 1, \quad u_z = 0, \quad y_0 = 0, \quad z_0 = -2.0. \quad (3)$$

This initialization indicates that all the lines are parallel to the  $y$ -axis and lie on the ground two meters below the sensor. Note that the  $x$ ,  $y$ , and  $z$  axes point in the right, forward and up directions, respectively. The  $x_0$  values for the lines were initialized to be separated with intervals of  $0.5w$ , where  $w$  is the initial lane width set as 3.4.

After initialization, we updated the  $x_0$  value of each line with the assumption that the LiDAR points for the line were stably found when the vehicle moved forward, whereas the points for the other road marks were discontinued. We designed an iterative expectation-maximization (EM) algorithm to determine reliable sets of line parameters. For each iteration, we assigned each LiDAR point to its nearest line by calculating the point-line distance. After the assignment, we updated  $x_0$  for each line to be the average of the  $x$ -coordinate values of the assigned points. This process was very simple and fast. We repeated the process 100 times for each spin of the LiDAR.

Finally, we selected  $L$  reliable sets of line parameters out of the  $K \geq 2L - 1$  updated sets after the EM-algorithm. For each line  $k$ , we calculated the number of assigned points for each line  $N_k$  and the standard deviation  $\sigma_k$  of the  $y$ -coordinate values of the assigned points.  $\sigma_k$  showed the level of distribution of the points in the direction of the  $y$ -axis.  $\sigma_k$  of the cluster of a line tended to be larger than that of the cluster of other road marks. From the cluster having the largest  $N_k$ , we selected the line parameters as a reliable set if  $\sigma_k$  was larger than a certain threshold. As the line was expected to be continued from the last spin, we also checked if the current line was continued from one of the detected lines from the last spin. That is, we calculated the minimum difference between  $x_{0k}$  with the  $x_0$  values of the detected lines from the last spin and considered it as the same line when the minimum difference was below a certain threshold. After selecting  $L$  sets of reliable line parameters, we ended the line selection process.

As a result, the detected  $L$  sets of line parameters were updated at every spin of the LiDAR. The lines were stably detected, while the vehicle changed lanes or went through curved roads. Algorithm 1 summarizes the proposed road line detection process.

### 3.4. Lane-Level Digital Map Generation

Nowadays, most of the map-making cars are equipped with a GPS/INS sensor, which provides the current GPS location of the vehicle, as well as the rotation matrix from the sensor coordinate to the world coordinate. The world coordinate generally refers to the north-east-down (NED) coordinate. Provided that the GPS/INS sensor was calibrated with the LiDAR sensor, we transferred the estimated parameter  $x_0$  of the detected road lines to the world coordinates for each spin of the LiDAR.



We used the current position in terms of the latitude and longitude along with the altitude provided by the GPS/INS sensor, as well as the rotation matrix that transformed the sensor coordinate frame to the local NED coordinate in real time. No additional method was used to post-process the positioning data. The latitude and longitude values were converted to the UTM coordinate to obtain the position values in meters. As the estimated parameter  $x_0$  was the  $x$ -coordinate value in the sensor coordinate frame, it was transformed to a position in the world coordinates by multiplying the rotation matrix and then adding the current sensor position in UTM.

---

**Algorithm 1:** Road line detection.
 

---

**Data:** LiDAR point cloud of a single spin categorized as road marks:  $P = \{x_p, y_p, z_p\}$   
**Result:**  $L$  sets of line parameters  $\{u_x, u_y, u_z, x_0, y_0, z_0\}$

```

1 /* Initialization */
2 for k ← 1 to K ≥ 2L − 1 do
3   Initialize parallel line parameters as
4   |    $u_{xk} \leftarrow 0, u_{yk} \leftarrow 1.0, u_{zk} \leftarrow 0,$ 
5   |    $x_{0k} \leftarrow x_{leftmost} + 0.5wk, y_{0k} \leftarrow 0, z_{0k} \leftarrow -2.0$ 
6   |   where  $w$  is the width of the lane
7 end
8 /* EM-like algorithm to estimate line parameters */
9 for i ← 1 to 100 do
10  Expectation: Assign each points to the nearest line by calculating point-line distance
11  Maximization: Update  $x_{0k}$  for each line as  $\sum_{N_k} x_p / N_k$ 
12  |   where  $N_k$  is the number of points assigned to line  $k$  and
13  |    $\sum_{N_k} x_p$  is the sum of the  $x$ -coordinate of the assigned  $N_k$  points.
14 end
15 /* Line selection */
16 for k ← 1 to K do
17  Calculate the final cluster size  $N_k$  for each line
18  Calculate the standard deviation  $\sigma_k$  of the  $y$ -coordinate of the assigned  $N_k$  points as
19  |    $\sigma_k = \sqrt{\frac{\sum_{N_k} (y_p - \bar{y}_p)^2}{N_k - 1}}$ 
20 end
21 Set  $i = 0$ 
22 for k ← in the order of having largest  $N_k$  do
23  if  $\sigma_k > 2.5$  and  $\Delta x_0 < 0.4$  then
24  |   Select  $k$  as a reliable set of line parameters
25  |    $i \leftarrow i + 1$ 
26  end
27  if  $i == L$  then
28  |   break;
29  end
30 end

```

---

We filtered out the duplicate dots, which were estimated while the vehicle stopped or moved very slowly. Then, we simply connected the dots to build the lane-level digital map. For qualitative evaluation, we overlapped the final lane-level digital map with the satellite images.

## 4. Results

### 4.1. Data Acquisition

The data to evaluate the proposed road lane detection method were collected using a vehicle equipped with a 32-channel 3D LiDAR (Velodyne HDR-32E). For the lane-level map generation, we used a GPS/INS sensor (Ekinox-D). Velodyne HDR-32E has 32 laser scanners with a horizontal field-of-view of 360° a vertical field-of-view of 40° and an angular resolution of 1.33°. We set the rotation rate to be 10 Hz and gathered approximately 69,000 points per spin in 0.1 s. Ekinox-D is an inertial navigation system with an integrated dual antenna GNSS receiver that provides highly accurate 6D vehicle poses. It combines an inertial measurement unit (IMU) and runs an enhanced on-board extended Kalman filter (EKF) to fuse real-time inertial data with internal GNSS information. The sensor embeds a GNSS receiver, capable of centimeter-scale accuracy by using an RTK solution [39,40]. The vehicle was driven at a normal speed, from 0–60 km/h, depending on the traffic situation. All experiments were carried out on a PC with a 3.40-GHz i7-6800K CPU.

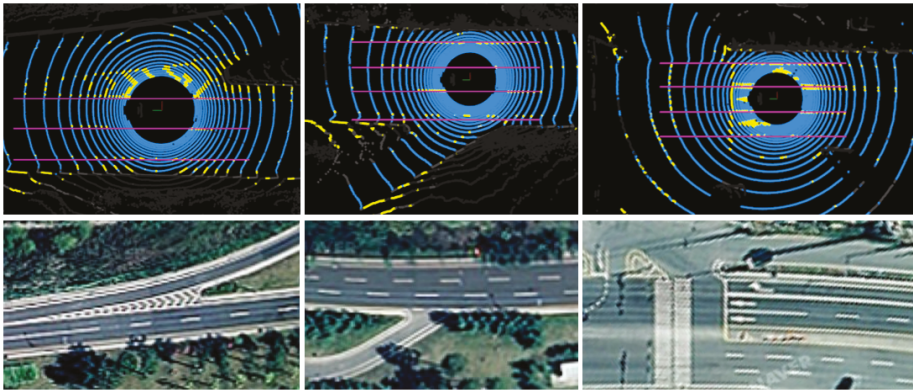
The dataset was collected along two different routes in the cities of Seongnam and Incheon, South Korea, which are the 10th and the third largest cities in the country with a population of nearly one million and three million, respectively. Route 1 is an expressway with sidewalks in the city of Seongnam. It is approximately 7 km in round trip and contains junctions, crosswalks, ramps and a tunnel. The number of lanes changes from one to three on one side of the road. Because of a wide separation area at the center of the road, we only processed one side of the road of the current driving direction first and then processed the other side on the way back. Route 2 travels along the major boulevard in the city of Incheon with a length of approximately 2 km in round trip. The route contains junctions, crosswalks and various road marks on the ground.

### 4.2. Qualitative Results

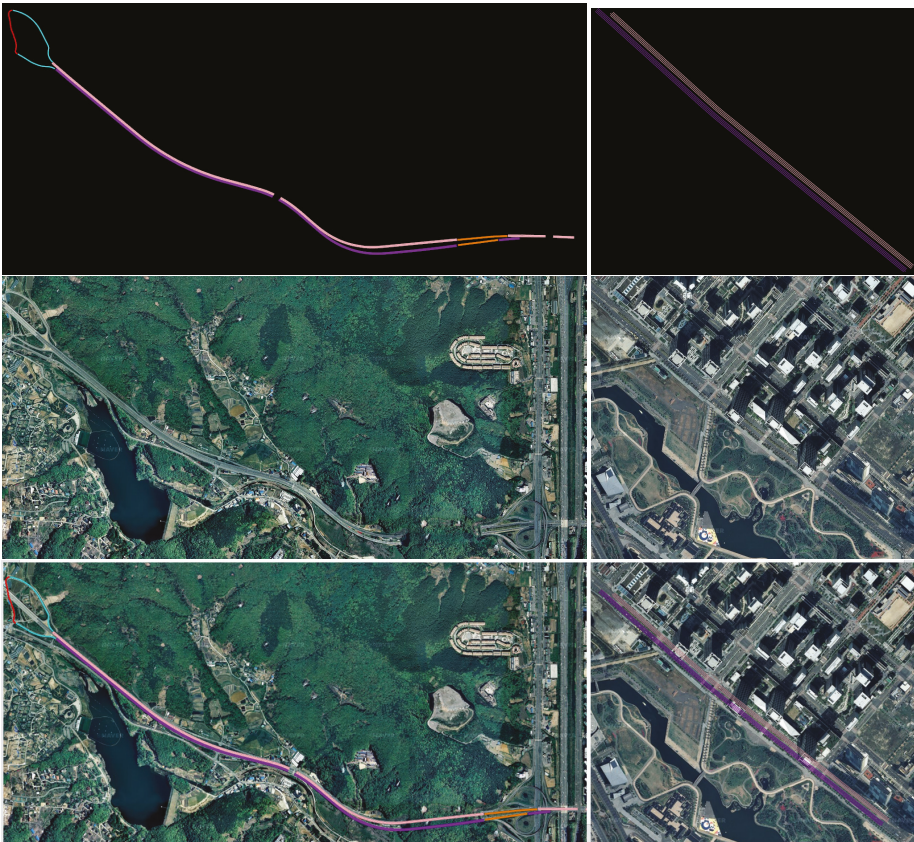
Both Route 1 and Route 2 were modeled using densely- and uniformly-distributed 3D points on the road lines. Figure 3 shows three different examples of drivable region categorization and road line detection with two or three lanes on the road in the presence of ramps or junctions. Gray points indicate the given 3D point cloud scanned by a 3D LiDAR; blue points indicate the drivable region; yellow points indicate the road marks on the ground; and magenta lines are the final results of the road line detection at each spin of the LiDAR sensor.

The detected road line was parameterized, and its parameters were recorded as a 3D point on the line in the NED coordinate with the help of a GPS/INS sensor. Figure 4 shows the result of the 3D lane-level map of Route 1 and Route 2 generated by connecting these 3D points, as well as the corresponding satellite images for a reference. In Figure 4, pink and magenta lines indicate three-lane regions in opposite directions. Orange lines indicate two-lane regions, and cyan lines indicate one-lane regions where the vehicle was driving on the ramp to get off or to get on the main driveway. The red line is an off-road region with no official road line on the ground. We added the red line manually just to show the entire route. The disconnected parts indicate junctions with crosswalks and stop lines. As the road lines were disconnected at the junctions, the resulting map lines were disconnected, as well.

As it was difficult to evaluate the accuracy of the generated lane-level map quantitatively, we overlaid the resulting lane-level map on the satellite image provided by Naver Map [41] for the qualitative evaluation. Figures 5 and 6 show the lane-level maps overlapped with the corresponding satellite images of Route 1 and Route 2 in magnified views, respectively. The left column shows the satellite images, the middle column the resulting lane-level map overlapped with the satellite images and the right column the existing road-level map provided by Naver Map [41]. We observed that the detected road lanes of the generated map were accurately synchronized with the road lanes in the satellite images.



**Figure 3.** Three different examples of drivable region categorization and road line detection with two or three lanes on the road in the presence of ramps on the right or left (**left, middle**) and at the junction (**right**). The figures below show the satellite images of the same scene for better understanding.



**Figure 4.** Result of the lane-level digital map and the corresponding satellite images of Route 1 (**left**) and Route 2 (**right**). Magnified views are provided below.



**Figure 5.** Detected road lanes on Route 1 are accumulated to build a lane-level digital map with the help of a GPS/INS sensor. The left column shows the satellite images, the middle column the resulting lane-level map overlapped with the satellite images and the right column the existing road-level map provided by Naver Map [41].



**Figure 6.** More results of the detected road lanes on Route 2 are accumulated to generate a lane-level digital map with the help of a GPS/INS sensor. The left column shows the satellite images, the middle column the resulting lane-level map overlapped with the satellite images and the right column the existing road-level map provided by Naver Map [41].

A comparison of the overall routes and the corresponding satellite images in Figure 4 revealed that the lanes were accurately detected and well represented by connecting the 3D points obtained from the line parameters. Although Route 2 was short and looked simple, it was a challenging route with heavy traffic and complex road signs on the ground. Moreover, many tall buildings interrupted

the GPS signals and made it more difficult to generate a lane-level map without the use of perception sensors. Note that the resulting lanes were accurately overlapped with the road lines in the satellite images, whereas the existing road-level map roughly represented the wide road in the macroscale, as shown in Figure 6.

**Table 1.** Consistency evaluation with the commercial road-level map [41].

Route 1	Mean (m)	St.Dev. (m)	Route 2	Mean (m)	St.Dev. (m)
Line 1 (West)	4.724	1.684	Line 1 (Northwest)	7.621	0.079
Line 2 (West)	1.332	1.692	Line 2 (Northwest)	4.226	0.083
Line 3 (West)	2.089	1.685	Line 3 (Northwest)	0.824	0.078
Line 4 (West)	4.717	1.693	Line 4 (Northwest)	2.572	0.075
Line 1 (East)	6.264	1.676	Line 1 (Southeast)	4.900	0.079
Line 2 (East)	2.868	1.674	Line 2 (Southeast)	1.491	0.084
Line 3 (East)	0.547	1.690	Line 3 (Southeast)	1.899	0.087
Line 4 (East)	3.933	1.683	Line 4 (Southeast)	5.288	0.084

In order to evaluate the consistency of the detected road lines with the existing road-level map, we extracted the middle line from the road-level map provided by Naver Map [41] and calculated the point-line distance from our final 3D point representation of the detected road lines. As our representation existed for multiple lanes and a road-level map provided only one line, the mean distance varied for each line, but the standard deviation remained small, which showed that our result was consistent with the road-level map. The standard deviation for Route 1 was relatively large because the middle line extracted from the road-level map of Route 1 fluctuated from Line 2 to Line 3. Route 2 was rather straight, so the middle line of the road-level map of Route 2 was relatively consistent near Line 3 (towards the northwest) and between Line 2 and Line 3 (towards the southeast). Therefore, the standard deviation remained very small, less than 10 cm. Note that as the reference (road-level map) did not provide the ground-truth road line position, Table 1 does not show the absolute performance of the proposed method, but provides some clues of the consistency of our road line detection results with the existing road-level map.

## 5. Discussion and Conclusions

### 5.1. Limitations and Future Work

There are several limitations to our system, which need to be addressed before a full solution for the given issue can be developed. The proposed method aimed for lane-level map generation by using the detected road lines rather than immediate lane keeping during autonomous driving as an application. Therefore, we did not consider heavy weather conditions, which might degrade the 3D LiDAR sensor input. When the purpose was mapping, we could simply choose a good weather day to scan the road, whereas this can be a critical limitation when the method is applied for road lane detection for a lane keeping system. When used for lane keeping in bad weather, the proposed method might be simplified to detect only two road lines on each side of the current vehicle, and the input data could be compensated by other perception sensors, such as cameras.

While it is one of the strengths that the proposed method was tested on complex urban roads, this can also indicate that the method was difficult to apply to unpaved roads. Roads with drastic slopes and severe curvatures in mountainous areas could be challenging and might require some different empirical threshold values for the proposed method.

Finally, while vehicles crossing road lines as they changed lanes during driving were never a critical problem for the proposed algorithm, it was difficult to detect the road lines beneath the parked cars on the side, completely obscuring the road lines. As the undetected road line was usually the rightmost line in this case, this issue could be alleviated by knowing the total number of lines to be detected beforehand. If we have prior knowledge that there should be another line to be detected,

we can adjust the algorithm to detect the rightmost line using less input data observed between the parked cars.

Potential future work would depend on a further application of the proposed road lane detection method. For example, if we aim to extend the proposed method for use in a lane-keeping assist system for autonomous driving vehicles, we should focus on improving the detection accuracy of the two lines on each side of the vehicle under various weather and lighting conditions. To extend the proposed method toward fully-automated lane-level map generation, we should test the method under various road conditions, such as unpaved roads, mountainous roads with severe slope and curvature, crowded urban roads and roads with noisy GPS signals.

## 5.2. Conclusions

In this paper, we presented a simple and practical real-time working prototype for road lane detection in an urban area by using 3D LiDAR points. The overall system consisted of two subsystems, including point categorization and road line detection. Given the 3D LiDAR point cloud, we categorized the points of the drivable region and distinguished the points of the road signs on the ground. Then, we presented an expectation-maximization process to detect and update the 3D line parameters in real time. The detected road lines were represented as densely- and uniformly-distributed 3D points on the lines with the help of a GPS/INS sensor and integrated to generate a lane-level digital map. The proposed system was tested to generate the lane-level maps of two complex urban routes in the cities of Seongnam and Incheon, South Korea. The accuracy of the resulting lane-level map was evaluated by comparing with the corresponding satellite images.

**Author Contributions:** Conceptualization, J.J.; Methodology, J.J.; Software, J.J.; Validation, S.-H.B.; Writing—Original Draft, J.J.; Writing—Review & Editing, S.-H.B.

**Funding:** This research was funded by Kyung Hee University (KHU-20170718) and the National Research Foundation of Korea (NRF-2017R1C1B5075945).

**Acknowledgments:** This work was partially implemented when the first author was with Naver Labs.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

2D	Two-dimensional
3D	Three-dimensional
LiDAR	Light detection and ranging
GPS	Global positioning system
INS	Inertial navigation system
GNSS	Global navigation satellite system
RTK	Real-time kinematic
UTM	Universal Transverse Mercator
PC	Personal computer
CPU	Central processing unit

## References

1. Reyher, A.; Joos, A.; Winner, H. A lidar-based approach for near range lane detection. In Proceedings of the IEEE Intelligent Vehicle Symposium, Las Vegas, NV, USA, 6–8 June 2005; pp. 147–152.
2. Lindner, P.; Richter, E.; Wanielik, G.; Takagi, K.; Isogai, A. Multi-Channel Lidar Processing for Lane Detection and Estimation. In Proceedings of the 12th International IEEE Conference on Intelligent Transportation Systems, St. Louis, MO, USA, 3–7 October 2009; pp. 202–207.
3. Kammel, S.; Pitzer, B. Lidar-based lane marker detection and mapping. In Proceedings of the 2008 IEEE Intelligent Vehicles Symposium, Eindhoven, The Netherlands, 4–6 June 2008; pp. 1137–1142.

4. Jordan, B.; Rose, C.; Bevly, D. A Comparative Study of Lidar and Camera-based Lane Departure Warning Systems. In Proceedings of the ION GNSS 2011, Portland, OR, USA, 20–23 September 2011.
5. Hata, A.; Wolf, D. Road marking detection using LIDAR reflective intensity data and its application to vehicle localization. In Proceedings of the International IEEE Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, 8–11 October 2014.
6. Yan, L.; Liu, H.; Tan, J.; Li, Z.; Xie, H.; Chen, C. Scan Line Based Road Marking Extraction from Mobile LiDAR Point Clouds. *Sensors* **2016**, *16*, 903. [[CrossRef](#)] [[PubMed](#)]
7. Amo, M.; Martinez, F.; Torre, M. Road extraction from aerial images using a region competition algorithm. *IEEE Trans. Image Process.* **2006**, *15*, 1192–1201. [[CrossRef](#)] [[PubMed](#)]
8. Hu, J.; Razdan, A.; Femiani, J.; Cui, M.; Wonka, P. Road network extraction and intersection detection from aerial images by tracking road footprints. *IEEE Trans. Geosci. Remote Sens.* **2007**, *45*, 4144–4157. [[CrossRef](#)]
9. Seo, Y.-W.; Urmson, C.; Wettergreen, D. Exploiting publicly available cartographic resources for aerial image analysis. In Proceedings of the International Conference on Advances in Geographic Information Systems, Redondo Beach, CA, USA, 7–9 November 2012; pp. 109–118.
10. Rogers, S. Creating and evaluating highly accurate maps with probe vehicles. In Proceedings of the IEEE Intelligent Transportation Systems, Dearborn, MI, USA, 1–3 October 2000; pp. 125–130.
11. Qi, H.; Moore, J. Direct Kalman filtering approach for GPS/INS integration. *IEEE Trans. Aerosp. Electron. Syst.* **2002**, *38*, 687–693.
12. Schroedl, S.; Wagstaff, K.; Rogers, S.; Langley, P.; Wilson, C. Mining GPS traces for map refinement. *Data Min. Knowl. Discovery* **2004**, *9*, 59–87. [[CrossRef](#)]
13. Caron, F.; Duflos, E.; Pomorski, D.; Vanheeghe, P. GPS/IMU data fusion using multisensor Kalman filtering: introduction of contextual aspects. *Inf. Fusion* **2006**, *7*, 221–230. [[CrossRef](#)]
14. Betaille, D.; Toledo-Moreo, R. Creating enhanced maps for lane-level vehicle navigation. *IEEE Trans. Intell. Transp. Syst.* **2010**, *11*, 786–798. [[CrossRef](#)]
15. Ziegler, J.; Dang, T.; Franke, U.; Lategahn, H.; Bender, P.; Schreiber, M.; Strauss, T.; Appenrod, N.; Keller, G.G.; Kaus, E; et al. Making Bertha Drive—An autonomous journey on a historic route. *IEEE Intell. Transp. Syst. Mag.* **2014**, *6*, 8–20. [[CrossRef](#)]
16. Roh, H.; Jeong, J.; Cho, Y.; Kim, A. Accurate Mobile Urban Mapping via Digital Map-Based SLAM. *Sensors* **2016**, *16*, 1315. [[CrossRef](#)] [[PubMed](#)]
17. Boyko, A.; Funkhouser, T. Extracting road from dense point clouds in large scale urban environment. *ISPRS J. Photogramm. Remote Sens.* **2011**, *66*, S2–S12. [[CrossRef](#)]
18. Mancini, A.; Frontoni, E.; Zingaretti, P. Automatic road object extraction from mobile mapping systems. In Proceedings of the IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications, Suzhou, China, 8–10 July 2012; pp. 281–286.
19. Guan, H.; Li, J.; Yu, Y.; Wang, C.; Chapman, N.; Yang, B. Using mobile laser scanning data for automated extraction of road markings. *ISPRS J. Photogramm. Remote Sens.* **2014**, *87*, 93–107. [[CrossRef](#)]
20. Joshi, A.; James, M.R. Generation of accurate lane-level maps from coarse prior maps and lidar. *IEEE Intell. Transp. Syst. Mag.* **2015**, *7*, 19–29. [[CrossRef](#)]
21. Safra, E.; Kanza, Y.; Sagiv, Y.; Doytsher, Y. Efficient integration of road maps. In Proceedings of the 14th annual ACM International Symposium on Advances in Geographic Information Systems, Arlington, VA, USA, 10–11 November 2006; pp. 59–66.
22. Haklay, M.; Weber, P. OpenStreetMap: User-generated street maps. *IEEE Pervasive Comput.* **2008**, *7*, 12–18. [[CrossRef](#)]
23. Vatavu, A.; Danescu, R.; Nedevschi, S. Environment perception using dynamic polylines and particle based occupancy grids. In Proceedings of the IEEE 7th International Conference on Intelligent Computer Communication and Processing, Cluj-Napoca, Romania, 25–27 August 2011; pp. 239–244.
24. Sivaraman, S.; Trivedi, M.M. Dynamic probabilistic drivability maps for lane change and merge driver assistance. *IEEE Trans. Intell. Transp. Syst.* **2014**, *15*, 2063–2073. [[CrossRef](#)]
25. Gikas, V.; Stratakos, J. A novel geodetic engineering method for accurate and automated road/railway centerline geometry extraction based on the bearing diagram and fractal behavior. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 115–126. [[CrossRef](#)]

26. Schindler, A.; Maier, G.; Pangerl, S. Exploiting arc splines for digital maps. In Proceedings of the 14th International IEEE Conference on Intelligent Transportation Systems, Washington, DC, USA, 5–7 October 2011; pp. 1–6.
27. Schindler, A.; Maier, G.; Janda, F. Generation of high precision digital maps using circular arc splines. In Proceedings of the IEEE Intelligent Vehicles Symposium, Alcalá de Henares, Spain, 3–7 June 2012; pp. 246–251.
28. Brummer, S.; Janda, F.; Maier, G.; Schindler, A. Evaluation of a mapping strategy based on smooth arc splines for different road types. In Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems, The Hague, The Netherlands, 6–9 October 2013; pp. 160–165.
29. Jo, K.; Sunwoo, M. Generation of a precise roadway map for autonomous cars. *IEEE Trans. Intell. Transp. Syst.* **2014**, *15*, 925–937. [CrossRef]
30. Gwon, G.-P.; Hur, W.-S.; Kim, S.-W.; Seo, S.-W. Generation of a Precise and efficient lane-level road map for intelligent vehicle systems. *IEEE Trans. Veh. Technol.* **2017**, *66*, 4517–4533. [CrossRef]
31. Clode, S.; Kootsookos, P.J.; Rottensteiner, F. The automatic extraction of roads from LIDAR data. In Proceedings of the XXth ISPRS Congress Technical Commission III, Istanbul, Turkey, 12–23 July 2004; pp. 231–236.
32. Clode, S.; Rottensteiner, F.; Kootsookos, P.J.; Zelniker, E. Detection and vectorization of roads from lidar data. *Photogramm. Eng. Remote Sens.* **2007**, *73*, 517–535. [CrossRef]
33. Zhang, W. LIDAR-based road and road-edge detection. In Proceedings of the IEEE Intelligent Vehicles Symposium, San Diego, CA, USA, 21–24 June 2010; pp. 845–848.
34. Han, J.; Kim, D.; Lee, M.; Sunwoo, M. Enhanced road boundary and obstacle detection using a downward-looking LIDAR sensor. *IEEE Trans. Veh. Technol.* **2012**, *61*, 971–985. [CrossRef]
35. Yuan, X.; Zhao, C.-X.; Zhang, H.-F. Road detection and corner extraction using high definition Lidar. *Inf. Technol. J.* **2010**, *9*, 1022–1030.
36. Fernandes, R.; Premebida, C.; Peixoto, P.; Wolf, D.; Nunes, U. Road detection using high resolution lidar. In Proceedings of the IEEE Vehicle Power and Propulsion Conference, Coimbra, Portugal, 27–30 October 2014.
37. Caltagirone, L.; Scheidegger, S.; Svensson, L.; Wahde, M. Fast LIDAR-based road detection using fully convolutional neural networks. In Proceedings of the IEEE Intelligent Vehicles Symposium, Los Angeles, CA, USA, 11–14 June 2017; pp. 1019–1024.
38. Fritsch, J.; Kuehnl, T.; Geiger, A. A new performance measure and evaluation benchmark for road detection algorithms. In Proceedings of the International Conference on Intelligent Transportation Systems, The Hague, The Netherlands, 6–9 October 2013; pp. 1693–1700.
39. EKINOXUM 1.3.1 Ekinox AHRS & INS. *Tactical Grade MEMS Inertial Sensors, User Manual*; SBG Systems: Rueil-Malmaison, France, 2015.
40. Teunissen, P., Montenbruck, O. (Eds.) *Handbook of Global Navigation Satellite Systems*; Springer: Cham, Switzerland, 2017; ISBN 978-3-319-42926-7.
41. Naver Map. Available online: <https://map.naver.com> (accessed on 1 July 2018).



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# Pano-RSOD: A Dataset and Benchmark for Panoramic Road Scene Object Detection

Yong Li <sup>1,†</sup>, Guofeng Tong <sup>1,†</sup>, Huashuai Gao <sup>1,†</sup>, Yuebin Wang <sup>2,3,\*</sup>, Liqiang Zhang <sup>3,\*</sup> and Huairong Chen <sup>1</sup>

<sup>1</sup> College of Information Science and Engineering, Northeastern University, Shenyang 110819, China; leoqiulin@126.com (Y.L.); tongguofeng@ise.neu.edu.cn (G.T.); gaohuashuai1995@163.com (H.G.); chenhr1993@163.com (H.C.)

<sup>2</sup> School of Land Science and Technology, China University of Geosciences, Beijing 100083, China

<sup>3</sup> The State Key Laboratory of Remote Sensing Science, Beijing Normal University, Beijing 100875, China

\* Correspondence: xxgcdxwyb@163.com (Y.W.); zhanglq@bnu.edu.cn (L.Z.); Tel.: +86-13911640890 (Y.W.); +86-13671186122 (L.Z.)

† The authors share the same contributions.

Received: 18 February 2019; Accepted: 11 March 2019; Published: 18 March 2019

**Abstract:** Panoramic images have a wide range of applications in many fields with their ability to perceive all-round information. Object detection based on panoramic images has certain advantages in terms of environment perception due to the characteristics of panoramic images, e.g., larger perspective. In recent years, deep learning methods have achieved remarkable results in image classification and object detection. Their performance depends on the large amount of training data. Therefore, a good training dataset is a prerequisite for the methods to achieve better recognition results. Then, we construct a benchmark named Pano-RSOD for panoramic road scene object detection. Pano-RSOD contains vehicles, pedestrians, traffic signs and guiding arrows. The objects of Pano-RSOD are labelled by bounding boxes in the images. Different from traditional object detection datasets, Pano-RSOD contains more objects in a panoramic image, and the high-resolution images have 360-degree environmental perception, more annotations, more small objects and diverse road scenes. The state-of-the-art deep learning algorithms are trained on Pano-RSOD for object detection, which demonstrates that Pano-RSOD is a useful benchmark, and it provides a better panoramic image training dataset for object detection tasks, especially for small and deformed objects.

**Keywords:** panoramic image dataset; road scene; object detection; deep learning; convolutional neural network

## 1. Introduction

Due to the wide availability of consumer-level panoramic video capturing and imaging devices, panoramic images are widely used in many fields [1–6]. For example, they are used in 360-degree object tracking [1,4], equirectangular super-resolution [3], privacy protection in Google Street View [5] and roadway inventory management about traffic signs [6]. Object detection based on panoramic images is one of the key technologies to make panoramic images widely applied. In intelligent transportation systems, the technology of object detection in panoramic images (with a wide field of view) can help autonomous driving assistance systems (ADAS) and autonomous navigation for unmanned aerial vehicles (UAV) detect the objects (e.g., vehicles, pedestrians) around the vehicle. From a map navigation perspective, panoramic maps, e.g., Baidu and Google among others, which are constructed by panoramic images, can express richer information such as location and scene. However, the information of pedestrians and vehicles in the panoramic map involves personal privacy and the speed of the private information removing or blurring based on manual methods is relatively slow.

Efficient and automatic object detection methods can be realized by deep learning for panoramic image object detection. Reference [5] proposed a probabilistic search algorithm to boost the efficiency of face detection in Google Street View so as to protect personal privacy. In smart city management and virtual reality, some object information can be retrieved and located through panoramic object detection. It can be seen that the research on panoramic object detection has important practical significance.

All of the research of panoramic object detection relies on a large-scale, high-quality training dataset. Although a variety of public datasets, e.g., PASCAL VOC [7], ImageNet [8] and COCO [9] are available for the identification and segmentation of multiple objects, they aim at generic object detection, not specific for panoramic object detection. Considering the difference between traditional images and panoramic images, the models pre-trained on the generic datasets are unsatisfactory commonly when directly applied in panoramic object detection. In addition, although there are also some street-view object detection datasets (including KITTI [10], Caltech Pedestrian Dataset [11] and UA-DETRAC Dataset [12]), these datasets mainly used for vehicle or pedestrian detection, street-view semantic and instance segmentation (including Mapillary Vistas [13], Cityscapes [14] and Apoloscope [15]). The publication of these datasets undoubtedly promotes the development of object detection. However, the public datasets specific to panoramic road scene object detection are still unavailable.

Moreover, to the best of our knowledge, there is no special object detection algorithm for panoramic images at present. The previous methods mainly use traditional hand-craft features, e.g., the histogram of oriented gradient (HOG), scale-invariant feature transform (SIFT), or the existing object detection methods based on transfer learning (e.g., pre-training in ImageNet [8] or COCO [9]). For example, Reference [16] adopted the detectors based on HOG algorithm to detect traffic signs from street-level panoramic images. Reference [17] used Faster Region-based CNN (RCNN) to detect object from indoor-level panoramic images.

Though much exciting progress on the object detection of road scenes has been extensively reported in recent years, there are two major issues that seriously limit the development of object detection in panoramic road scene images:

- A lack of panoramic object detection datasets for deep learning. A panoramic image usually contains more objects and some distorted objects due to its special imaging mechanism, which is different from the ordinary. Therefore, an object detection task for panoramic images needs a new panoramic dataset to train and test for the purpose of adapting the differences.
- Although the existing object detection methods of common images can be transferred to the panoramic object detection, there is a lack of model, evaluation statistics and benchmarks specifically for the panoramic object detection.

Aiming at the above problems, we construct a panoramic road scene object detection dataset (Pano-RSOD) and carry out experiments based on the state-of-the-art algorithms for object detection to construct a benchmark. The Pano-RSOD contains 9402 images and four categories objects, i.e., vehicles, pedestrians, traffic signs and guiding arrows. The constructed dataset is quantitatively and qualitatively compared with other datasets in several aspects, e.g., the number of object samples, the number of images, number of categories, resolution of images, the type of images and etc. Besides, we train five state-of-the-art detectors: faster RCNN (VGG-16) [18], faster RCNN (ResNet-101) [19], region-based fully convolutional networks (R-FCN) [20], YOLOv3 [21], and single shot multibox detector (SSD) [22,23]. The transfer learning method is adopted for the five detectors designed in this paper with the pre-trained models of ImageNet [8] and COCO [9]. Furthermore, the benchmark of Pano-RSOD is constructed.

In summary, the major contributions of this paper are as follows:

- We present a novel and promising topic for panoramic road scene object detection, which will have potential applications in ADAS, UAV and panoramic mapping. Compared with the normal view, a panoramic view can cover a larger perspective and contain more objects in one single image. It could be possible to cover some complicated situations which are not covering in the

most existing datasets. And the object detection on the online panoramic map is challenging. Thus, a panoramic road scene dataset is needed and important. In order to provide more research foundation for solving the object detection in panoramic image problems, related object detection methods and datasets are compressively overviewed (Section 2).

- We construct a high-resolution, panoramic road scene object detection dataset (Pano-RSOD) with more annotations and small object diversity. The data set is, to the best of our knowledge, the first high-resolution panoramic road scene dataset and the images are with high intraclass diversity. The dataset can provide a better experimental dataset for the object detection algorithm based on the panoramic image. Besides, the dataset can also evaluate the advantages and disadvantages of object detection algorithms, which aimed at small and deformed objects (Section 3).
- We introduce the baseline methods of the object detection (Section 4), and compare several state-of-the-art object detection algorithms, i.e., faster RCNN [18,19], R-FCN [20], YOLOv3 [21], SSD [22,23] trained with Pano-RSOD. These can serve as the baseline results for the future work (Section 5).

## 2. Related Works

This section mainly discusses the object detection datasets of road scene and networks for object detection. Thus, we summarized the related works from these two aspects.

### 2.1. Existing Object Detection Dataset of Road Scene

Usually, vehicles, pedestrians, traffic signs, etc. are the most common objects in road scenes. The detection of these objects has very broad applications. The most common datasets including the above objects are as follows:

- **Pascal VOC Dataset [7]:** This dataset is used as a standardized dataset for image detection and classification. There are two versions of voc2007 and voc2012. voc2007 has a total of 9963 images, and voc2012 has a total of 17,125 images. They include 20 categories to be detected, e.g., cars, pedestrians and etc. The image size in the dataset is different, and the horizontal image size is about  $500 \times 375$  pixels, and the vertical image size is about  $375 \times 500$  pixels. Each image corresponds to an xml format label file, which records the image size, ground-truth of object coordinates and other information. This dataset is widely used as an evaluation criterion in various object detection algorithms [18,20–22,24,25].
- **Object Detection Evaluation 2012 [10]:** This is a dataset for 2D object detection and azimuth estimation in the KITTI database. It consists of 7481 training images and 7518 test images. A total of 80,256 objects are marked, covering the car, pedestrian and cyclist. Among them, the precision-recall (PR) curve is used for object detection evaluation. The dataset has a wide range of applications in vehicle detection and pedestrian detection due to a large number of car and pedestrian samples in the dataset.
- **Pedestrian Detection Dataset:** Pedestrian detection is one of the important tasks in the fields of video surveillance and automatic driving. Therefore, the pedestrian detection dataset also plays an important role in evaluating various object detection algorithms. The INRIA person dataset [26] was created by Daal, where the training set contains 614 positive samples (including 2416 pedestrians) and 1218 negative samples, and the test set contains 288 positive samples (including 1126 pedestrians) and 453 negative samples. The dataset is currently used widely in static pedestrian detection. The NICTA Pedestrian Dataset [27] is a larger static pedestrian detection dataset at present. There are 25,551 images containing single pedestrian, 5207 high-resolution images containing non-pedestrian. The training set and test set have been divided to facilitate the comparison of different classifiers in the database. The Caltech pedestrian dataset [11] is currently a large pedestrian database, which is captured by a car camera in the urban traffic environment. The dataset is a video about 10 h, and the resolution is  $640 \times 480$ , 30 frames per second. The dataset labels about 250,000

bounding boxes (including 350,000 rectangles and 2300 pedestrians). In addition, there are other pedestrian detection datasets, such as ETH [28] and CVC [29].

- **Vehicle Detection Dataset:** Vehicle detection is a key step in vehicle analysis, and it is the basis for subsequent vehicle identification and vehicle feature recognition. Earlier, there is CBCL Car Database [30] created by MIT, which contains 516 images in ppm format with a resolution of  $128 \times 128$ , mainly using for vehicle detection. The UA-DETRAC dataset [12] is a larger vehicle detection and tracking dataset. It contains 10 h of video taken at different locations in Beijing and Tianjin, China. The resolution of video is  $960 \times 540$ , and the frame ratio is 25 frames per second. The dataset labels 8250 vehicles and 1.21 million object bounding boxes. BIT-Vehicle Dataset [31] covers 9850 images of bus, microbus, minivan, sedan, SUV, and truck, which can be used to evaluate the performance of multi-class vehicle detection algorithms.

However, public datasets specific for panoramic image detection remain unavailable. Therefore, the need for panoramic image detection dataset has become more urgent.

## 2.2. Object Detection Methods

The field of image classification by deep learning has great breakthroughs, and it also promotes the field of object detection to make great progress. Especially the convolutional neural network (CNN) plays a very important role in feature extraction [32]. Girshick et al. proposed regions with CNN features (RCNN) [33] in 2014, which applied CNN to object detection. Its extensions, fast RCNN [34] and faster RCNN [18] further improved the detection speed. After that, R-FCN [20], PVAnet [24], feature pyramid networks (FPN) [35], and mask R-CNN [36] have been improved and optimized on the basis of Faster RCNN, which improves the detection speed and accuracy. In addition, in order to meet the real-time requirements of some scenes, Redmon et al. proposed a regression-based one-stage method YOLO [37] based on OverFeat [38] in 2015. Then they proposed its extensions YOLOv2 [39] and the latest YOLOv3 [21]. Another branch of the one-stage methods implements the approach with multiple feature layers to predict, such as SSD [22], RainDow SSD (R-SSD) [40], and Deconvolutional Single Shot Detector (DSSD) [25].

At present, the object detection methods using deep learning can be mainly divided into two major categories: two-stage detection framework and one stage detection framework [41]. The former firstly generates the proposals in the proposal stage, and then use CNN to classify these proposals. The latter has no proposal stage, and directly converts the problem of object positioning into regression problem. The comparison results of typical object detection algorithms based on deep learning are shown in Table 1.

**Table 1.** Comparison of typical object detection algorithms based on deep learning. The symbol \* represents the multi-feature layer fusion. Methods evaluated in this work are bold-faced.

Methods	Two-Stage			One-Stage				
	Fast RCNN	Faster RCNN	PVAnet	R-FCN	YOLO	YOLOv3	SSD	DSSD
Region proposal	selective search [42]	RPN [18]	RPN	RPN	grid cells	anchor boxes	default boxes	default boxes
Prediction layer	One	one	One *	one	one	Multiple *	multiple	Multiple *

### A. two-stage detection framework.

RCNN is the basis of most current two-stage detection framework. It firstly uses the selective search [42] algorithm to generate candidate bounding boxes of interest. Then each proposal is sent to the CNN network for feature extraction to generate feature vectors. Finally, support vector machine (SVM) is used for classification. Its extension fast RCNN optimizes the runtime of the algorithm.

Faster RCNN further reduces the running time of the algorithm, and it designs the region of proposal (RPN) [18], which directly generates proposals without increasing the computation cost.

This way end-to-end object detection can be achieved, and computational cost is reduced. Besides, the problem of algorithm accuracy reduction caused by excessive proposals can be avoided.

In addition, many two-stage methods have been improved based on Faster RCNN. For example, PVANet [24] optimizes the feature extraction network and proposes a lightweight network. Besides, R-FCN [8] introduces position-sensitive score maps on the basis of Faster R-CNN, and the feature sharing can be realized on the whole image and the detection speed is improved.

### B. one-stage detection framework

Like YOLO [37], its upgraded version of YOLOv3 [21], SSD [22], DSSD [25] and other one stage detection frameworks have no obvious proposal stage. YOLO directly performs feature extraction, candidate bounding boxes regression and classification in the same convolution network. This method performs poorly for detection of small and multiple objects appearing in the same grid cell. Then, its extension YOLOv3 proposes the Darknet53 [39] and implements a multi-scale prediction method according to FPN [35] so as to obtain better predictions under the premise of speed increase.

SSD sets discretized and multi-scale default boxes on feature maps with different resolutions (SSD512 uses 7 layers). Meanwhile, small convolution kernels are added to each feature map as the final prediction layer to complete classification and the bounding box regression. DSSD changes the feature extraction network from VGG-16 [43] to ResNet-101 [19] to enhance the network feature extraction capability. At the same time, deconvolution is used to extract contextual semantic information so as to improve the detection accuracy of small objects.

## 3. Panoramic Road Scene Object Detection Dataset

### 3.1. New Dataset for Object Detection of Road Scene (Pano-RSOD)

Currently, there are many public datasets used for object detection, but most of them are not panoramic images. Besides, there are relatively few datasets of large traffic scene images. In recent years, with the development of panoramic imaging technology, panoramic images have had obvious advantages over traditional images in terms of overall scene perception. Then, panoramic image can be more widely used in digital cities, intelligent transportation and automatic driving. Therefore, we construct a panoramic road scene object detection dataset, namely, **Pano-RSOD** (Dataset link: <https://pan.baidu.com/s/1H9RsXfXCCfBgpF2bY2LGeA>).

The Pano-RSOD is captured from the streetscape of downtown Zhongshan City, Guangdong Province, China. It contains a total of 9402 images. The size of each image is  $2048 \times 1024$  pixels. The labels are produced in PASCAL VOC format, including vehicles (50,255 bounding boxes), pedestrians (11,227), traffic signs (8622) and guiding arrows (17,438). Each image averagely contains about nine objects. For an easier representation of our dataset, we use a car, person, sign and line to represent vehicles, pedestrians, traffic signs and guiding arrows in the remaining of the paper.

In general, the Pano-RSOD is a multi-scale panoramic object detection dataset in road scenes, and there are more objects in a single panoramic image. Besides, the panoramic image contains a large number of small objects. It is also important to point out that objects in panoramic images are often accompanied by distortion. Therefore, the Pano-RSOD provides data sources for training, test and evaluation of object detection algorithms aimed at panoramic image, objects with distortion or small objects in road scenes. Some example images of Pano-RSOD are shown in Figure 1.



Figure 1. Cont.



Figure 1. Different road scenes of panoramic images. (a) crossroad; (b) overpass; (c) urban road; (d) suburban road.

### 3.2. Dataset Construction

#### 3.2.1. Panoramic Image Acquisition

In order to construct a road scene panoramic image dataset, a panoramic image acquisition system is constructed. The system is composed of a multi-camera panoramic vision system (i.e., Ladybug5) and a vehicle. The images are collected through the system by driving the vehicle in different road scenes. The panoramic image acquisition vehicle is shown in Figure 2.

The multi-camera panoramic vision system uses multiple sub-cameras distributed in different orientations to acquire image information that can be perceived by the current viewpoint. The panoramic image of the Ladybug5 satisfies the spherical camera theory, which can establish the projection relationship between each sub-image and panoramic image. As shown in Figure 3, the right panoramic image can be acquired by the left multi-camera panoramic vision system. The multi-camera panoramic vision system provides high-resolution, dead-band panoramic images with the synchronization and fast speed of data acquisition.

In the process of panoramic image acquisition, we record the location of image collection, i.e., the name of the road. The images of the training set, the validation set and the test set of the Pano-RSOD all come from different roads of the city to avoid the repetition. In addition, in order to avoid images having large similarities in the same set, every 15 frames of the image sequences are firstly adopted for dataset construction, and we manually remove images with large similarities.



Figure 2. Panoramic image acquisition vehicle.



**Figure 3.** The multi-camera panoramic vision system and the acquired panoramic image.

### 3.2.2. Dataset Labeling

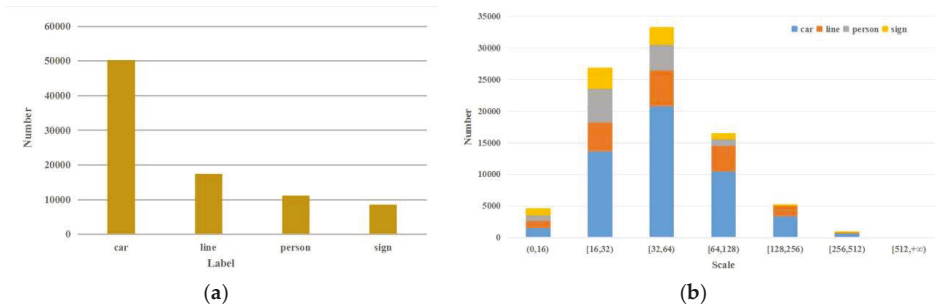
Making dataset labels is an important part of image classification, object detection and segmentation results. The quality of label making is directly related to the final accuracy of the training model. In this paper, we use an open source image annotation tool on GitHub, namely, LabelImg (<https://github.com/tzutalin/labelImg>). The output is the xml file, which is the same as PASCAL VOC [7].

In the field of intelligent transportation and panorama mapping, the detection of vehicles, pedestrians, traffic signs and guiding arrows often plays an important role. Thus, this paper selects those four most common objects in traffic road scenes to label. When labeling, we try to completely cover the object with the rectangular bounding box. Besides, car class only labels vehicles with four wheels, person class contains any people, e.g., walking, standing, sitting or riding people, sign class includes any traffic sign in the traffic scene, line class only labels all kinds of guiding arrows on the roads.

In order to build high quality datasets, we set strict control over the data labeling process. Ten researchers who study the object detection are asked to process the data. These ten researchers are divided into two groups on average. For each image, five researchers (the first group) are arranged to manually annotate the images, including the object category label and the coordinates of the rectangular box. After all the images have been labelled, we asked the other five researchers (the second group) to check the labelled data. Then, the voting method determines whether to pass the verification. If more than three persons pass the vote, the image is verified to pass. Otherwise it is relabeled until the checking passes. In the end, we have labeled 4 categories with a total of 87,542 object bounding boxes.

### 3.2.3. Dataset Statistics and Analysis

Our road scene panoramic image dataset contains a large number of labeled samples. Each class has sufficient samples (the minimum number of samples for a category is more than 8500). The sample information of the vehicle is the most abundant, and the minimum number of traffic signs is more than 8500. Moreover, each type of sample is acquired from different road traffic scenarios, such as a city intersection, suburban road, and urban road, which can provide rich foreground and background feature information for CNN feature extraction. In addition, the dataset contains a large number of small objects and objects with occlusion and overlap. This can increase the difficulties of the object detection task, which can also help us evaluate the advantages and disadvantages of the object detection algorithms. Figure 4a counts the number of objects for each type of dataset. Figure 4b counts the number of objects at different scales. Specifically: approximately 37% of objects are small (scale  $\leq 32$ ), 56% are medium ( $32 < \text{scale} \leq 128$ ), and 7% are large (scale  $> 128$ ).



**Figure 4.** Dataset statistics results. (a) Number of objects per category; (b) Number of objects per scale. We define scale as the square root of the object’s area.

The Pascal VOC dataset has a relatively small image size and few object types. Besides, there are relatively few traffic scenarios and traffic objects. Compared with UA-DETRAC Dataset [12] and BIT-Vehicle Dataset [31], the Pano-RSOD includes richer background information, and it covers different traffic road scenes such as urban road, crossroad, overpass, and suburban road, which can maximize the diversity of the background. In addition, most of the pedestrian and vehicle datasets are only for a single type of object, and the number of objects in the scene is relatively small, which is not suitable for object detection in complex traffic scenarios.

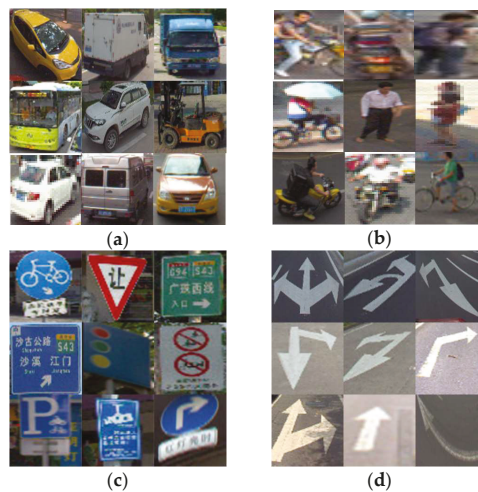
Compared with Pascal VOC Dataset which contains relatively few traffic scenarios and traffic objects, the panoramic images in the Pano-RSOD are high-resolution, and the average number of objects per image is up to 9, so that the object detection is not needed to use more images to train. Compared with 10,053 labeled vehicles in BIT-Vehicle Dataset, Pano-RSOD contains up to 50,255 labeled vehicles (with wider scale), which can be used for vehicle detection and other tasks. Compared with UA-DETRAC Dataset, the Pano-RSOD includes richer background information, and it covers different traffic road scenes such as urban road, crossroad, overpass, and suburban road, which can maximize the diversity of the background. In contrast with cityscapes dataset [13], mapillary vistas (Images with strong wide-angle view or 360-degree images were removed) [14] which are both used for semantic street-level understanding, the images in Pano-RSOD have a 360-degree angle of view instead of single view, so that they can contain more objects with various scales and perceive the whole road scene in single image. Of course, other datasets are not panoramic images that are essentially different from Pano-RSOD, and we just give roughly qualitative comparison. Table 2 lists the comparison results of the Pano-RSOD and the existing object detection datasets. Compared with other road scene object detection datasets, the dataset of this paper has the following characteristics:

**Table 2.** Comparison of Statistical Results of Pano-RSOD and Other Datasets.

Dataset	Pascal VOC 2007	Object Detection Evaluation 2012	BIT-Vehicle	Pano-RSOD	UA-DETRAC	Cityscapes (Semantic)	Mapillary Vistas (Semantic)
Panorama	No	No	No	Yes	No	No	No
Traffic Scene	No	Yes	Yes	Yes	Yes	Yes	Yes
Resolution	$\sim 375 \times 500$	$1240 \times 376$	$1600 \times 1200$ $1920 \times 1080$	$2048 \times 1024$	$960 \times 540$	$2048 \times 1024$	$1920 \times 1080$
Number of Categories	20	3	6	4	4	30	18 (object)
Number of Cars	2500	Unknown	10,053	50,255	8250	Unknown	~200 thousand
Number of Images	9963	14,999	9850	9402	140 thousand	25,000	25,000
Number of Samples	24,640	80,256	10,053	87,542	1.21 million	Unknown	Unknown



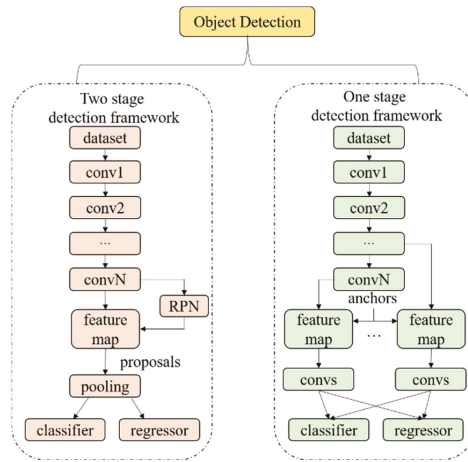
- **Panorama:** According to the information collected from the Internet, the current public object detection datasets are basically not panoramic images, but our road scene panoramic dataset can provide a good reference for the panoramic technology applied in the object detection. Besides, objects in panoramic image often have distortion which provides a challenge for object detection task.
- **Large-scale and high resolution:** According to the comparison results in Table 2, the Pano-RSOD has more labeled sample sizes, especially with the most abundant vehicle information. Besides, the Pano-RSOD has higher image resolution.
- **Multi-scales and more small objects:** As can be seen from Figure 4b, the Pano-RSOD has a wide range of scales. Especially for small objects, it has as many as 31,579 samples with a scale less than 32.
- **Diversity:** Figure 5 lists the objects of different types of labels in the Pano-RSOD. As shown in Figure 5, the Pano-RSOD is rich in object types. For instance, vehicle samples cover different types (e.g., truck, sedan, bus, SUV, taxi, etc.), orientations and scales, person samples include cycling, walking, standing, crowded people, traffic sign samples contain different shapes, sizes, colors and contents, guide arrow samples have different shapes, colors, and directions of representation.



**Figure 5.** The different types of objects in Pano-RSOD. (a) car; (b) person; (c) sign; (d) line.

#### 4. Baseline Methods

Since the top rank methods for object detection in the PASCAL VOC or KITTI dataset has recently adopted a convolutional neural network, we chose the baseline methods based on CNN. In this section, we evaluate different object detection algorithms based on one stage detection framework and a two-stage detection framework reviewed in Section 2.2. A simple algorithm flow diagram about two kinds of methods used in this paper is shown in Figure 6.



**Figure 6.** General pipeline of two types of object detection baseline methods adopted in this paper. The difference between two kinds of methods is described in Section 2.2, and we only give a simple and intuitive diagram.

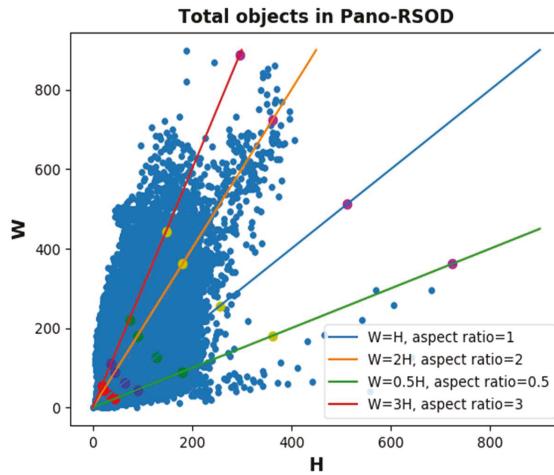
**Anchor box settings.** In the training and prediction stages, the five baseline detectors in this paper use the method of pre-setting anchor boxes, which provide the reference for final prediction (bounding boxes). If the anchors are not set properly, it will inevitably lead to more positional regression errors. Therefore, it is especially important to set the anchors with appropriate scales and aspect ratios. In this paper, the data distribution of Pano-RSOD is statistically analyzed. The scale distribution is shown in Figure 4b. The length and width of object are clustered by K-means algorithm, as illustrated in Table 3. Since the number of objects detected in this paper is mainly divided into four categories, the number of clusters is set to be 4.

**Table 3.** Different aspect ratios after clustering. The second and third column, i.e., H and W, separately height and width of objects in Pano-RSOD after clustering. The last column, i.e., aspect ratio, is calculated by dividing W by H.

Terms	H				W				Aspect Ratio			
Classes	Car	Sign	Line	Person	Car	Sign	Line	Person	Car	Sign	Line	Person
First	61	26	129	32	98	24	465	18	1.6	0.9	3.6	0.6
Second	118	146	81	114	211	135	239	63	1.8	0.9	1.7	0.6
Third	28	187	26	139	41	396	39	245	1.5	2.1	1.5	1.8
Fourth	182	67	68	64	435	57	114	33	2.4	0.9	1.8	0.5

Considering the scale distribution of object as shown in Figure 4b, the aspect ratio after clustering as shown in Table 3, and the hardware conditions of the experiment, the scale of anchors in Faster RCNN and R-FCN is {32, 64, 128, 256, 512} and the aspect ratio is {0.5, 1, 2, 3}. Figure 7 shows the distribution of anchors in the dataset. It can be seen that the anchors with scales and aspect ratios used can cover the entire samples to a great extent. For the SSD, an additional convolutional layer is added on the basic feature extraction network InceptionV2 [23]. It generates a total of six feature layers to predict. The scale and aspect ratio settings of the anchors are calculated using the method of Ref. [22], and each prediction layer sets anchors with multiple scales and aspect ratios. YOLOv3 performed k-means clustering on the object sizes of the training set (using the IoU value as the distance

indicator) [21] to set up 9 different anchors. Table 4 shows the detailed parameters settings for anchors of the baseline methods in this paper.



**Figure 7.** The distribution of anchors in the dataset (separate points are the distribution values of the anchors, and the gradient of the line represents the aspect ratio).

**Better feature extraction network.** Designing better feature extraction network can provide more information for object detection task. Compared with VGG-16, ResNet-101 has characteristics of low complexity, deeper network and higher accuracy for classification. Thus, we also use ResNet-101 instead of VGG-16 in the feature extraction network of Faster-RCNN and R-FCN to improve the feature extraction ability.

**Training strategy and model parameters.** We use SGD as backpropagation algorithm for the five detectors and stepwise reduce the learning rate. Considering the depth of the network and other factors for each detector, the proper iteration steps and initial learning rates are set to ensure the convergence of the network. For the relatively deep network, we set the smaller initial learning rate to avoid gradient explosion. The iteration steps of Faster RCNN (VGG-16 based and ResNet-101 based) and R-FCN are both 100 k steps, and the initial learning rates are set to  $1 \times 10^{-2}$ ,  $1 \times 10^{-3}$  and  $1 \times 10^{-3}$ , respectively. And then the learning rates are reduced to one-tenth of the original at the 80 k steps. The iteration steps of SSD and YOLOv3 are both 150 k steps. Their initial learning rates are  $4 \times 10^{-3}$  and  $1 \times 10^{-3}$ , respectively. Then the learning rates drop to one-tenth of the original at 80 k steps. The selected hyper-parameters for the five detectors are shown in Table 5.

**Table 4.** Detailed setting parameters of five detectors' anchors. We define the type of scales as S, the type of aspect ratios as A, so for Faster RCNN, R-FCN, and SSD, the type of anchors as  $S \times A$ . For YOLOv3, we directly list anchor's width and height.

Methods	Scale	Aspect Ratio	Anchor Type Number
Faster RCNN (VGG-16)	{32,64,128,256,512}	{0.5,1,2,3}	20
Faster RCNN (ResNet-101)	{32,64,128,256,512}	{0.5,1,2,3}	20
R-FCN (ResNet-101)	{32,64,128,256,512}	{0.5,1,2,3}	20
SSD (InceptionV2)	First layer:51.2	{0.5,1,1,2}	30
	Second layer:102.4	{0.5,0.333,1,1,2,3}	
	Third layer:198.4	{0.5,0.333,1,1,2,3}	
	Fourth layer:294.4	{0.5,0.333,1,1,2,3}	
	Fifth layer:390.4	{0.5,1,1,2}	
Sixth layer:486.4	{0.5,1,1,2}		
YOLOv3 (DarkNet53)	{[14.96,13], [27.04,17], [21.1,32.06], [39.94,23.96], [38.1,51], [64.10,32.06], [80.08,52.02], [128,82.02], [291.02,142.04]}		9

**Table 5.** Hyper-parameters used in training process.

Hyper-Parameters	Faster RCNN (VGG-16)	Faster RCNN (ResNet-101)	R-FCN (ResNet-101)	SSD (InceptionV2)	YOLOv3 (Darknet53)
Steps	100 k	100 k	100 k	150 k	150 k
Initial learning rate	0.01	0.001	0.001	0.004	0.001
Batch size	1	1	1	4	4
Momentum	0.9	0.9	0.9	0.9	0.9
IoU threshold	0.5	0.5	0.5	0.5	0.5

## 5. Experiments and Benchmark Statistics

In order to test the dataset and build a benchmark for the Pano-RSOD, we train and test the state-of-the-art algorithms (Faster-RCNN, R-FCN, SSD and YOLOv3) on the Pano-RSOD. Among the 9402 images of the datasets, 7000 images are manually selected as training set, 1000 images selected as test set, and 1402 images are used as validation set to detect four classes of objects, i.e., car, person, sign and line. The images of training set, validation set and test set are collected from different roads of the city, and they all cover urban and suburban scenes. In the experiment, the transfer learning method is implemented, and the network is fine-tuning with the pre-training model [44]. Faster RCNN(VGG-16) and YOLOv3 used the pre-training model based on ImageNet [8], and the other three detectors use the pre-training model based on COCO [9]. Besides, the training and testing images are resized to the fixed size  $1024 \times 512$  pixels for all the detectors.

All evaluations are done on Intel Core i7-3930 k (3.80 GHz) CPU (24 GB memory), a single TESLA P100 GPU (16 G memory). YOLOv3 is carried out experiment based on Darknet framework while the other detectors based on Tensorflow framework.

### 5.1. Evaluation Metrics

Currently, the values of average precision (AP) and mean average precision (mAP) are used to evaluate the performance of the object detection algorithms [33–40]. In order to compare performance of the state-of-the-art object detection algorithms on the Pano-RSOD, we use AP and mAP to evaluate the detection results of each category and all categories for every learned model, respectively.

If intersection-over-union (IoU) of the detection result and ground truth bounding box is larger than the given threshold, the object can be detected, namely, true positive ( $tp$ ). If multiple detection results matching with ground truth, the one with largest IoU is the  $tp$ , and others are false positives ( $fp$ ). After matching all the detection results, all the ground-truth without detection results matched are false negatives ( $fn$ ). All the detection results without ground-truth matched are false positives ( $fp$ ). The equation of the AP calculation is as follow:

$$AP = \frac{1}{11} \sum_{r \in \{0.0, 0.1, \dots, 1\}} \max_{R(c): R(c) \geq r} P(R(c)) \quad (1)$$

where the recall  $R(c) = tp(c)/(tp(c) + fn(c))$ ,  $P(R(c)) = tp(c)/(tp(c) + fp(c))$ , both for a given confidence threshold  $c$ , i.e., IoU.

mAP is calculated according to the AP of each category, and the calculation equation is as follows:

$$mAP = \frac{1}{n} \sum_{i=1}^n AP(i) \quad (2)$$

where  $n$  is the number of object classes.

We use two metrics in the next evaluation, i.e., AP@0.5(PASCAL VOC's metric [7]) and AP@0.5:0.95 (COCO's metric [9]). While the former is computed at a single IoU of 0.5, the latter are averaged over multiple IoU values, i.e., ten IoU thresholds from 0.5 to 0.95 with equal difference 0.5. All the abbreviated forms of AP and mAP refer to AP@0.5 and the mean of each AP@0.5 in the remaining of the paper.

## 5.2. Benchmark Statistics

### 5.2.1. Qualitative Evaluation

In order to give a qualitative analysis of the performance of different detectors, we show the object detection results of the five baseline methods in four road scenes. As shown in Figure 8, the detection results for small objects are not performed well by the baselines methods. Besides, there are some missing and false detection results. For example, as shown in Figure 8a, some vehicles with larger size are not detected by faster RCNN(VGG-16) and R-FCN. The advertising board is mistakenly detected as traffic sign by faster RCNN(VGG-16), as shown in Figure 8c. This also reflects the diversity of background information in Pano-RSOD, which poses a severe challenge to object detection in large scene. Thus, how to correct the background and foreground is still the key task to improve the detection performance of our dataset.

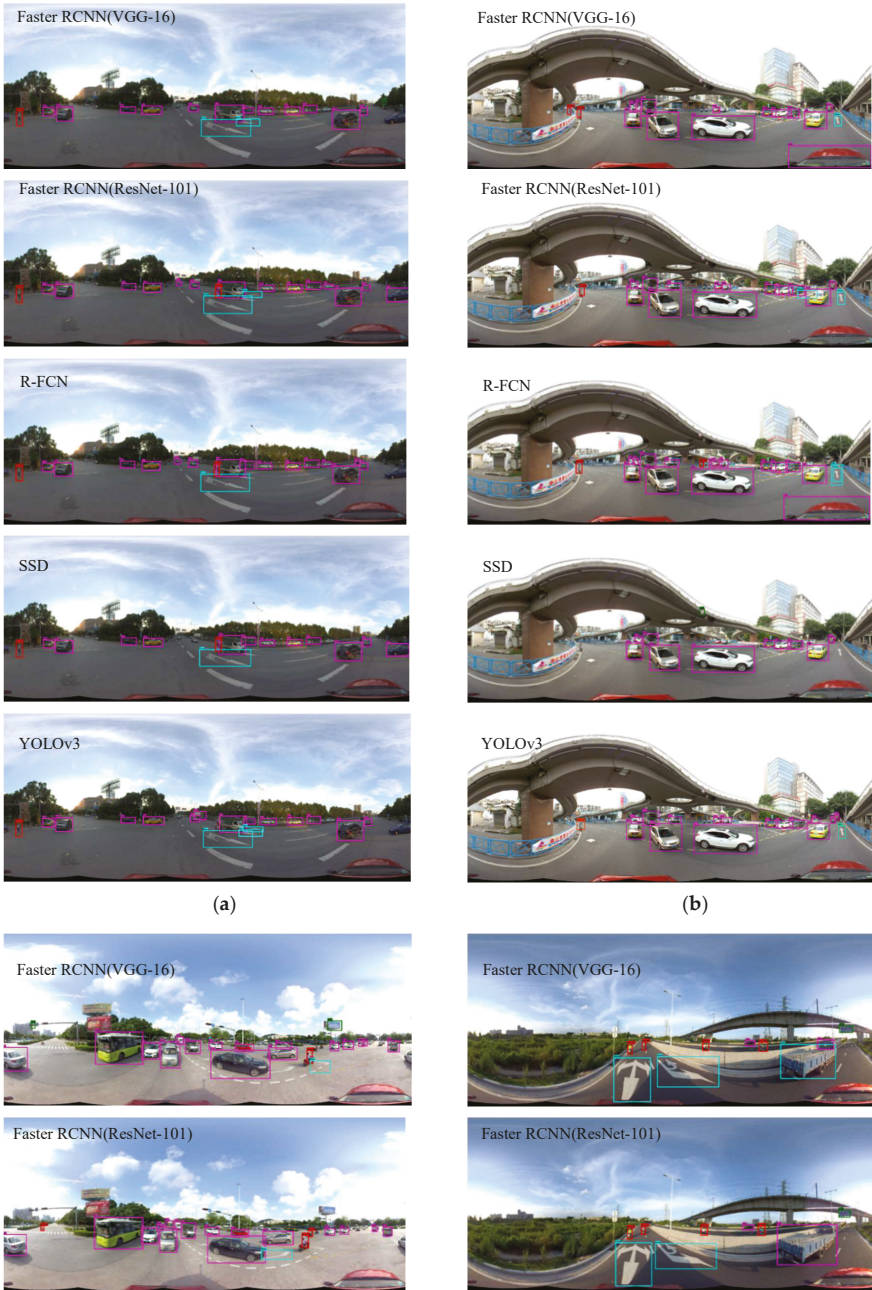
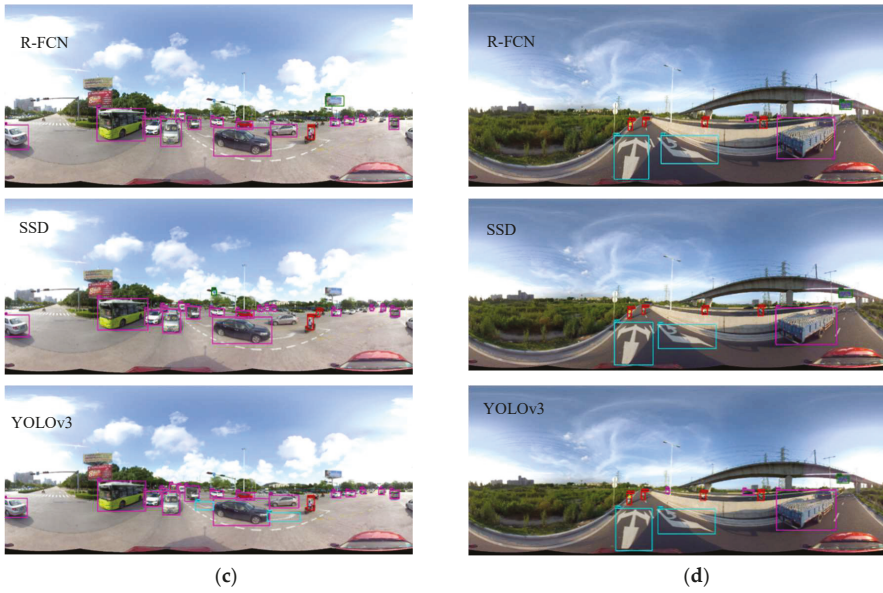


Figure 8. Cont.



**Figure 8.** The detection results of five algorithms in different road scenes. (a) crossroad; (b) overpasses; (c) crowded urban road; (d) suburban road. The text in the upper left corner of each image represents the algorithm adopted in the paper. The magenta box, red box, green box and cyan box separately represent car, person, sign and line.

### 5.2.2. Quantitative Evaluation

In order to quantitatively analyze the performance of various algorithms, we evaluate and compare the performance of five detectors through mAP metric, and analyze the difficulty of the object detection of different categories by AP metric. In addition, we count the time required for the detector to test each image to measure the speed of the algorithm. Table 6 shows the specific performance statistics for different detectors. For a more direct comparison of the detection performance of the detectors, we plot the precision-recall curve per category of each detector with the IoU threshold 0.5. The specific results are shown in Figure 9.

**Table 6.** Performance Statistics of Object Detection Using Different Algorithms. The best results are bold-faced.

Method	Car	Person	Sign	Line	mAP	Speed(ms)
Faster RCNN (VGG-16)	81.17	52.02	58.46	73.52	66.29	~58
Faster RCNN (ResNet-101)	82.56	58.93	<b>63.59</b>	<b>77.15</b>	70.56	31.58
R-FCN (ResNet-101)	81.22	55.46	61.96	74.79	68.36	25.41
SSD (InceptionV2)	77.36	48.96	51.82	69.00	61.79	16.38
YOLOv3 (Darknet53)	<b>83.60</b>	<b>65.06</b>	61.53	77.13	<b>71.83</b>	<b>~13</b>

As shown in Figure 9 and Table 6, from the overall mAP, YOLOv3 has achieved top performance, which is mainly due to its reference to the structure of FPN [10] feature pyramids. That structure

combines low-resolution, semantically strong features with high-resolution, semantically weak features. It shows that the detection of a person has a large advantage. It can be seen that the AP of the person is 6.13 percentage points higher than the second best Faster RCNN (ResNet-101). From the performances of the detectors for each category, the car category gets the best performance, and the person category gets the worst performance (YOLOv3 is a counter-example). This is mainly because the car category has more training samples than the person category, and can provide more feature information. What's more, there is also a considerable relationship with almost small objects of the person category in the images (as shown in Figure 4b, its scale is almost less than 64). In addition, the mAP of Faster RCNN (ResNet-101) is 4.27 higher than Faster RCNN (VGG-16). It can be seen that a better feature extraction network is very helpful for object detection tasks. On the whole, SSD gains slightly weaker performance. We assume that it can be a lack of higher-quality proposals compared to faster RCNN or R-FCN and not added to semantic information in context compared to YOLOv3. To sum up, these elements, i.e., better feature extraction network, higher-quality proposals and richer semantic information, all contribute to the promotion of detection performance in Pano-RSOD.

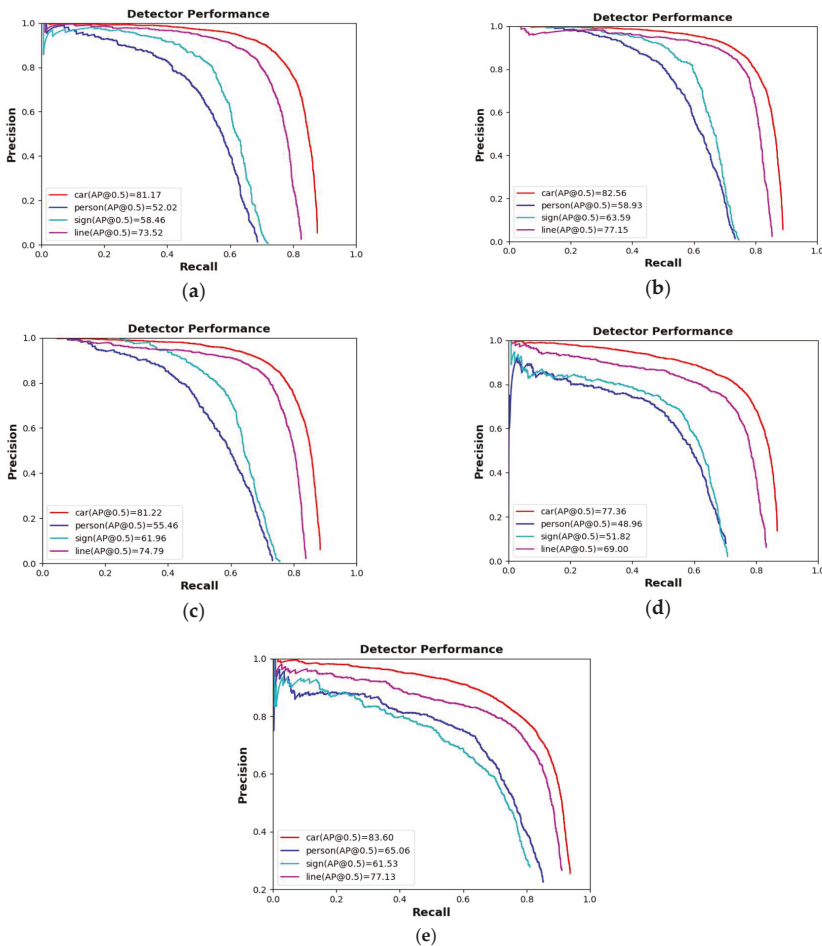


Figure 9. Performance comparison of different detectors. (a) Faster RCNN(VGG-16); (b) Faster RCNN(ResNet-101); (c) R-FCN(ResNet-101); (d) SSD(InceptionV2); (e) YOLOv3(Darknet53).



In terms of the speed of the algorithms, YOLOv3 also achieves top performance. On the one hand, YOLOv3 uses the Darknet deep learning framework, which is written in C language, to improve the running time of the program. On the other hand, it mainly benefits from its network structure and matching mechanism optimization between anchor and the ground-truth, such as the plenty of  $1 \times 1$  convolution and shortcut structures in Darknet53, each ground-truth only matches one a priori box, which greatly reduces the complexity of the model.

SSD, which is an end-to-end detection method, also achieves good performance. In addition, R-FCN replaces the RoI pooling layer and the fully connected layer of faster RCNN with position-sensitive score maps composed of full convolutional layers, which reduces the computational complexity of the head and increases the prediction speed by 6.17 ms.

For a more intuitive analysis of the five detectors, we have drawn their speed versus accuracy diagram, as shown in Figure 10. It can be seen that Faster RCNN and R-FCN are significantly better than SSD in terms of detection accuracy. With regard to speed, the result is the opposite. For example, the mAP of Faster RCNN (ResNet-101) as the second-best result, is 8.77 percentage points higher than SSD, and the speed of faster RCNN (ResNet-101) is slower than the SSD. Besides, YOLOv3 has a good trade-off in terms of speed and accuracy, and achieved the best performance.

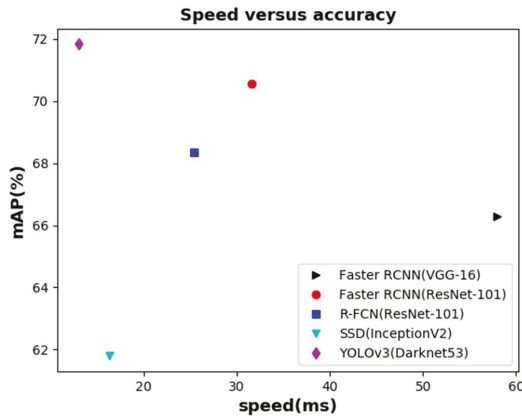


Figure 10. Speed (ms) versus accuracy (mAP) on Pano-RSOD.

### 5.2.3. Comparisons with a General Dataset

To demonstrate the differences between the models trained on conventional non-panoramic datasets and the model trained on the Pano-RSOD, we compare its performance with other object detection datasets, i.e., COCO, KITTI and UA-DETRAC. Based on the experimental results of the five baseline detectors in Table 6, we find the YOLOv3 has a good trade-off in terms of detection accuracy and speed. Therefore, we choose the YOLOv3 (with pre-trained model) as the baseline method in the following comparative experiments. We totally set up five comparative experiments: COCO as training set and Pano-RSOD as test set, KITTI as training set and Pano-RSOD as test set, Pano-RSOD as training set and KITTI as test set, Pano-RSOD as training set and UA-DETRAC as test set, Pano-RSOD as training set and test set. Since common categories between Pano-RSOD and COCO, KITTI are vehicle and pedestrian, we used these two categories for experiments for fair comparisons.

Table 7 shows the experimental results of different training set in terms of AP, on conditions that IoU threshold is set 0.5. It is obvious that the model trained on COCO has a poor performance in panoramic dataset, i.e., Pano-RSOD. The reasons can be summarized in two aspects: (1) the traffic scenes in COCO are relatively few. (2) the images in Pano-RSOD are different from the COCO because Pano-RSOD's panorama attribute will bring some optical distortions. Although the KITTI is a large-scale street-level object detection dataset whose scene is the same with Pano-RSOD, the AP of

model trained on KITTI is still about 20% lower than the model trained on Pano-RSOD. This is good evidence that models trained on conventional non-panoramic imagery perform worse than trained on panoramic images. In contrast, when testing on KITTI and UA-DETRAC, the models trained on Pano-RSOD can achieve relatively good results due to the diversities of object scales of Pano-RSOD.

Table 7. Detection Results of Different Training Set.

Training	Test	Vehicle (AP)	Pedestrian (AP)
COCO	Pano-RSOD	40.75	28.92
KITTI	Pano-RSOD	62.18	47.53
Pano-RSOD	KITTI	64.46	39.25
Pano-RSOD	UA-DETRAC	57.21	-
Pano-RSOD	Pano-RSOD	83.60	65.06

5.2.4. About Robustness of Detectors for Small Object

To evaluate the robustness of these detectors against varying IoU threshold, we evaluate five detectors with AP@0.5:0.95(COCO’s metric).

From Table 8, we know that the accuracy of each algorithm is significantly reduced when the COCO evaluation metric is adopted, which indicates that the object detection algorithm is particularly sensitive to the selection of IoU threshold.

Table 8. AP@0.5:0.95 of five detectors.

Method	Faster RCNN (VGG-16)	Faster RCNN (ResNet-101)	R-FCN (ResNet-101)	SSD (InceptionV2)	YOLOv3 (Darknet53)
AP@0.5:0.95	35.70	38.80	37.00	26.10	29.48

Then we increase the threshold from 0.4 to 0.8 by 0.1 increments and calculate AP regarding to each IoU threshold for each detector and plot IoU versus AP curve. The results are shown in Figure 11. As we can be seen from Figure 11, when the matching IoU value increases, the person category for every detector has a much sharper drop in the AP value than the car category, and falls to the worst result when IoU = 0.8. Such case implies that the detected bounding boxes do not have a high overlap ratio with the ground-truth and detection of small objects is more sensitive to IoU values. Therefore, more effort should be put into developing detectors that can better handle small objects for Pano-RSOD.

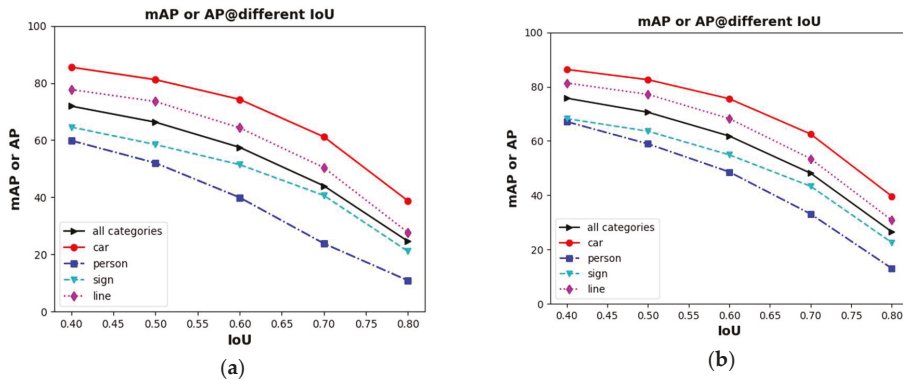
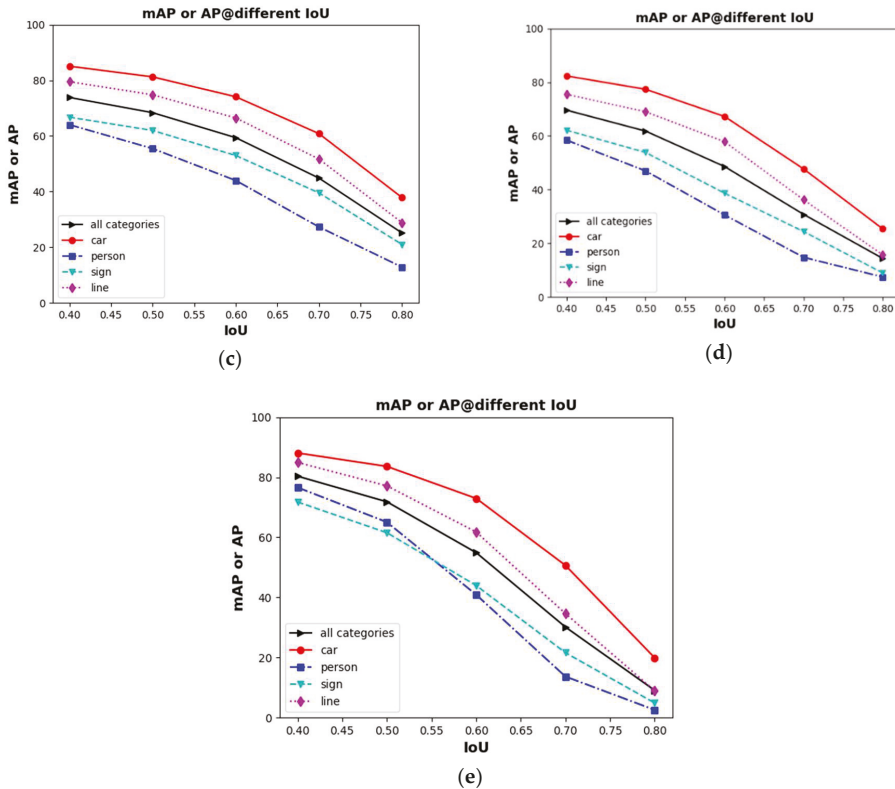


Figure 11. Cont.



**Figure 11.** Detection performance under different IoU threshold for each detector when predicting. (a) Faster RCNN(VGG-16); (b) Faster RCNN(ResNet-101); (c) R-FCN(ResNet-101); (d) SSD(InceptionV2); (e) YOLOv3 (Darknet53).

## 6. Conclusions

Pano-RSOD is a panoramic road scene dataset for object detection. It has distinctive characteristics: high-resolution, panorama, the richness of annotations and small objects, and diversity. Experiments have been conducted with different object detection algorithms based on deep neural networks. From the experimental results, we can conclude that Pano-RSOD can be used as a benchmark for performance evaluations of object detection. In that benchmark, YOLOv3 (Darknet53) has achieved the best results of AP (Car and Person), mAP and speed. While, the best results of AP (sign and line) and AP@0.5:0.95(COCO’s metric) have been achieved by Faster RCNN(ResNet-101).

However, there are still challenges, such as the detection of small and hidden objects, and the panoramic view distortion. In future work, the method of dealing with the panoramic view distortion or directly converting from the panoramic view to normal view can be added to the new object detection algorithm. Further, object detection using new structures, like spherical CNN, which can directly process from the panoramic view, can be proposed. Besides, we also plan to extend Pano-RSOD and apply the dataset to other tasks such as semantic or instance segmentation in panoramic scene.

**Author Contributions:** Conceptualization, Y.L., H.G. and G.T.; methodology, Y.L., H.G. and Y.W.; software, Y.L., H.G. and H.C.; formal analysis, Y.L., Y.W. and L.Z.; data curation, Y.L., G.T., H.G. and H.C.; writing—original draft preparation, Y.L. and H.G.; supervision, G.T. and L.Z.; funding acquisition, Y.W. and L.Z.

**Funding:** This research was funded by National Natural Science Foundation of China, grant number 41801241 and 41371324. The APC was funded by Y.B Wang and L.Q. Zhang.

**Acknowledgments:** The authors would like to thank Shanshan Yin, Wenbo Zhao, Hao Wang, Liwei Gao, Tingting Zhu, Mantang Liu, Lin Yang and Changjian Ge in Northeastern University for helping to build the dataset of this paper.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Bertozzi, M.; Castangia, L.; Cattani, S.; Prioletti, A.; Versari, P. 360° Detection and tracking algorithm of both pedestrian and vehicle using fisheye images. In Proceedings of the 2015 IEEE Intelligent Vehicles Symposium (IV), Seoul, Korea, 28 June–1 July 2015; pp. 132–137.
2. Lin, M.; Xu, G.; Ren, X.; Xu, K. Cylindrical panoramic image stitching method based on multi-cameras. In Proceedings of the 2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), Shenyang, China, 8–12 June 2015; pp. 1091–1096.
3. Fakour-Sevom, V.; Guldogan, E.; Kämäräinen, J.K. 360 panorama super-resolution using deep convolutional networks. In Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, Funchal, Portugal, 27–29 January 2018; pp. 159–165.
4. Kart, U.; Kamarainen, J.K.; Fan, L.; Gabbouj, M. Evaluation of visual object trackers on equirectangular panorama. In Proceedings of the International Conference on Computer Vision Theory and Applications, Funchal, Portugal, 27–29 January 2018.
5. Frome, A.; Cheung, G.; Abdulkader, A. Large-scale privacy protection in Google Street View. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 29 September–2 October 2009.
6. Balali, V.; Ashouri, A.; Golparvar, F. Detection, classification, and mapping of U.S. traffic signs using google street view images for roadway inventory management. *Vis. Eng.* **2015**, *3*, 3–15. [[CrossRef](#)]
7. Everingham, M.; VanGool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The Pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.* **2009**, *88*, 303–338. [[CrossRef](#)]
8. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Li, F.F. ImageNet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
9. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014.
10. Geiger, A.; Lenz, P.; Urtasun, R. Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3354–3361.
11. Dollár, P.; Wojek, C.; Schiele, B.; Perona, P. Pedestrian detection: An evaluation of the state of the art. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 743–761. [[CrossRef](#)] [[PubMed](#)]
12. Wen, L.; Du, D.; Cai, Z.; Lei, Z.; Chang, M.C.; Qi, H.; Lim, J.; Yang, M.H.; Lyu, S. UA-DETRAC: A New Benchmark and Protocol for Multi-Object Detection and Tracking. 2015. Available online: <https://arxiv.org/abs/1511.04136> (accessed on 4 September 2015).
13. Gerhard, N.; Tobias, O.; Samuel, R.B.; Peter, K. The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 4990–4999.
14. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 3213–3223.
15. Huang, X.Y.; Cheng, X.J.; Geng, Q.C.; Cao, B.B.; Zhou, D.F.; Wang, P.; Lin, Y.Q.; Yang, R.G. The ApolloScape Dataset for Autonomous Driving. CoRR. 2018. Available online: <https://arxiv.org/abs/1803.06184> (accessed on 26 September 2018).
16. Hazelhoff, L.; Creusen, I.; de With, P.H.N. Robust detection, classification and positioning of traffic signs from street-level panoramic images for inventory purposes. In Proceedings of the 2012 IEEE Workshop on the Applications of Computer Vision (WACV), Breckenridge, CO, USA, 9–11 January 2012; pp. 313–320.

17. Deng, F.; Zhu, X.; Ren, J. Object Detection on Panoramic Images Based on Deep Learning. In Proceedings of the 2017 3rd International Conference on Control, Automation and Robotics (ICCAR), Nagoya, Japan, 24–26 April 2017; pp. 375–380.
18. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. In Proceedings of the International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 91–99.
19. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
20. Dai, J.; Li, Y.; He, K.; Sun, J. R-FCN: Object detection via region based fully convolutional networks. In Proceedings of the Advances in Neural Information Processing Systems 29 (NIPS 2016), Barcelona, Spain, 5–10 December 2016; pp. 379–387.
21. Redmon, J.; Farhadi, A. Yolov3: An Incremental Improvement. 2018. Available online: <https://arxiv.org/abs/1804.02767> (accessed on 8 April 2018).
22. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; Springer: Cham, Switzerland, 2016; pp. 21–37.
23. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conferences on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
24. Hong, S.; Roh, B.; He, K.; Kim, Y.; Cheon; Park, M. Pvanet: Lightweight deep neural networks for real-time object detection. *arXiv* **2016**, arXiv:1611.08588.
25. Fu, C.Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional Single Shot Detector. *arXiv*, 2017; arXiv:1701.06659.
26. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the IEEE Computer Society, San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
27. Overett, G.; Petersson, L.; Brewer, N.; Andersson, L.; Pettersson, N. A new pedestrian dataset for supervised learning. In Proceedings of the IEEE Intelligent Vehicles Symposium, Eindhoven, The Netherlands, 4–6 June 2008; pp. 373–378.
28. Ess, A.; Leibe, B.; Gool, L.V. Depth and appearance for mobile scene analysis. In Proceedings of the IEEE 11th International Conference on Computer Vision, Rio de Janeiro, Brazil, 14–21 October 2007; pp. 1–8.
29. Gerónimo, D.; Sappa, A.; López, A.; Ponsa, D. Adaptive image sampling and windows classification for on-board pedestrian detection. In Proceedings of the International Conference on Computer Vision Systems, Bielefeld, Germany, 21–24 March 2007.
30. Leung, B. Component-Based Car Detection in Street Scene Images. Master’s Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, May 2004.
31. Dong, Z.; Wu, Y.; Pei, M.; Jia, Y. Vehicle type classification using a semisupervised convolutional neural network. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 2247–2256. [[CrossRef](#)]
32. Razavian, A.S.; Azizpour, H.; Sullivan, J.; Carlsson, S. CNN features off-the-shelf: An astounding baseline for recognition. In Proceedings of the IEEE Conferences on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 806–813.
33. Girshick, R.B.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conferences on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
34. Girshick, R.B. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Las Condes, Chile, 11–18 December 2015; pp. 1440–1448.
35. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conferences on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 936–944.
36. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.

37. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conferences on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
38. Sermanet, P.; Eigen, D.; Zhang, X.; Mathieu, M.; Fergus, R.; LeCun, Y. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv* **2013**, arXiv:1312.6229.
39. Redmon, J.; Farhadi, A. Yolo9000: Better, faster, stronger. In Proceedings of the IEEE Conferences on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
40. Jeong, J.; Park, H.; Kwak, N. Enhancement of SSD by concatenating feature maps for object detection. *arXiv* **2017**, arXiv:1705.09587.
41. Li, L.; Ouyang, W.; Wang, X.; Fieguth, P.; Chen, J.; Liu, X.; Pietikäinen, M. Deep Learning for Generic Object Detection: A Survey. *arXiv* **2018**, arXiv:1809.02165.
42. Uijlings, J.R.R.; van de Sande, K.E.A.; Gevers, T.; Smeulders, A.W.M. Selective Search for Object Recognition. *Int. J. Comput. Vis.* **2013**, *104*, 154–171. [[CrossRef](#)]
43. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
44. Shrivastava, A.; Gupta, A.; Girshick, R. Training region-based object detectors with online hard example mining. In Proceedings of the IEEE Conferences on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 761–769.



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

Article

# A New Dataset and Performance Evaluation of a Region-Based CNN for Urban Object Detection

Alex Dominguez-Sanchez, Miguel Cazorla \* and Sergio Orts-Escolano

RoViT, University of Alicante, Carretera San Vicente del Raspeig s/n 03690, San Vicente del Raspeig, Alicante 03690, Spain; alexdominguez09@yahoo.co.uk (A.D.-S.); sorts@ua.es (S.O.-E.)

\* Correspondence: miguel.cazorla@ua.es

Received: 13 September 2018; Accepted: 31 October 2018; Published: 6 November 2018

**Abstract:** In recent years, we have seen a large growth in the number of applications which use deep learning-based object detectors. Autonomous driving assistance systems (ADAS) are one of the areas where they have the most impact. This work presents a novel study evaluating a state-of-the-art technique for urban object detection and localization. In particular, we investigated the performance of the Faster R-CNN method to detect and localize urban objects in a variety of outdoor urban videos involving pedestrians, cars, bicycles and other objects moving in the scene (urban driving). We propose a new dataset that is used for benchmarking the accuracy of a real-time object detector (Faster R-CNN). Part of the data was collected using an HD camera mounted on a vehicle. Furthermore, some of the data is weakly annotated so it can be used for testing weakly supervised learning techniques. There already exist urban object datasets, but none of them include all the essential urban objects. We carried out extensive experiments demonstrating the effectiveness of the baseline approach. Additionally, we propose an R-CNN plus tracking technique to accelerate the process of real-time urban object detection.

**Keywords:** real-time object detection; autonomous driving assistance system; urban object detector; convolutional neural networks

---

## 1. Introduction

Recent advances in computer vision algorithms are paving the way for the development of future intelligent transportation systems: automatic traffic analysis, autonomous driving assistance systems (ADAS), autonomous navigation for unmanned aerial vehicles (UAV), etc. In recent years, convolutional neural networks (CNN) and other deep learning techniques have demonstrated impressive performance in many computer vision problems. Therefore, we believe they can be the perfect approach to these problems. A contributing factor to their success is the availability of differently scaled synthetic and real datasets (CamVid [1], CityScapes [2], Kitti [3], etc.).

Before the deep learning period, the detection and recognition of 2D objects were conducted using local features within the actual image to be analyzed. A classic method for detecting these images was based on Haar-like features, first used in [4].

Another common hand-crafted feature used for pedestrian detection is the histogram of oriented gradients (HOG) [5]. The idea behind this descriptor is that local object appearance and shape within an image can be described by the intensity distribution of gradients or edge directions. The image is divided into small, connected areas, and a histogram of gradient directions is generated for the pixels within each area. Finally, the descriptor is the concatenation of these histograms.

SIFT (scale-invariant feature transform) [6] is an algorithm used to extract local features within an image, which is also used for object detection. The ability to find distinctive key points that are invariant to location, scale and rotation made this method a good candidate for object detection approaches.

Another popular algorithm is SURF (speeded up robust features), presented in [7], which reduced the computation cost of SIFT by approximating Laplacian of Gaussian with kernels. One of the key advantages of this approximation is that performing convolutions with these kernels can be easily computed and can be done in parallel for different scales. It is worth mentioning that it is also possible to design ad hoc kernels for this application, and many works have studied how to design such kernels [8]. However, since the advent of deep learning networks, most traditional techniques have been surpassed by more accurate and efficient object detectors based on CNNs. We later present and evaluate some of the state-of-the-art techniques which use CNNs.

In this work, we present a novel study that evaluates a state-of-the-art technique for urban object localization. Furthermore, we propose a new dataset which has been created using existing data from other works and also new acquired data. The new acquired data is labeled using previously trained models, and we demonstrate that using this technique, which we name “auto updated learning”, we obtain better results, improving the rate of false negative detections.

More specifically, our contribution is as follows:

- A novel study that evaluates the proposed dataset and provides a baseline method for benchmarking urban object detection. Moreover, we evaluated additional state-of-the-art techniques, such as Faster R-CNN with ResNet101 and YOLOv2.
- A new, real, large-scale dataset for outdoor urban object detection and localization, which provides more than 200 k images with more than 600 k annotations. The dataset provides bounding-box-level annotations for the following classes: pedestrian, bus, bike, bicycle, car, traffic light and traffic sign. Currently, no dataset includes all these key urban objects, and more importantly, none of them additionally include 43 different types of traffic signals that we can find in most urban driving scenarios. The full dataset is available for download at <http://www.rovit.ua.es/dataset/traffic/>.
- A novel method to accelerate the object detection process and to enable the implementation of such a system in low-level GPU platforms. We achieve this by merging tracking techniques into the R-CNN detection algorithm.

The rest of the paper is organized as follows: Section 2 reviews related works on object detection and existing datasets (outdoor urban object detection). Section 3 describes the proposed dataset. In Sections 4 and 5, we present a baseline method and the results of its evaluation. Section 6 implements a mixed method using detection and tracking. Finally, Section 7 draws conclusions and presents future work.

## 2. Related Works

In 2012, the ImageNet competition started to change the methodologies used to recognize and detect 2D objects. Convolutional neural networks (CNNs) were implemented on GPUs, deploying deeper neural network models with increasing numbers of layers. Deep learning and CNN-based methods have become the state of the art in 2D-object detection in computer vision. They create a hierarchical representation with a larger order of abstraction from lower to higher layers of neurons. One of the first deep networks, created in [9], was a seven multi-layer net (AlexNet). This network has been shown to work well for image classification.

Subsequently, in 2014, an improved architecture was created [10]. The improvement of AlexNet replaced large kernel filters (11 and 5 in the first and second convolutional layers, respectively) with multiple  $3 \times 3$  kernels. It proved that multiple stacked smaller kernels work better than larger sized kernels because multiple nonlinear layers increase the depth of the network. This enables the learning of more complex features and reduces the computational cost.

In 2014, it was proposed a module for CNNs called inception [11], which was introduced in the GoogLeNet architecture. The idea behind this is that most of the activations in deep networks are unnecessary (value of zero) or redundant because they correlate with each other. Thus, most efficient



deep CNN architectures will have a sparse connection between the activations, which means that none of the 512 output channels will have a connection with any of the 512 input channels. Since only a small number of neurons within layers are effective, the width/number of the convolutional filters of a particular kernel size is kept small. It also uses convolutions of different sizes to capture details at multiple scales ( $5 \times 5$ ,  $3 \times 3$ ,  $1 \times 1$ ).

In 2015, it was developed an even deeper, simplistic and effective network based on the concept of residual networks (ResNet) [12]. The residual network creates a direct path between the input and output to this residual block entailing an identity mapping: The added layer just needs to learn the features on top of the already available input.

There are many publications related to object detection, but, in recent years, the topic has dramatically benefited from machine learning-based approaches. A novel method for locating 2D objects was described in [13]. It consisted of segmenting an image and then making a search strategy of the different segmented regions. The study generated all possible object locations in an exhaustive search based on groupings by color spaces and groupings based on the features of the object, such as texture, size and shape (selective search).

Another work, presented in [14], developed an object detection system based on mixtures of multi-scale deformable part models. Their system was able to represent variable object classes with good results as early as 2009. It was based on deformable part models and methods to discriminate training with partially labeled data. It was combined with an approach using a fine-tuned support vector machine (SVM).

In 2012, a work was presented measuring the objectness of an image [15]; in other words, the probability of an image region enclosing a defined object of any class. These authors trained their system to differentiate objects based on defined boundaries, differentiating, for example, cows from a background such as grass. In a Bayesian framework, the measure combines several image cues measuring characteristics of objects, such as appearing different from their surroundings and having a closed boundary, the density of edges, color contrast and what they call multi-scale saliency based on a spectral residual of the fast Fourier transform (FFT), which favors areas with a unique appearance within the entire image.

After the CNN revolution in 2012, another paper was presented with an integrated method for using CNNs for classification, localization and detection [16]. This study proved how a multi-scale and sliding window approach can be implemented within the same ConvNet. The authors used the same structure as in [9], but for the three tasks simultaneously. They also introduced a deep learning approach to localization by learning to predict bounding boxes, which are then accumulated in order to increase detection scores.

R-CNNs have been used in recent years as the best object localization CNNs [17]. Fast R-CNN proposed a single-stage training method that learns to detect and classify object proposals, providing their bounding boxes [18]. The latter improves speed and accuracy compared to the former by sharing the computation of the convolutional layers between different proposals, and swapping the order for generating region proposals. Subsequently, Faster R-CNN was presented to combat the complex training pipeline of both R-CNN and Fast R-CNN [19]. This system added a region proposal network (RPN) for learning to predict regions that contain objects. It decreased the complexity of the training process compared to Fast R-CNN.

In 2016, a novel approach for object detection was presented [20]. The authors addressed object detection as a regression problem. One neural network predicts bounding boxes and class probabilities directly from color images in just a single inference. YOLO divides the input image into  $S \times S$  blocks, and then predicts the score for each box for every object class in training. As the whole detection process uses a single network, it can be optimized end-to-end, improving training/inference speed.

Another state-of-the-art method is single shot detector (SSD) [21], which presents a good balance between speed and accuracy. SSD runs a CNN on an input image only once and works out a feature map. SSD also uses bounding boxes at various aspect ratios (scales), similar to Faster R-CNN, and

learns the offset rather than learning the bounding box. In order to handle the scale, SSD predicts bounding boxes after multiple convolutional layers. Since each convolutional layer operates at a different scale, it is able to detect objects of various scales.

Currently, there are various datasets related to traffic environments. For instance, SYNTHIA –a SYNTHetic collection of Imagery and Annotations– is a traffic dataset rendered in 3D to add semantic segmentation and different environments to understand problems in the context of several driving scenarios [22]. It is composed of more than 200,000 high-resolution images and also simulates a 360° view with eight simulated cameras. People, cars and bicycles are moving objects in the scene.

Another traffic simulator is CARLA [23]. It is an open-source simulator for autonomous driving research and can be used to extract rendered images for training and building a traffic-oriented dataset.

KITTI [3] is a real-image dataset that includes objects, such as cars, vans, trucks, pedestrians and cyclists, captured using a variety of sensor modalities, including GPS. The KITTI dataset has been used in many challenges, such as depth from stereo, object tracking, 3D object localization and flow estimation. There exist two different versions, one released in 2012 and an extension of some of its challenges (new data), which was released in 2015.

Mighty AI is another example of a real traffic-based dataset, acquired from a car. In this case, the segmentation of objects (pixel-level), such as cars, buses, trucks and lane marks, is provided.

A similar dataset to KITTI is the Oxford Robotcar [24], which includes LIDAR, GPS and inertia data for over 20 million images captured in an urban environment.

Many datasets have been created, offering pixel-level annotations for roughly the same objects. Another example is the CityScapes dataset [2], which has 25,000 semantic annotated images of the most common objects found in an urban scene, including buildings, roads, trees and even sky.

It is interesting that most of these datasets do not provide traffic sign data. Traffic signs are one of the main objects in autonomous driving and the need to identify them is as important as the localization of other cars or traffic objects in the immediate area. However, a number of isolated traffic sign datasets have been created to fill this gap.

Examples of existing traffic sign datasets include the German Traffic Sign Recognition Benchmark [25] (GTSRB), which includes 43 classes and around 40,000 images. There is also the Traffic dataset from Linköping University [26] (Sweden), which has around 4000 annotated signs and 22 classes. LISA [27] (USA) is a further traffic sign dataset with 47 classes and about 8000 images. There is also a traffic sign dataset from the University of Alcalá de Henares [28] (Spain), which has subdivided images according to their shape and color information.

### 3. Dataset Description

In this work, we created a novel dataset that gathers images from existing real traffic datasets and new acquired data (with an auto-updated annotation). It has been divided into two groups: traffic objects and traffic signs. The first is a group of images and annotations of traffic objects (2D bounding boxes), specifically, seven classes: car, motorbike, person, traffic light, bus, bicycle and traffic sign (see Figure 1).

The second group contains 43 different traffic signs (classes) which are most commonly found on European roads. The annotation data includes the class of the object and the bounding box coordinates.

The dataset is a compendium of different publicly available datasets, such as PASCAL VOC [29] and Udacity [30], and the remaining data was acquired and annotated from real-life images. From PASCAL VOC, we took only the classes of bicycle, bus, car, motorbike and person, which encompassed 22% of the total dataset. Udacity provided 65% with the classes of bicycle, car, person and traffic light. Then, as we needed to complete the dataset with more objects, we added more images using buses and motorbikes from Internet videos, and bicycles from videos recorded in urban environments and on roads in Alicante (Spain). Finally, we added a small set of traffic lights from our own capture video that accounts for no more than 1%. See Figure 2 for the distribution of

images from the different datasets, and refer to Table 1 for a later dataset distribution used during the experimentation phase. See Figure 3 for the initial class distribution.

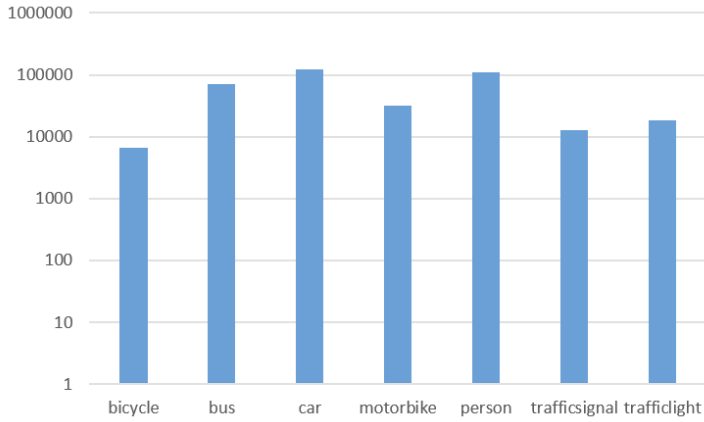


Figure 1. Number of annotations per class.

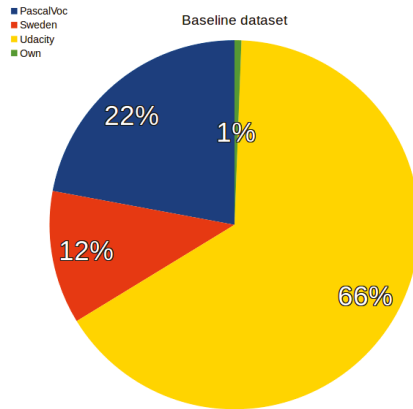


Figure 2. Percentage distribution of the baseline dataset.

Table 1. All mAPs per individual class across all experiments.

AP	Initial	AutoUpdated	YOLOv2	Faster R-CNN ResNet101
Original Datasets	99%	34%	34%	34%
AP bicycle	0.419	0.618	0.466	0.613
AP bus	0.890	0.980	0.904	0.962
AP car	0.670	0.724	0.636	0.703
AP motorbike	0.592	0.730	0.665	0.715
AP person	0.649	0.760	0.603	0.707
AP traffic light	0.536	0.472	0.346	0.481
AP traffic sign	0.819	0.774	0.719	0.692
<b>Mean AP</b>	<b>0.713</b>	<b>0.742</b>	<b>0.620</b>	<b>0.696</b>



**Figure 3.** Image samples from the proposed dataset (seven classes).

All the traffic signs found in the dataset (more than 12,000 labeled objects) were annotated within one single class: traffic signal. As explained later, a second dataset containing only traffic signs was created using two publicly available datasets: the GTSRB from the Institut für Neuroinformatik (Germany) [31] and another from Linköping University (Sweden) [26]. The German dataset contains around 40,000 traffic signs divided unevenly into 43 classes, while the Swedish one has 22 classes and around 6000 traffic signs, all annotated.

The dataset is imbalanced. For the experimentation, we balanced the different classes using data augmentation (rotations, zoom, affine transformation, blurring, etc.) and reduced the three classes with the most elements (car, bus and person).

## 4. 2D Object Detection and Recognition

### 4.1. Baseline Method

The main goal of this work was to design a reliable system able to detect the main objects found in a driving situation in any urban or motorway environment. For this purpose, we chose to use a state-of-the-art CNN network to detect the main seven objects within driving environments: cars, motorbikes, people, traffic lights, buses, bicycles and traffic signs.

In order to achieve this, we needed to rely on a robust detection and classification system. It is necessary not just to be able to classify the objects but also to locate them within the scene. A region proposal method was used for this task. Specifically, we built a dataset based on [19] and their Faster R-CNN work, previously described in Section 2.

We fine-tuned Faster R-CNN with VGG16 convolutional blocks for this purpose, modifying it to detect eight classes: seven common traffic objects and the background of the scene. Once all the 2D objects were located and identified, it was still necessary to classify all the traffic signs in order for the system to have a robust knowledge of the traffic scene at any given time. Since we grouped all the traffic signs together in a single class, we performed a second step using a fine-tuned CNN to classify the detected traffic signs (43 subclasses). We could have included every single traffic sign as a new class across the rest of the objects, but, as a traffic sign is a class per se, and there are many different types of signs to classify, it seemed obvious and semantically organized to separate them into a different dataset. In practical terms, if we needed to include more types of traffic signs, it would not be necessary to retrain the whole model, but only the traffic sign model with the new traffic sign dataset. The same would apply if we needed to classify different types of cars, brands or models, types of motorbikes, etc.

### 4.2. Training

The Faster R-CNN was trained using an existing model (PASCAL VOC) based on [19]. This provided several advantages in certain classes. For instance, the person class, including PASCAL VOC, offered generalization capabilities provided by this general-purpose dataset. The PASCAL VOC dataset includes many people from a variety of angles and sizes, and in different light conditions

and poses. In a traffic scene, 99% of persons are normally riding a bicycle or a motorbike or walking alongside a road. Hence, we added this type of object from our recordings to the dataset. The same applies to bicycles and motorbikes. Thus, we also added those objects from our recordings and from the Internet.

The UDacity dataset provided even greater robustness when detecting cars, as all the recordings were acquired from a car's perspective. UDacity also provides the kind of images and objects one would find in a driving situation: cars and other vehicles from the driving vehicle perspective and angle. This increased the accuracy when detecting cars in our model.

Subsequently, around 375,000 annotated objects from 106,920 images were used for training. The train/validation/test split sizes were 40% for training, 40% for validation and 20% for testing purposes. However, we also evaluated other configurations, such as 60%, 20%, 20% and 50%, 25%, 25%, but the most accurate and robust results were obtained using 40%, 40%, 20%.

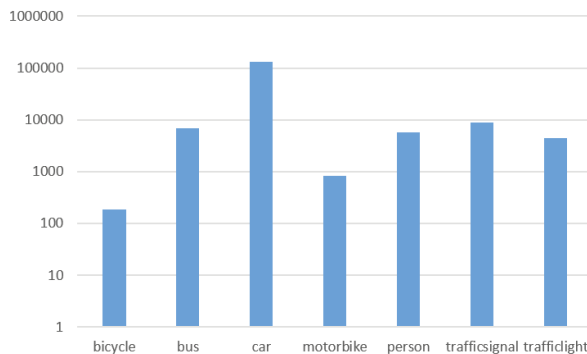
We trained the approach for 80,000 iterations, using a learning rate starting at 0.001 and decreasing it 10 times, after every third of the training process. Momentum was set to 0.9 and weight decay to 0.0005.

#### 4.3. Auto-Updated Learning

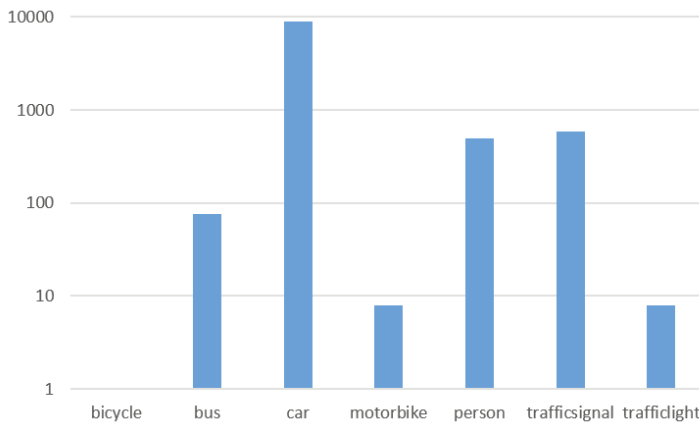
Once we had a stable and accurate system—let us call it the baseline model—we wanted to test how adding new images increases the accuracy. We call this method the auto-updated learning process. In order to test the auto-updated learning, we composed two automatically annotated datasets with two high-speed cameras: one at high resolution and one at low resolution.

We recorded 7 h 13 m 40 s of  $1280 \times 720$  video at 60 fps in 23 different situations (urban, countryside and motorways). We then converted the video to a lower frame rate. In this case, 10 fps proved to be a good balance between a reasonable number of frames to train without causing many repetitions due to the similarity of sequentially recorded frames.

Recorded sequences were automatically annotated using the baseline model. Once we had this new automatic annotated dataset (see Figure 4), we added it to the baseline dataset and trained the same architecture used for the baseline model (Faster R-CNN with VGG16 model) again to see how the new model could improve. The baseline model with the new dataset showed an improvement in detecting some of the objects for which it was trained. In a way, the baseline model was learning new features by being trained with new images annotated using the baseline model (see Table 1). This process was repeated using a different set of images (see Figure 5) captured with the low-resolution camera, which was able to record at 60 fps but at VGA resolution.



**Figure 4.** Number of annotations per class from the first set of automatic annotated images. There were 157,406 annotated objects captured with an HD sport camera with a  $1280 \times 720$  resolution.

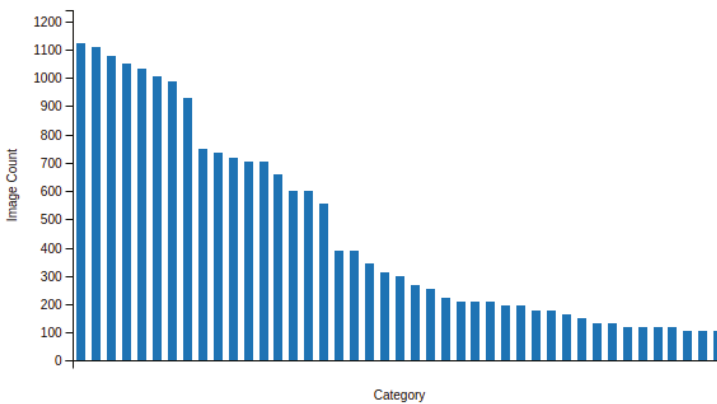


**Figure 5.** Number of annotations per class from the second set of automatic annotated images. There were 10,167 annotated objects captured with a low-resolution SONY camera with a  $640 \times 480$  resolution. Note there is only 1 annotation for the bicycle class.

4.4. Traffic Sign Recognition

As previously mentioned, one of the output classes of our trained network is ‘traffic sign’. The CNN network was trained using thousands of images. In order not to just detect but also to classify (traffic sign type) them, we fine-tuned (ImageNet weights) a ResNet50 architecture but now added 43 classes, with these being the most commonly used traffic signs in the European Union. ResNet50 was chosen as one of the latest architectures in the state of the art. To test the model, we used the GTSRB dataset [31], which also has 43 object classes.

Figure 6 shows the number of images for each traffic sign type (GTSRB dataset). We trained a ResNet50 network using this dataset. The images are square, with sizes that range from 40 to 170 pixels. Figure 7 shows some traffic sign bounding boxes extracted from the Traffic Signs Dataset [26]. For training purposes, we implemented data augmentation to compensate categories with a lower number of elements.



**Figure 6.** Number of images per traffic sign in the GTSRB dataset.



Figure 7. Image samples from the Traffic Signs Dataset [26].

## 5. Experiments

### 5.1. Setup

We primarily used GPUs from NVIDIA (Titan Xp, GTX 1070 and Quadro P6000) for training the proposed system. At inference time, we also used an NVIDIA GTX 1060.

One of the recording devices we used for acquiring new data was an HD (1280 × 720 resolution) sport camera (H5 Midland) mounted on the front of a vehicle. This camera is able to record video at 60 fps and has a CMOS of 5M pixels and a wide-angle lens of 170°. Moreover, we used a second camera model, the SONY Playstation Eye, which is able to record video at 60 fps (640 × 480 resolution). It uses a VGA CMOS and a wide angle lens of 75°.

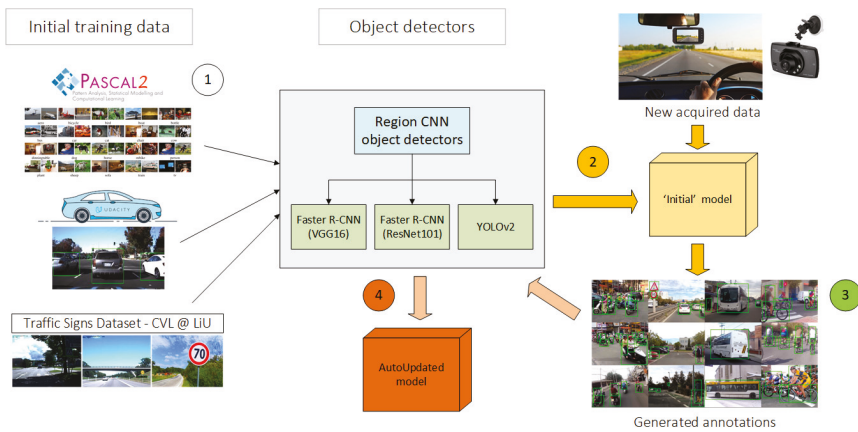
A Linux-based system was used along with the Python implementation of the Faster R-CNN [19], which uses the Caffe Deep Learning framework. Many scripts were modified for our experiments. We fine-tuned the model according to the classes used for the experimentation phase.

We also used the NVIDIA DIGITS 5.1, running the Caffe fork. Finally, for the classification experiments (second step: traffic sign classification), we used the Caffe version of the ResNet50 architecture.

### 5.2. Performance Evaluation: Urban Object Detection

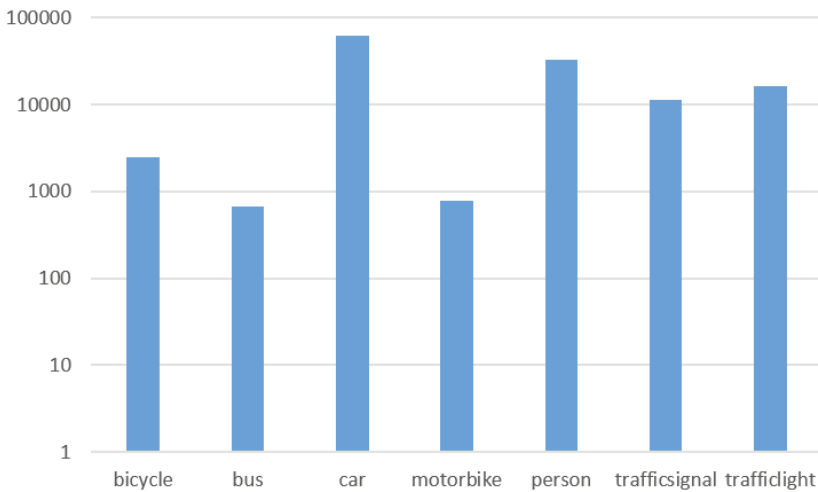
In this section, we describe all the experiments performed in this work. As a measure of accuracy, we use the mean average precision (mAP), as used in [19]. We split the dataset into 40% for training, 40% for validation and 20% for testing. Figure 8 shows an overview of the steps we followed to perform the evaluation of the baseline object detector. It also describes how we trained the 'Initial' and 'AutoUpdated' versions of the urban object detector model. In step 1, we compiled an initial dataset for urban object detection by filtering particular classes of existing datasets. In step 2, we trained a Faster R-CNN network (VGG16) using the initial version of the dataset. For step 3, using the trained model and new acquired data on real driving scenarios, we automatically annotated new images by using the initial model. In step 4, finally, we retrained the baseline network and other state-of-the-art CNN-based networks (ResNet101 & YOLOv2) using the final version of the dataset (compilation of existing datasets and new recorded data that was automatically annotated).

We ended up with a total of 166,139 objects (Figure 9). For this experiment and the subsequent ones, we used the Faster R-CNN architecture, using VGG16 convolutional blocks [18], which is more computationally costly (slower training/inference than YOLOv2) but provides greater accuracy.



**Figure 8.** Overview of the process for the training of the 'Initial' and 'AutoUpdated' versions of the object detector model.

We noticed that the mAP for the traffic light class was not as accurate as with the other classes, even using 10% of the total dataset (16,564 traffic lights, see Figure 9). Many of the bounding boxes for traffic lights were very small (fewer than 240 squared pixels), so we decided to filter out the very small ones, as they would provide no practical information in a driving situation because, with the bounding box being so small, the traffic lights would be extremely far away from the vehicle.



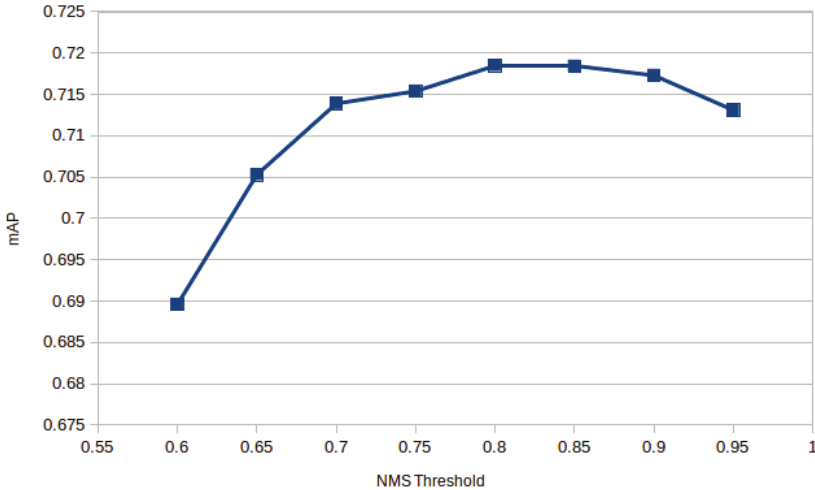
**Figure 9.** Final number of annotations per class included in the proposed dataset.

In [19], the author used a non-maximum suppression (NMS) threshold for the removal of candidate regions, setting a fixed threshold of 0.713. We tried several thresholds to select the correct one. The best NMS threshold was 0.8 (see Figure 10). We trained the baseline model for 70,000 iterations, starting with a learning rate of 0.001. The results are shown in Table 1, Baseline method.

We observed that the detection accuracy for bicycles and motorbikes was not as high as for other classes, such as cars or people. Furthermore, the distribution of bicycle and motorbike objects in the dataset was also too low in comparison with the other classes. Buses were also too low, and many



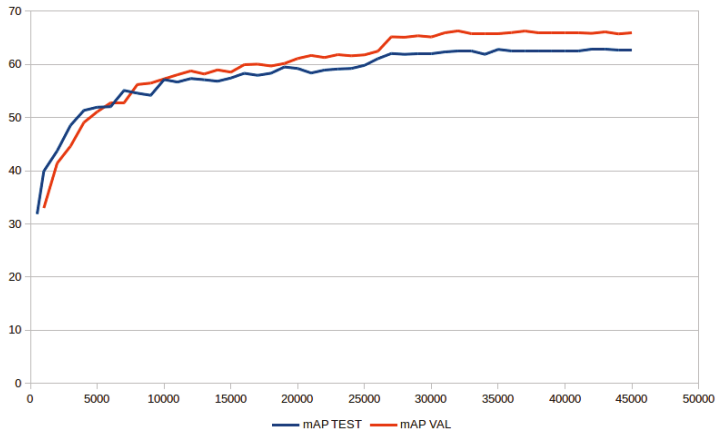
buses in real-life tests were not detected. So, we decided to add more of these objects (buses, bicycles and motorbikes) to the dataset, again using an auto-updated technique. We annotated 7 h of recordings (10 fps  $1280 \times 720$  videos, recorded with the H5 camera). From those labeled images, we increased the number of buses, bicycles and motorbikes (and the other classes such as cars, people, traffic signs were also shown in the new images).



**Figure 10.** Mean average precision (mAP) values for different non-maximum suppression (NMS) thresholds.

We then trained for 80,000 iterations. We implemented data augmentation as explained before, thus ensuring all the classes were evenly included in the training and validation set, achieving a mAP of 0.7420. Table 1 shows the results for the different R-CNN architectures that we evaluated. The first column shows the mean AP score for the baseline method (Faster R-CNN with VGG) using the ‘initial’ dataset (compilation of images from existing datasets). The second column, ‘AutoUpdated’, shows the mean AP score of the baseline method trained using the final version of the dataset (compilation of initial images and new recorded sequences). The new recorded sequences were annotated by using the initial model. Finally, Table 1 shows the mean AP score for the YOLOv2 and Faster R-CNN with ResNet101 architectures. Both of these networks were trained using the final version of the proposed dataset (AutoUpdated). As we can see, using ResNet101 convolutional blocks for object detection produces slightly worse results compared to using VGG blocks. Moreover, we show how YOLOv2 produces a lower mean AP score compared to the Faster R-CNN methods; however, YOLOv2 has also a lower runtime, which makes it suitable for real-time systems.

As the goal of this work is to deploy the developed system in an embedded system (like a Jetson card or FPGA), we need a fast implementation of the object detection component. Therefore, we evaluated the Darknet version of YOLOv2 [20] in our seven-class dataset. We applied a learning rate of 0.0001 and trained for 45,000 iterations, with a momentum of 0.9 and decay of 0.0005. We also achieved good results with our dataset, achieving a mAP of 0.62 in the test set. After approximately the 27,000th iteration, the mAP stabilized, achieving no further increase in accuracy (see mAP evolution in Figure 11). See Table 1, YOLOv2, for results.



**Figure 11.** mAP for test and validation sets in YOLOV2 for 45,000 iterations.

### 5.3. Additional Test, Processing a Mix of Real-Life Clips Acquired in Various Urban Environments

After observing the mAP results within the previous experiments, we wanted to test whether our latest model could improve in terms of detection accuracy even though the mAP slightly decreased due to the lack of balance in terms of the number of annotations per class.

We recorded real-life videos of scenes containing many of the urban objects defined in the classes of the proposed dataset. The actual video has 8992 frames, and was mainly recorded in very busy urban environments (see Figure 12). The video does not represent a ground truth of detected objects, but it gave us information on how much better (in terms of number of detected objects) one CNN model is with respect to the others. It also provides a visual confirmation of the mAP coefficients measured earlier. After detecting all the objects in the scenes and computing various statistics for the processed video, we concluded that the auto-updated training strategy improved the accuracy when detecting objects (see Table 2). The accuracy was not greater in terms of mAP, but the number of correct bounding boxes that were predicted increased compared to previous experiments. Moreover, in some cases, the improved model was able to detect bounding boxes that were not previously detected (decreased the number of false negative detections).



**Figure 12.** Image samples from the real-life video (test split).

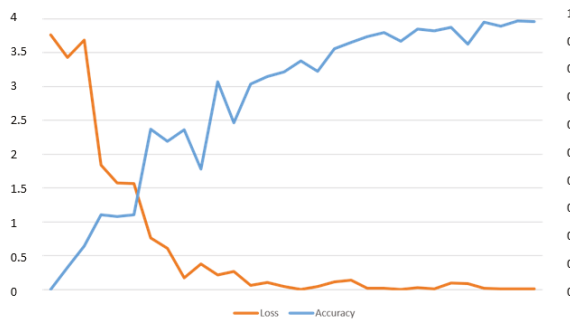
**Table 2.** Number of objects detected in real-life video across all experiments.

Model	Cars	Motorbikes	Person	Traffic Lights	Buses	Bicycles	Traffic Signs	mAP
Initial	8907	1044	19,684	1628	465	346	562	0.713
AutoUpdated	10,485	4013	21,739	1280	1260	1406	1030	0.74
ResNet101	10,188	4231	20,711	1178	1319	1301	1479	0.696
YOLOv2	10,552	3937	22,357	1006	1708	843	1250	0.620

#### 5.4. Traffic Sign Classification

In the last years, CNNs have surpassed most traditional methods, achieving a 99% classification accuracy on publicly available datasets for traffic sign recognition [32]. In order to complete the creation of a more complete urban object localization system, we focused on the traffic sign classification task. We needed to be able not just to detect a traffic sign, but also to classify it. We chose a ResNet50 architecture for this task due to its proven speed and accuracy for classification tasks [16,33]. In this work, we fine-tuned the original model, which was trained using the ImageNet dataset, using the German traffic dataset (GTSRB) for evaluation purposes. The GTSRB dataset contains 39,495 images of 43 different traffic signs (see Figures 6 and 7). Additionally, we created a new Spanish traffic sign dataset with 3300 images divided into 45 traffic signs. This was acquired from the previously presented dataset (Section 3) by automatically cropping the resulting detected traffic signs using our previous Faster R-CNN baseline model.

We trained for 16,000 iterations, using a learning rate starting at 0.005 with a polynomial decay gamma value of 0.1 and weight decay of 0.0005. The dataset was split as follows: 50% for training, 25% for validation and 25% for testing. It obtained 98.09% accuracy on the test set (choosing the class with the greatest probability) and 99.76% accuracy when performing a vote using the top five predicted classes. As a final test, we added 100% of the Spanish dataset to the GTSRB and applied the same distribution. Testing against the ground truth resulted in 99.7% accuracy on Top1 and 99.9% on Top5 (see accuracy and loss in Figure 13).

**Figure 13.** ResNet50 training accuracy and loss for the traffic sign classification task.

#### 5.5. Discussion

While creating this dataset, we discovered that the amount and variety of data used for training a CNN are crucial for the accuracy of our application to unseen data. As we added more data from different sources and domains, with different scenarios and backgrounds, we saw that we were able to decrease the generalization error (unseen data). However, in some of the experiments, we also realized that, due to the increase in the diversity of our training data sources, the system was performing worse on certain classes due to imbalanced data. To address this problem, we had to capture new data focusing on creating new annotations for imbalanced classes. After training a new model considering

these two factors, we were able to increase the accuracy of the trained model and also reduce its generalization error.

## 6. Detection and Tracking

In this section, we explain how tracking objects in a video feed can benefit from the detection process of a region CNN.

### 6.1. Motivation

The main goal of an urban R-CNN model would be to implement such a system in a real-life vehicle, in a real traffic environment. As we know, such systems require great GPU performance; most (probably small) embedded systems would be more realistic final hardware to implement such a network, so a good way to accelerate the detection process using software is needed. Consequently, we tracked the detected object in certain ways in order to alleviate the GPU for such work.

### 6.2. Strategy

The strategy to follow in the detection-tracking mix would be to first detect the objects with an R-CNN, which should occur quickly enough to work in small hardware, then track the objects either until they are lost or until the reliability of the tracker is found to be diminished and there is insufficient confidence in the localization of the object detected in the first place.

We analyzed several tracking algorithms: Correlation tracker [34], Deep sort [35], Boosting [36], Mil [37], KCF [38], TLD [39], MedianFlow [40] and Mosse [41].

All those algorithms were evaluated in three different GPUs from an NVIDIA 940M, a 1050 Ti and a 1080. We tested them for speed, accuracy and robustness. Moreover, we studied the point at which it is best to relaunch our R-CNN detector to detect the objects again. In some cases, launching the R-CNN every  $n$  frames was a good solution; in others, launching as soon as the tracker lost the object also achieved good results. Hence, a balance of these two strategies proved to be the best option for real-time processing given a limited computing budget.

Some algorithms are good for relocating the tracked object after a partial occlusion, but, as this is not strictly relevant to our purpose, we did not focus on this feature.

Additionally, the goal of the detection-tracking bundle is to prove that hungry GPU processes, such as R-CNNs, can benefit from tracking techniques, not to present an exhaustive study about tracking algorithms.

We chose three of the nine algorithms tested: KCF, MedianFlow and Mosse. These were found to be quite fast with reasonably good accuracy in urban environments.

Tracking objects in a traffic environment from a moving car has a particular drawback related to tracking objects in the outside edges of the camera view, such as parked cars, pedestrians or overtaken vehicles such as cars or buses. These objects can very rapidly increment their size and location in the scene, and most trackers do not perform well with such large and fast variations. However, objects moving beside or in front of the car can be tracked for a long time. An example of this can be seen in Figure 14.



Figure 14. Objects changing size rapidly after a few frames. Red bus vs. blue bus.

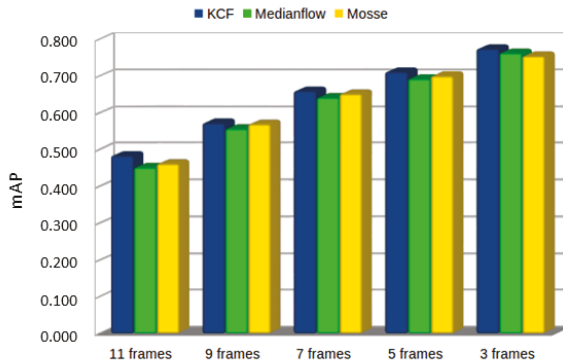
6.3. Experiment: Detecting with YOLOv2 Every  $n$  Frames.

Although Faster R-CNN provided better mAP than YOLOv2, we chose the latter for our experiments. This is due to YOLOv2 having a shorter execution time, and this system is intended to be embedded in devices such as the Jetson TX2 from NVIDIA.

The first experiment was based on detecting objects in an urban environment video with  $1280 \times 738$  pixels in RGB format at 30 fps. With our seven-class urban dataset, we used YOLOv2 to detect objects (cars, buses, pedestrians, etc.) and let the Mosse tracker follow them for  $n$  frames. As can be seen in Table 3, the longer the period to relaunch YOLOv2, the faster the algorithm. The goal of this experiment was to prove that YOLOv2 detection would benefit from intermediate tracking and to determine how long a tracker was a beneficial option (Figure 15).

**Table 3.** YOLOv2 + Mosse tracker performance every  $n$  frames in an urban environment video. YOLO detection is performed every  $n$  frames, starting with  $n = 30$  and ending with  $n = 1$  (tracking disabled). We can see how, as we perform tracking for a larger number of frames, for example  $n = 25$ , the overall performance increases, with it being able to process video at 27.8 fps.

MOSSE Urban (8117 Frames)	30 Frames	25 Frames	20 Frames	15 Frames	10 Frames	5 Frames	1 Frame
GPU 1080	29.0 fps/2381	27.8 fps/2599	27.1 fps/2810	25.0 fps/3235	22.6 fps/3734	18.6 fps/4900	17.4 fps/8117
GPU YOLOv2 usage	29.3%	32.0%	34.4%	39.9%	46.0%	60.4%	100%



**Figure 15.** mAP for KCF, Median flow and Mosse tracking algorithms every  $n$  frames.

6.4. Experiment: Urban and Motorway Environments

For these tracking algorithms, we needed to know how much the tracker would help the YOLOv2 R-CNN detection.

We set two traffic videos of roughly the same length, about 4 min 20 s: one in a busy urban environment and the other in a motorway environment. Videos were  $1280 \times 738$  at 30 fps. We launched YOLOv2 just every 30 frames, and in the rest of the frames, the tracker was in charge of locating the objects. We measured the average frames per second, comparing the video using just YOLOv2 with the same video using no tracking help at all. We also measured the percentage (based on number of frames) where YOLOv2 was used. We tested the three tracking algorithms, the results of which are presented in Table 4. Note that frames with no objects in the scene were not taken into account for the percentage results.

It can be seen that the use of GPU (YOLOv2 usage) can be dramatically reduced by between 10% and 14% when using a tracking algorithm, accelerating the calculation process as a whole. Consequently, the average speed (in fps) can be increased.

**Table 4.** Comparison between traffic environments: YOLOv2 and YOLOv2 + Tracking algorithms.

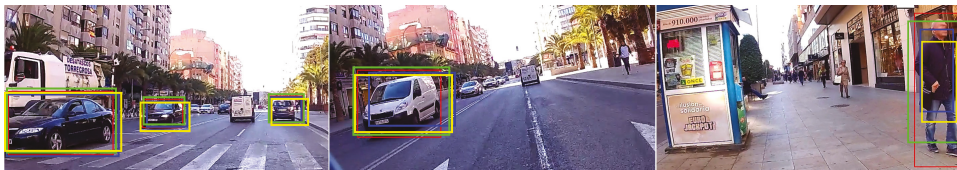
GPU	Just YOLOv2	KCF Urban	KCF Motorway	MEDIANFLOW Urban	MEDIANFLOW Motorway	MOSSE Urban	MOSSE Motorway
940M	2 fps	4.9 fps	2.3 fps	5.8 fps	2.3 fps	5.9 fps	2.3 fps
1050 Ti	9.4 fps	13.5 fps	10.4 fps	21.2 fps	11.2 fps	21.1 fps	11.4 fps
1080	17.4 fps	18.6 fps	17.3 fps	29.2 fps	19.7 fps	29.0 fps	20.3 fps
GPU YOLOv2 ussage	100%	10.8%	13.7%	10.3%	12.7%	12.0%	13.3%

6.5. Experiment: mAP Accuracy in Trackers

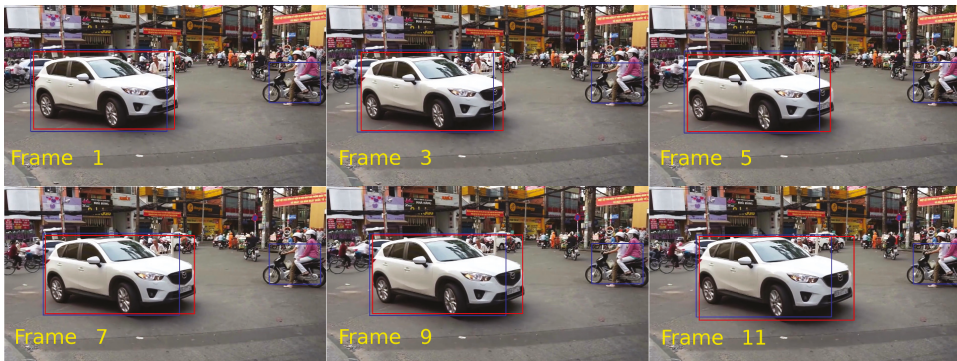
Finally, we tested how accurate these tracking algorithms were and for how long the tracking algorithm could keep track of the detected object, with the tracked object being sufficiently reliable or as reliable as the YOLOv2 detection.

We set the ground truth of all bounding boxes detected by YOLOv2 in every single frame (8117 frames). We then let the tracking algorithm locate these objects for n frames and compared the bounding boxes produced by the tracking algorithm with the YOLOv2 ground truth in order to have a mAP. As can be seen in Figure 15, the tracking location is reasonably accurate until the 10th frame but, after that, the mAP is less than 0.5. It is also noticeable that KCF is slightly more accurate than the other two trackers. Figure 16 shows examples of how accurate the trackers are in relation to the number of frames between YOLOv2 detections.

Figure 17 shows that the tracking of detected bounding boxes differed between YOLOv2 detection and that, after the 11th, this can be noted visually.



**Figure 16.** Examples of tracked objects related to YOLOv2 detection, skipping 3, 7 and 11 frames. Ground truth (YOLOv2) in red. KCF in blue, Medianflow in green and Mosee in yellow.



**Figure 17.** Example of tracking using KCF compared with YOLOv2 detection. Sequence of 11 frames. YOLOv2 in red, KCF in blue. At the front of the car, it can be seen that the longer the tracking period, the less accurate the result, with the KCF bounding box increasingly differing from the YOLOv2 bounding box.

We can conclude that tracking definitely accelerates the detection process as a whole. In Table 4, the increase in speed obtained in this experiment by applying any tracking algorithm can be seen.

For instance, in an urban environment, using a reasonably low GPU, such as an NVIDIA 1050 TI, we go from 9.4 fps using only YOLOv2 to 21.2 fps by adding a MedianFlow tracker, which represents an increase of over 100% in calculation speed.

## 7. Conclusions

In this work, we present a new dataset for urban object localization, which is a compilation of new and existing data. It includes annotations not just for common moving objects, such as cars, pedestrians, bicycles, etc., but also for static traffic signs and traffic lights.

We trained a state-of-the-art Region-based CNN, Faster R-CNN architecture, and proved how well this dataset can be used for real-life traffic situations, such as urban and motorway scenarios, achieving a mean average precision accuracy of 0.74. Moreover, by retraining the proposed architecture following an auto-updated strategy, we demonstrated how the proposed R-CNN network improved its accuracy (fewer number of false negatives) in real-life driving situations. Additionally, we evaluated other state-of-the-art detectors, such as Faster R-CNN with ResNet101 convolutional blocks and YOLOv2. From these experiments, we concluded that Faster R-CNN with ResNet101 performed very similar to the proposed baseline, and YOLOv2 achieved lower mean AP scores but it considerably improved overall performance (runtime).

We also proved that tracking techniques can noticeably improve R-CNN object detection (YOLOv2) and its efficiency by balancing the use of both, detection and tracking, without losing accuracy.

As a future work, we plan to merge the existing traffic sign datasets used in this project. We plan to repeat the same process followed in this study and extend these datasets using an auto-updated strategy.

**Author Contributions:** Conceptualization, A.D.-S.; Methodology, A.D.-S., M.C. and S.O.-E.; Validation, A.D.-S.; Investigation, A.D.-S., M.C. and S.O.-E.; Writing-Original Draft Preparation, A.D.-S., M.C. and S.O.-E.; Writing-Review & Editing, M.C. and S.O.-E.

**Funding:** This work has been partially funded by the Spanish Government TIN2016-76515-R grant for the COMBAHO project, supported with Feder funds. It has also been supported by the University of Alicante project GRE16-19. Experiments were made possible by a generous hardware donation from NVIDIA.

**Conflicts of Interest:** The authors declare no conflict of interest. The funding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

## References

1. Brostow, G.J.; Fauqueur, J.; Cipolla, R. Semantic object classes in video: A high-definition ground truth database. *Pattern Recogn. Lett.* **2009**, *30*, 88–97. [CrossRef]
2. Cordts, M.; Omran, M.; Ramos, S.; Enzweiler, M.; Benenson, R.; Scharwächter, T.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset. 2015. Available online: [https://www.visinf.tu-darmstadt.de/media/visinf/vi\\_papers/2015/cordts-cvprws.pdf](https://www.visinf.tu-darmstadt.de/media/visinf/vi_papers/2015/cordts-cvprws.pdf) (accessed on 5 November 2018).
3. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
4. Viola, P.A.; Jones, M.J. Robust Real-Time Face Detection. *Int. J. Comput. Vis.* **2004**, *57*, 137–154. [CrossRef]
5. Dalal, B.T.N. Histograms of Oriented Gradients for Human Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, CA, USA, 21–23 September 2005.
6. Lowe, D. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
7. Bay, H.; Tuytelaars, T.; Van Gool, L. *SURF: Speeded up Robust Features*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 404–417.
8. Zorzi, M.; Chiuso, A. The Harmonic Analysis of Kernel Functions. *Automatica* **2018**, *94*, 125–137. [CrossRef]
9. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*; Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2012; pp. 1097–1105.

10. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
11. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. *arXiv* **2014**, arXiv:1409.4842v1.
12. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016.
13. Uijlings, J.R.R.; Van De Sande, K.E.A.; Gevers, T.; Smeulders, A.W.M. Selective search for object recognition. *Int. J. Comput. Vis.* **2013**, *104*, 154–171. [[CrossRef](#)]
14. Felzenszwalb, P.F.; Girshick, R.B.; McAllester, D.; Ramanan, D. Object Detection with Discriminatively Trained Part Based Models. *IEEE Trans. Pattern Anal. Mach. Intell.* **2009**, *32*, 1–20. [[CrossRef](#)] [[PubMed](#)]
15. Alexe, B.; Deselaers, T.; Ferrari, V. Measuring the objectness of image windows. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 2189–2202. [[CrossRef](#)] [[PubMed](#)]
16. Sermanet, P.; Eigen, D.; Zhang, X.; Mathieu, M.; Fergus, R.; LeCun, Y. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. *arXiv* **2013**, arXiv:1312.6229.
17. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
18. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
19. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv* **2015**, arXiv:1506.01497.
20. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. *Nuclear Instrum. Methods Phys. Res. Sect. A Accel. Spectrom. Detect. Assoc. Equip.* **2015**, *794*, 185–192.
21. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In *ECCV 2016: Computer Vision—ECCV 2016*; Lecture Notes in Computer Science; Cham, Switzerland, 2015; pp. 21–37.
22. Ros, G.; Sellart, L.; Materzynska, J.; Vazquez, D.; Lopez, A.M. The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 3234–3243.
23. Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; Koltun, V. CARLA: An Open Urban Driving Simulator. *arXiv* **2017**, arXiv:1711.03938.
24. Maddern, W.; Pascoe, G.; Linegar, C.; Newman, P. 1 year, 1000 km: The Oxford RobotCar dataset. *Int. J. Robot. Res.* **2017**, *36*, 3–15. [[CrossRef](#)]
25. Stallkamp, J.; Schlipsing, M.; Salmen, J.; Igel, C. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In Proceedings of the International Joint Conference on Neural Networks, San Jose, CA, USA, 31 July–5 August 2011; pp. 1453–1460.
26. Larsson, F.; Felsberg, M. *Using Fourier Descriptors and Spatial Models for Traffic Sign Recognition*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 238–249.
27. Møgelmoose, A.; Trivedi, M.M.; Moeslund, T.B. Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 1484–1497. [[CrossRef](#)]
28. Maldonado-Bascon, S.; Lafuente-Arroyo, S.; Gil-Jimenez, P.; Gomez-Moreno, H.; Lopez-Ferreras, F. Road-sign detection and recognition based on support vector machines. *IEEE Trans. Intell. Transp. Syst.* **2007**, *8*, 264–278. [[CrossRef](#)]
29. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. *Int. J. Comput. Vis.* **2012**, *2012*, 1–45.
30. Udacity—21st Century University. Available online: <https://github.com/udacity/self-driving-car> (accessed on 5 November 2018).
31. Stallkamp, J.; Schlipsing, M.; Salmen, J.; Igel, C. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Netw.* **2012**, *32*, 323–332. [[CrossRef](#)] [[PubMed](#)]



32. Mao, X.; Hijazi, S.; Casas, R.; Kaul, P.; Kumar, R.; Rowen, C. Hierarchical CNN for traffic sign recognition. In Proceedings of the 2016 IEEE Intelligent Vehicles Symposium (IV), Gothenburg, Sweden, 19–22 June 2016; pp. 130–135.
33. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [[CrossRef](#)] [[PubMed](#)]
34. Danelljan, M.; Häger, G.; Shahbaz Khan, F.; Felsberg, M. Accurate Scale Estimation for Robust Visual Tracking. In Proceedings of the British Machine Vision Conference, Nottingham, UK, 1–5 September 2014; pp. 65.1–65.11.
35. Wojke, N.; Bewley, A.; Paulus, D. Simple Online and Realtime Tracking with a Deep Association Metric. *arXiv* **2017**, arXiv:1703.07402.
36. Grabner, H.; Grabner, M.; Bischof, H. Real-Time Tracking via On-line Boosting. In Proceedings of the British Machine Vision Conference, Edinburgh, UK, 4–7 September 2006; pp. 1–10.
37. Babenko, B.; Yang, M.H.; Belongie, S. Visual Tracking with Online Multiple Instance Learning. 2009. Available online: <http://faculty.ucmerced.edu/mhyang/papers/cvpr09a.pdf> (accessed on 5 November 2018).
38. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Comput. Soc.* **2014**, doi:10.1109/TPAMI.2014.2345390. [[CrossRef](#)]
39. Kalal, Z.; Mikolajczyk, K.; Matas, J. Tracking-learning-detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 1409–1422. [[CrossRef](#)] [[PubMed](#)]
40. Kalal, Z.; Kalal, Z.; Mikolajczyk, K.; Matas, J. Forward-backward error: Automatic detection of tracking failures. In Proceedings of the 20th International Conference on Pattern Recognition, Istanbul, Turkey, 23–26 August 2010; pp. 2756–2759.
41. Bolme, D.S.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual object tracking using adaptive correlation filters. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 2544–2550.



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# Efficient Neural Network Implementations on Parallel Embedded Platforms Applied to Real-Time Torque-Vectoring Optimization Using Predictions for Multi-Motor Electric Vehicles

Martin Dendaluce Jahnke <sup>1,2,\*</sup>, Francesco Cosco <sup>3</sup>, Rihards Novickis <sup>4</sup>, Joshué Pérez Rastelli <sup>2</sup> and Vicente Gomez-Garay <sup>1</sup>

<sup>1</sup> System Engineering and Automation Department, University of the Basque Country (UPV/EHU), 48013 Bilbao, Spain; vicente.gomez@ehu.es

<sup>2</sup> Automotive Department of Tecnalia Research and Innovation, 20009 Donostia - San Sebastián, Spain; joshue.perez@tecnalia.com

<sup>3</sup> Department of Mechanical Engineering at KU-Leuven, 3001 Leuven, Belgium; francesco.cosco@kuleuven.be

<sup>4</sup> Institute of Electronics and Computer Science (EDI), 1006 Riga, Latvia; rihards.novickis@edi.lv

\* Correspondence: mdendaluce001@ehu.eus; Tel.: +49-15252958611

Received: 31 December 2018; Accepted: 13 February 2019; Published: 22 February 2019

**Abstract:** The combination of machine learning and heterogeneous embedded platforms enables new potential for developing sophisticated control concepts which are applicable to the field of vehicle dynamics and ADAS. This interdisciplinary work provides enabler solutions -ultimately implementing fast predictions using neural networks (NNs) on field programmable gate arrays (FPGAs) and graphical processing units (GPUs)- while applying them to a challenging application: Torque Vectoring on a multi-electric-motor vehicle for enhanced vehicle dynamics. The foundation motivating this work is provided by discussing multiple domains of the technological context as well as the constraints related to the automotive field, which contrast with the attractiveness of exploiting the capabilities of new embedded platforms to apply advanced control algorithms for complex control problems. In this particular case we target enhanced vehicle dynamics on a multi-motor electric vehicle benefiting from the greater degrees of freedom and controllability offered by such powertrains. Considering the constraints of the application and the implications of the selected multivariable optimization challenge, we propose a NN to provide batch predictions for real-time optimization. This leads to the major contribution of this work: efficient NN implementations on two intrinsically parallel embedded platforms, a GPU and a FPGA, following an analysis of theoretical and practical implications of their different operating paradigms, in order to efficiently harness their computing potential while gaining insight into their peculiarities. The achieved results exceed the expectations and additionally provide a representative illustration of the strengths and weaknesses of each kind of platform. Consequently, having shown the applicability of the proposed solutions, this work contributes valuable enablers also for further developments following similar fundamental principles.

**Keywords:** machine learning; neural networks; predictive; vehicle dynamics; electric vehicles; FPGA; GPU; parallel architectures; optimization

## 1. Introduction

As vehicle electrification moves forward [1,2], new powertrain topologies are appearing and attractive research and innovation opportunities for developing enhanced propulsion systems can be identified [3]. The potential of the increased degrees of freedom and controllability of powertrains driven by multiple electric motors can be unleashed by exploiting several enabler

technologies which are addressed in this work, aiming to implement sophisticated Torque-Vectoring techniques [4–6]. Furthermore, this kind of advanced active chassis control systems not only can be eventually categorized inside the field of ADAS by themselves but they also can be exploited to support further ADAS- or even automated driving- functionalities, such as active support in critical evasive manoeuvres or to provide predictions and estimations of unmeasurable variables to take corresponding actions.

The cited applications imply multiple requirements which are satisfied by modern high performance heterogeneous embedded platforms. While keeping reasonable cost points and power consumption, they are bringing vast computing power and intrinsic parallelism (for performance) and redundancy (for safety). This power can be harnessed to implement complex algorithms, including Machine Learning, in both the cited application fields [7–18]. Furthermore, such advanced controller designs can be greatly supported by the means of the continuously improving model-based development solutions. However, major challenges appear not only on the notably complex technical layers but also in the regulatory layer addressing safety-critical applications [19,20].

This context has motivated the research presented in this work, in which after a deeper discussion about the abovementioned topics, we use Machine Learning to target a complex optimization problem—as are vehicle dynamics- in a typically constrictive domain—as are automotive systems-, thus providing an illustrative, innovative and challenging application. Specifically, this work leads to a detailed analysis of the embedded implementation of a conventional neural network (NN) aiming to rapidly provide batches of predictions within a real-time optimization algorithm. The representative use case consists in predicting the slip of the wheels for a batch of possible future control actions determined by a multi-objective Torque-Vectoring optimization algorithm. We have implemented the computationally demanding NN inference, which delivers batches with many evaluations per control cycle, on two different highly parallel platforms, aiming to evaluate their suitability and potential. The embedded platforms are a FPGA and a GPU, each integrated inside SoC devices with adequate industry suitability.

The remainder of the paper is structured as follows. Section 2 extends the key topics discussed in this introduction section, highlighting the most relevant aspects of the technological context which motivate and support this research, focusing particularly on the embedded platforms, as they are the keystone to enable the upcoming implementation work. Section 3 introduces the particular automotive control use case, targeted to illustrate the applicability of the proposed solution. Section 4 describes the proposed control solution, starting with an overview over the general approach, discussing relevant restrictions regarding automotive systems and finally introducing the NN itself. Sections 5 and 6 carefully describe the implementation of the said NN inference, respectively using embedded GPU and FPGA components. After a brief introduction focusing on the different computation paradigms, implementation details are presented and discussed by means of theoretical discussion and intermediate results. Section 7 summarizes the final results, first regarding the prediction capability of the NN itself and then focusing on the embedded implementation. Finally, Sections 8 and 9 summarize conclusions and future work respectively.

## 2. Technological Context

### 2.1. Heterogeneous Embedded Platforms

The field of embedded platforms has typically shown a steady evolution regarding computing power and features. But beyond that, recent years are showing an uprising variety of new solutions providing notable and attractive computing capabilities, as exemplified in Figure 1. This is fuelled not only by faster clock rates and multi-core designs—which are only providing a moderate performance growth rate- but also by the integration of different computing paradigms into heterogeneous embedded platforms which can act as accelerators for complex algorithms. This evolution has been strengthened by the fact that the application fields for platforms such as FPGAs and GPUs have been

widened. The result is a remarkable leap forward regarding computing capacity powered by their vast intrinsic parallelism, with the potential of providing at least one order of magnitude gain with respect to conventional processor-based devices [7,21–26].

The remainder of this section provides an overview about the more relevant embedded platform types and the evolution of their topologies, emphasizing the trend towards heterogeneous devices and SoCs.

Microcontrollers keep enhancing their capabilities for hard real-time and safety critical applications by implementing redundant cores—i.e., lockstep- and diverse error protection mechanisms. Besides, they are also showing performance gains not only through higher clock frequencies and parallel cores but also offloading functions to dedicated hardware modules [27,28].

Application oriented microprocessors provide typically higher clock frequencies and more cores than microcontrollers, often even with notable 64 bit processing capabilities providing faster double precision floating point operations. They represent a powerful solution for a wide variety of demanding applications but unless combined with elaborate software solutions or some other elements—e.g., real-time co-processing cores and certain integrity mechanisms- their suitability for safety-critical real-time control functions is rather limited. Equally to the microcontrollers, their sequential code execution paradigm only is capable of providing limited parallelism [29,30].

FPGAs enable very fast cycle times and massive throughput due to their programmable hardware nature with intrinsic parallelism and pipelining, as better detailed in Section 6. As their performance figures grow at a remarkable rate and good capabilities are available even for cost-sensitive devices, their current and potential application fields are widening [7,21,22,24,26,31].

GPUs base their instruction execution paradigm on a massive multi-core architecture which also provides high intrinsic parallelism. They have evolved from graphical and multimedia applications to also tackle computing tasks in desktop environments, also propagating this trend to embedded devices. An in-depth discussion on these devices is given in Section 5 [7,9,10,25,32].

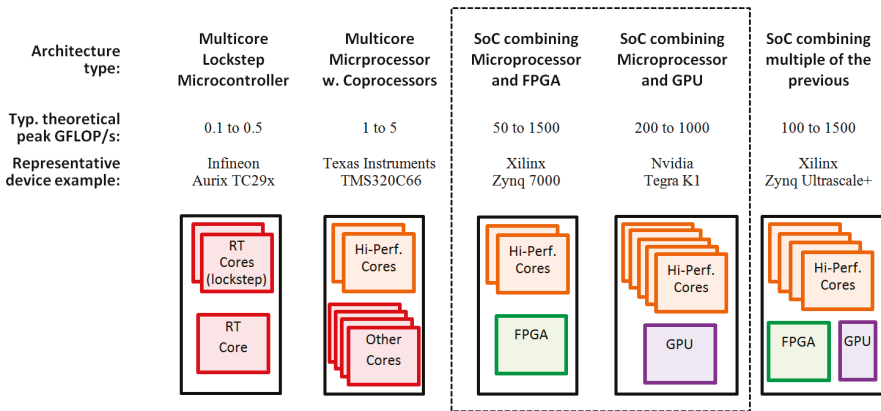
Finally, SoCs are bringing a new generation of heterogeneous devices to the market which offer different combinations of the abovementioned device types. Besides SoC approaches typically used in other domains such as multimedia consumer products, new computation and control oriented SoCs offer great improvements. Over one order of magnitude of performance gain can be expected by combining multi-core processors with at least one of the previously mentioned highly parallel GPU and FPGA computing solutions [21–25,33].

Figure 1 illustrates the mentioned variety of platform types, with representative examples focusing on the topology of the architectures. Additionally, some indicative GFLOP/s (giga floating point operations per second) values are given to illustrate the performance order of magnitude that can be expected. It is worth noting that these values are rough theoretical peak numbers and must not be misinterpreted: the effective performance figures depend greatly on both the application itself and its implementation, thus requiring exhaustive analysis for each case. Achieving maximum performance requires correctly exploiting the capacity of the instruction sets, caches, parallelism, different kinds of pipelining and so forth. Such complex implementation aspects are discussed in detail, for the selected FPGA and GPU platforms, in the upcoming Sections 5 and 6 [26,34,35].

Furthermore, it must be noted that some of the highest performing devices are excessively costly to be considered suitable for the typical automotive price ranges. The highest performing microcontrollers and microprocessors might have a borderline reasonable cost point but currently only the SoCs on the lower performance end are considered to be suitable for the typical automotive cost constraints, although technology and the market keep improving the performance/cost ratios.

The computing power provided by the more powerful platforms previously discussed, enables implementing relatively complex and computationally demanding algorithms. A clear example is the currently very active domain of perception algorithms for ADAS and automated driving, with current research involving Machine Learning and Deep Neural Networks (DNNs) [7,8]. Similarly, this paper

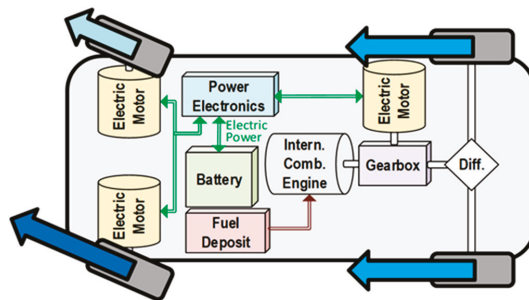
aims at benefiting from the discussed computation gains but for relatively smaller networks within more restrictive applications, as will be explained in Section 3.



**Figure 1.** Overview over relevant embedded platform types in the market, illustrating a simplified block diagram of their topology, providing indicative examples of the typical theoretical peak GFLOP/s performance for each type and one representative device example. Platforms used in this work are highlighted with dashed line box [24,25,29,33].

2.2. Multi-Electric-Motor Powertrains

Electrified vehicles—i.e., hybrid and pure electric vehicles- are showing a steadily growing sales trend [1,2]. This is not only pushed by environmental and geo-political/economic considerations which have brought stringent regulations, related to emission issues [36] and even certain scandals [37]. It is also driven by improving acceptance rates as technology is evolving and providing better capabilities at more affordable prices [1,2]. Consequently, some electrified powertrain topologies are evolving towards multi-motor configurations which are beyond conventional hybrids typically equipped with just one motor of each kind. Such a topology is illustrated in Figure 2.



**Figure 2.** Diagram of the Honda/Accura NSX hybrid powertrain topology. Blue arrows indicate Torque-Vectoring algorithm applying different torque to wheels while driving a curve [3].

Powertrains with independent motors for each of the wheels on a same axis enable the implementation of refined Torque-Vectoring algorithms. Torque-Vectoring relies on controlling the torque of each wheel aiming not only to enhance cornering performance but also to enhance stability and consequently increasing safety [38,39], as will be further discussed in Section 3.

Although these new technologies are being introduced mostly in high-performance cars, they are expected to propagate to the mainstream sector, where they could provide the abovementioned

technical benefits, besides enabling constructive advantages like better space utilization. Table 1 summarizes a selection of relevant electrified vehicles, either hybrid or full electric. It illustrates that beyond conventional hybrids, multi-motor powertrains are progressively appearing, most of them enabling the implementation of Torque-Vectoring on two or four wheels, which is the foundation of the presented use case.

**Table 1.** Overview of some relevant multi-motor vehicles [3,40–47].

Powertrain Type	Hybrid FWD		Hybrid 4WD		Electric RWD	Full Elec. 4WD	Full Electric 4-Motor 4WD		
Vehicle	Chevrolet Volt	Porsche 918	Honda-Accura NSX	Mercedes AMG Project One	BMW i3	Tesla P100D	Mercedes AMG SLS e-Drive	Rimac Concept One	NIO EP9
Highlights	Value	Perform.Innovat.	4 motors (incl. petrol)		Optional range extender	Range Accelerat.	4 independent electric motors; Handling by Torque Vectoring; Record-breaking performance		
Motors	2 electric	2 electric (per axle) 1 petrol (rear axle)	3 electric (2 front, 1 rear) 1 petrol (rear)		1 electric 1 petrol	2 electric (per axle)	4 electric (per wheel)		
Torque Vect.	×	~ per axle	✓ front axle	✓ front axle	×	×	✓	✓	✓
Power	149 HP	887 HP	573 HP	~1200 HP	170 HP	525 HP	751 HP	1224 HP	1360 HP
Year (appear)	2015	2013	2016	2018	2014	2015	2013	2016	2016/TBD
Mass	1607 kg	1700 kg	1725 kg	TBD	1422 kg	~2200 kg	2110 kg	1900 kg	1735 kg
0–100 km/h	7.5 s	2.6 s	3.1 s	2.5 sec	7.0 s	2.5 s	3.9 s	2.5 s	2.7 s
Price	~34.000 \$	~900.000 \$	~150.000 \$	TBD	~43.000 \$	~140.000 \$	~400.000 \$	TBD	TBD

### 2.3. Complexity, Safety Criticality and Regulations Concerning Automotive Control Systems

Relevant changes worth considering are occurring, besides the powertrain control domain, in the field of vehicle control systems in general. As the number of components and functions in modern vehicles increases, the complexity and its derived issues are reaching hardly sustainable levels, with up to over 100 million lines of source code distributed on board among over 100 ECUs (electronic control units). In this scenario, besides infotainment and comfort features, the advent of vehicles with multiple motors and energy sources, as well as ADAS and Automated Driving capabilities, are strongly contributing to the above mentioned complexity issue [48].

The technical challenges associated to the introduction of modern complex control systems are increased further by several industry-related constraints. Firstly, some well-known aspects inherent to the big-scale vehicle industry need to be considered, such as extreme cost sensitivity and diverse modularity requirements, all combined with accelerating product cycles [49]. Secondly, major restrictions have been added to this already challenging scenario, with the introduction of new standards and regulations addressing safety critical systems from the functional point of view. Standards—such as the recently updated ISO-26262 [50]- require costly and time-consuming tasks to be fulfilled for the certification process. This includes methodically addressing requirement traceability, validation and documentation and the need for exhaustive analysis like HARA (hazard analysis and risk assessment) and FMEA (failure mode effects analysis) [28,51].

In what respects to the software development, several enabler technologies can be highlighted to provide means to tackle some of the challenges associated to these points. From the software point of view, steadily advancing toolchains enable elaborate model-based development approaches to be applied, typically aligned with the V development methodology. This includes highly automated tools providing hardware abstraction through automatic code-generation—or hardware synthesis for FPGAs-, which can be helpful to tackle the complexity of modern embedded platforms and exploit the features of heterogeneous devices. It also includes tools for automated testing—from Model-in-the-Loop (MiL) to Hardware-in-the-Loop (HiL)- and also requirement management and documentation solutions. Consequently, besides providing means for agile, efficient and modular

developments, they can also notably facilitate the required test and validation of complex algorithms, as the ones discussed in this work [6,51,52].

In what respects to the hardware, the embedded platforms discussed in the previous Section 2.1 also represent a set of enabler technologies with notable relevance by providing protection mechanisms, redundancy and diversity. Furthermore, the gains in computational power and the multicore and heterogeneous topologies, do enable a further solution: consolidating the control architecture by integrating multiple control units into one ECU [48,53].

### 3. Targeted Application: Quadruple Electric-Motor Vehicle

The bottom-line of the above discussed topics leads to the motivation already anticipated in the introduction: we aim to provide enabler solutions targeting advanced Torque-Vectoring algorithms with real-time optimization, which represent a challenging application and a restrictive domain. In particular, our research efforts were dedicated to assessing the usability of automotive-suitable parallel embedded platforms for deploying such advanced control algorithms on vehicles with up to four independent electric motors.

The controller we propose uses NNs to estimate unmeasurable vehicle dynamics signals, in this case the prediction of future values. Being the target signals in the future, their values depend on the different control actions that the Torque-Vectoring could apply, meaning that the effect of many possible control actions will have to be evaluated and optimized in real time. Details on the control algorithm will be disclosed in Section 4.

Firstly focusing on the vehicle itself, quadruple-electric-motor powertrains present several advantages from the control point of view. By having independent control over the torque applied to each of the four wheels, they allow the implementation of sophisticated control strategies, basing on the Torque-Vectoring approach previously mentioned also in Section 2.2 and Figure 2 [6,38,54]. In comparison to the hybrid topology illustrated in this figure, in the targeted vehicle there is no differential on the rear axis, thus, in the same way as on the front axis, the torque on each of the rear wheels can also be directly controlled. This further increases the degrees of freedom of the controller. Besides, electric motors offer better controllability than internal combustion engines—as they provide a faster and more precise torque delivery [4,5]-, which is another reason for aiming at a more refined controller with short control cycles.

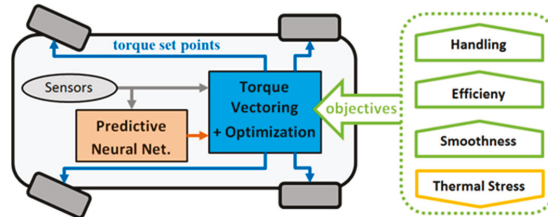
From the vehicle dynamics point of view, having four independent motors enables ideal Torque-Vectoring capabilities. The principle of operation is the following: when driving in a curve, dynamic weight transfer occurs along the vehicle's chassis, meaning that the wheels on the exterior side of the curve will temporarily support a greater normal force and provide greater traction capacity. This can be very conveniently exploited by reassigning a part of the torque from the interior wheels to the exterior ones. As has also already assessed by means of race-track tests in previous works [6,39,54,55], this does not only avoid the inner wheel to spin because of the lower traction capacity but it also generates an additional yaw moment, helping the vehicle rotate over its vertical axis towards the direction of the curve, thus mitigating understeer—i.e., the front wheels turning the car less than expected according to their steering angle- [38,56].

Besides the vehicle dynamics aspects, having four motors also provides enhancement potential regarding energy efficiency. This is due to the fact that each electric motor will have different degrees of efficiency depending on their operating point—i.e., current rotation speed and applied torque, as well as temperature-. Therefore, depending on the situation, a reassignment of the torque demand from one motor to another—i.e., a motor increasing its torque delivery with another being set to handle less power- could lead to a better global efficiency.

Furthermore, also thermal aspects are to be taken into account. Excessive temperatures on the motor components and power electronics must be avoided. As the motor loads will be unbalanced, so will the heat accumulation. This means that trying to re-balance the temperature distribution should be added as an extra aspect for the control function.

Finally, besides the handling, efficiency and thermal optimization, a fourth aspect to be considered is the smoothness of operation, aiming at improving stability and comfort, while reducing mechanical stress and wear on different components.

The aspects to be controlled that have been discussed in the previous paragraphs are illustrated in Figure 3 and represent a challenging multi-objective optimization problem which is addressed with an elaborate control solution, as described in the following Section 4.



**Figure 3.** Diagram from the control system perspective representing the targeted vehicle with 4 electric motors, illustrating a simplified block diagram of the algorithms and highlighting its objectives for the application.

#### 4. Advanced Controller Design: NNs for Batch Prediction Based Real-Time Optimization

##### 4.1. Description of the Control Problem

As described in the previous Section 3, the torque of four independent motors needs to be controlled in order to optimize not only the handling but also the efficiency, the thermal load and the smoothness of operation, all of which depends on different input variables. This means that it is not only a multi-objective optimization problem but also a MIMO (multiple input multiple output) system.

The 4 outputs of the system are the torque set points for each wheel ( $T_{FL}$ ,  $T_{FR}$ ,  $T_{RL}$ ,  $T_{RR}$ ). However, one degree of freedom is reduced and the control variables reformulated from 4 to 3, by imposing their sum to equal the requested total torque set point ( $T_{tot}$ ) according to the throttle command. A natural way to reformulate this problem from the automotive engineering perspective is to define the variables as follows:

$$\begin{aligned}
 T_{FL} + T_{FR} + T_{RL} + T_{RR} &= T_{tot} = \\
 &= (1 - D_{long}) \left[ (1 - D_{front}) T_{tot} + D_{front} T_{tot} \right] \\
 &+ D_{long} [(1 - D_{rear}) T_{tot} + D_{rear} T_{tot}]
 \end{aligned} \tag{1}$$

being  $D_{long}$  the longitudinal torque distribution along axes (% of torque to rear axis) and  $D_{front}$  and  $D_{rear}$  the lateral distribution among each of the axes.

Figure 4 provides a simplified overview of the controller-level diagram, highlighting in bold the parts of most relevance for this paper and marking with dashed lines the NN inference part that needs to be accelerated and on which the following sections will focus on.

In essence, the controller relies on a simpler Torque-Vectoring algorithm to determine the default torque distribution value. Then it applies a multi-objective optimization algorithm in order to find an optimized torque distribution in the surroundings of the default value.

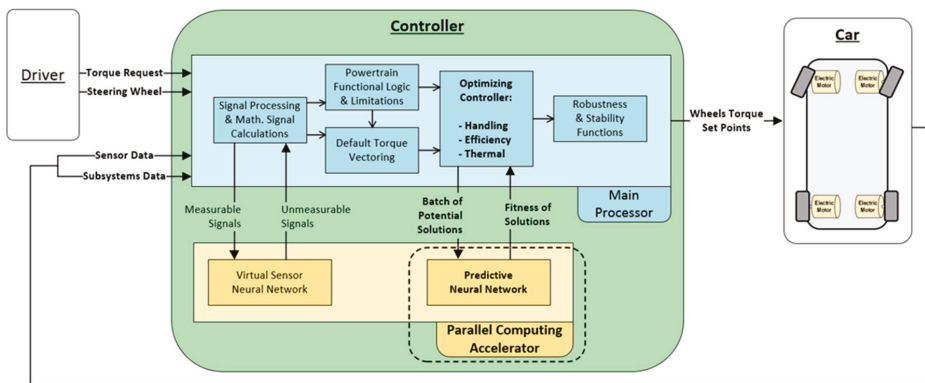
It is worth noting that the multi-objective MIMO optimizing strategy dynamically adjusts the priorities of each of the objectives according to the situation in real-time, evaluating several inputs which provide information about the driving situation of the vehicle. Vehicle dynamics is generally the top priority objective, especially when the driving state is getting closer to critical limits. Otherwise, energy efficiency is kept predominant, unless temperatures get closer to the functional limits, which progressively increases the weight of the corresponding objective.

Vehicle dynamics are the most critical part from the safety point of view and they are also critical in what respects to their computation for an algorithm such as the one targeted in this work. Good



evaluation functions for aspects concerning efficiency, temperature and comfort can be integrated in the controller with reasonable effort due to their nature. But the complex and strongly non-linear interactions of the many physical magnitudes and degrees of freedom involved in vehicle dynamics are notably more complicated to model and control [56,57].

This paper focuses on the technical challenge of predicting such magnitudes in real-time under the constraints of an automotive control system. A relevant reference point is that even running a single instance of a vehicle dynamics model is already a major challenge on an embedded platform [57]. It must be considered that the targeted application requires not one but a batch of evaluations of the vehicle dynamics -to predict the effect of the control actions for the optimizing algorithm- which we have determined in a typical range between several hundreds and a few thousands. In order to reduce the computational cost and make it possible to execute all the evaluations in the stringent time of a control cycle which has been specified with 5 ms, we chose a Machine Learning approach as prediction solution. In particular, we selected a NN, which is further described in Section 4.3.



**Figure 4.** Diagram from the control system perspective representing the targeted vehicle with 4 electric motors, illustrating a simplified block diagram of the algorithms and highlighting its objectives for the application.

The targeted use case in this paper is the estimation of future slip values of each wheel for a batch of possible Torque-Vectoring set points. These predictions are used to select the torque distribution that will reduce unnecessary slip, while also satisfying the efficiency, thermal load and the additional smoothness criteria. In order to achieve good predictions, a careful selection of the relevant input signals is necessary, which was based on a combination of expert knowledge in the field of vehicle dynamics and evaluation of the outcome of different input sets. The signals firstly need to include the torque set points for each wheel, as it is their effect to be evaluated. The remaining signals are related to the current state of the vehicle and therefore are equal for all the potential solutions. These include, the current wheel slips, the steering wheel position, inertial sensor (accelerations and angular velocity on 3 axes) and vehicle speed, adding up to 16 inputs. Additionally, an extended input set of 24 is also evaluated, including derivatives of the inertial sensor and two metrics of theoretical nature related to the steering wheel and vehicle speed. These are meant to reflect the transient behaviour of the involved dynamics over time.

#### 4.2. Restrictions of Automotive Systems

One notable characteristic of automotive control systems is their series of relatively stringent constraints, as already anticipated in Section 2. On the side of the embedded electronics there are the requirements affecting their cost and features and consequently their performance. This means that the algorithm should be as computationally efficient as possible. But significant restrictions are also to

be considered in what respects to the functionality, especially for safety-critical control functions—as is the case for powertrain applications like the one presented- which are addressed by regulations like the ISO-26262 and the corresponding certification implications [28,51,52].

The control architecture and the individual sub-functions have been carefully conceived to reduce the degree of uncertainty, facilitate its mathematical analysis and mitigate the impact of an eventual prediction malfunction. This inevitably induces the control algorithm and the predictive and virtual sensing algorithms to be designed avoiding unnecessary complexity, not only to enhance computational efficiency but also to enhance robustness and furthermore facilitate the analysis of its intended behaviour. In this sense, a relatively simple NN architecture is preferred, which additionally reduces computational cost.

Another fundamental consideration to be highlighted is that the malfunction of a virtual sensor should be considered similarly to a malfunction of a physical sensor, as both can potentially suffer some kind of error and need the corresponding support mechanisms to mitigate the effect.

The final consideration is that the designed optimizing algorithm is conceived to provide an enhanced set-point over the default Torque-Vectoring algorithm, aiming to reduce the slip but it must not be permitted to cause critical situations. It might happen that anyway certain slip occurs, in the same way slip can happen with a generic Torque-Vectoring or simply a normal torque distribution. The worst case scenario would be some other unwanted effect which could eventually affect the stability of the vehicle. But in such an unlikely situation, a superior layer of conventional traction and stability control functions—i.e., TCS and ESP- can override the torque set-points.

To mitigate the risk of stability systems having to intervene in such a worst case scenario, a series of additional pre-emptive mechanisms have been put in place in order to enhance the robustness and stability both of the estimations and the control actions. These include simple elements such as saturators and rate-limiters but also more elaborate functions that detect excessive fluctuations in multiple time windows. Furthermore, a simplified mathematical model is used to enable plausibility checks. Whenever unexpected values are detected, logical functions trigger an error and smoothly fall back to the default Torque-Vectoring values.

#### 4.3. Definition of the Algorithm: A Neural Network

The remainder of this work focuses on the NN in particular and its implementation on the two selected embedded platforms. We chose a relatively simple Feedforward NN algorithm for a variety of reasons, having also considered other plausible approaches for a system with a dynamic time behaviour as the one described, like for instance Recurrent Neural Networks (RNN) [58–60] or Long Short Term Memory (LSTM) Networks [61–63], as well as solutions involving Kalman Filter based approaches [64–66]. Considering that a review over the broad diversity of possible solutions or even a more detailed comparison with respect to the ones just mentioned, cannot be covered in the scope of this paper, it is still worth noting some relevant points which have supported the selected option. The fundamental reason is that the performance of this simpler NN can be considered as adequate regarding the accuracy and tolerances for this controller, as will be seen in the results in Section 7. Furthermore, in general terms, the simplest possible design is preferred for the sake of reducing complexity in what refers to computation -for speed- as well as in what refers to the analysis of the NNs behaviour -for robustness and regulations-, as already discussed in the previous sections. Furthermore, information about the variation in time of signals for which this information is relevant can be included into to NN by the means of feeding filtered derivatives of those signals into its inputs. Furthermore, this approach simplifies the layered kernel approach on the GPU, as well as the hardware implementation on the FPGA. In fact, the very same logical blocks that are used in the FPGA for the batch predictions, can easily also be used for simpler virtual sensing purposes, which is convenient to save programmable area and associated costs.

This section describes briefly the fundamentals of the proposed NN archetype, the Multilayer Perceptron [67,68], for the purpose of setting a common ground for the upcoming platform dependent implementation discussions (Sections 5 and 6).

As illustrated in the flow chart in Figure 5, once the input signals are received, they need to be scaled to normalized values, before the execution of the actual NN can start with the hidden layers. For each hidden layer, each of its neurons needs to apply a weight to each of the signals from the previous layer (or the inputs, in the case of the first layer) and calculate the resulting sum. This can be represented as a multiplication of a vector with a matrix and can be also implemented as dot products. Having the intermediate results of the previous layer, the bias values are added to each output. The final output of the layer is obtained after applying the activation function for each neuron. This process is repeated for each hidden layer, until reaching the output layer. This last layer does not have an activation function and will provide the results to be scaled to get the actual NN outputs.

It is relevant to understand that due to the vectorial and matrixial character of the operations, the computational complexity and spatial complexity are not neglectable. The dimensions are represented as  $L_0$  for the amount of inputs and  $O$  for the output count. Each hidden layer numbered from 1 to  $H$  with the index  $i$ , has  $L_i$  neurons, meaning that the dimension of the weight matrix will be  $L_i \times L_{i-1}$  and the bias vector  $L_i$ . Consequently, the count of parameters (weights and biases) and basic math operations are expressed in (2) and (3) respectively.

$$NN_{Params} = 2L_0 + \sum_{i=1}^H (L_i(L_{i-1} + 1)) + O(L_H + 3) \tag{2}$$

$$NN_{Ops} = 2L_0 + \sum_{i=1}^H (L_i(2L_{i-1} + 1)) + O(2L_H + 3) \tag{3}$$

The mathematical operations from (3) are decomposed in multiplications and additions in (4) and (5).

$$NN_{MultOps} = L_0 + \sum_{i=1}^H L_i L_{i-1} + O(L_H + 1) \tag{4}$$

$$NN_{AddOps} = L_0 + \sum_{i=1}^H (L_i(L_{i-1} + 1)) + O(L_H + 2) \tag{5}$$

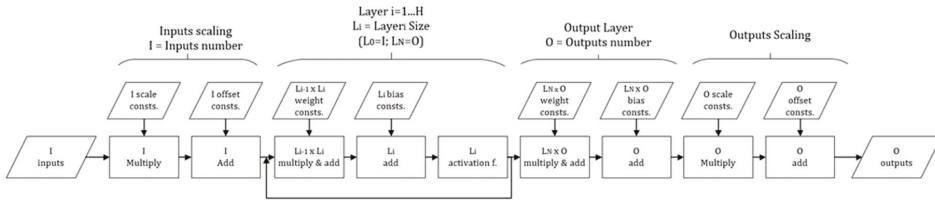
To the previous basic arithmetic operations, the operations for the computation of the activation function needs to be added. As each hidden neuron includes an activation function, the number of activation functions equals the number of hidden neurons, as in (6).

$$NN_{hidden\_neurons} = \sum_{i=1}^H (L_i) \tag{6}$$

In this work a NN with a sigmoidal activation function, has been selected, expressed as in (7).

$$\text{Sigmoid}(x) = \frac{2}{1 + e^{-2x}} \tag{7}$$

Besides increasing both the addition and multiplication count by one per hidden neuron, the reciprocal division and especially the exponential function can represent a notable computation cost, depending on the platform and the implementation. To tackle this issue, an approximate function could be implemented by the means of a look-up table (LUT) with a specific amount of data points. Therefore these operations are counted separately.



**Figure 5.** Flow chart of the generalized version of the implemented neural network inference algorithm, indicating the dimensions of the data and the amount of operation.

The following Table 2 collects the theoretical complexity magnitudes—parameter and operation counts- for a few representative NN topology examples. A NN with three hidden layers  $L_{1-3}$  of 32, 16 and 8 neurons respectively with a reduced input set  $In$  of 16 is taken as default topology for the targeted use case. As for all other cases, the output count  $Out$  is 4—one per wheel-, thus the topology is {16, 32 | 16 | 8, 4}. This is also taken as baseline to normalize relative complexity for other topologies. The same topology with an extended input set of 24 is also considered in the second row. The third row reflects a smaller topology which will be discussed during the FPGA development in Section 6. The remaining rows show examples for bigger topologies which might be of interest for other applications, illustrating the notable growth of complexity and the fact that even for the same quantity of neurons, these magnitudes change depending on the distribution across layers.

**Table 2.** This theoretical values of complexity for different neural network topologies.

Topology		Complexity						
In (L0)	Hidden Layers			Out (O)	Parameters	Operations	Activation Func. (Hidden Neurons)	
	L1	L2	L3	L4				
16	32	16	8	0	4	1284	2468	56
24	32	16	8	0	4	1556 (+21%)	2996 (+21%)	56 (+0%)
8	16	12	8	0	4	512 (−60%)	960 (−61%)	36 (−36%)
32	32	16	8	0	4	1828 (+42%)	3524 (+43%)	56 (+0%)
32	32	24	0	0	4	2020 (+57%)	3908 (+58%)	56 (+0%)
32	64	32	16	0	4	4860 (+279%)	9532 (+286%)	112 (+100%)
32	128	32	16	0	4	9020 (+602%)	17,788 (+602%)	176 (+214%)
32	64	64	32	16	4	9020 (+602%)	17,788 (+602%)	176 (+214%)

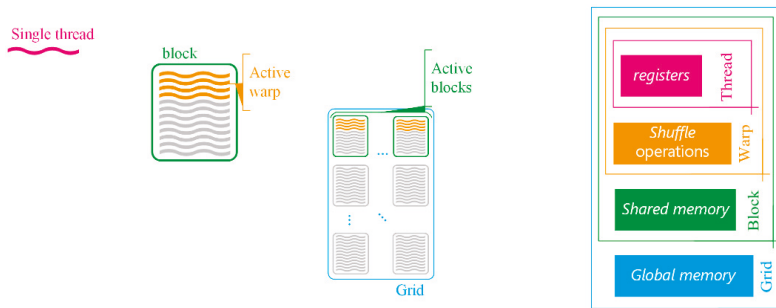
The impact of the scaling operations is almost negligible, accounting for  $L_0 + O$  multiply and addition operations (e.g., just 40 operations for the baseline NN). Furthermore, as these can be optimized by implementing them in the interfacing functions instead of repeating them for each NN, they are not included in our performance metrics. Similarly, as the presented work focuses on the computation and memory aspects of the GPU and FPGA implementations—and being the data transfer for the presented use case relatively small- the application and platform dependent overhead related to the different interfacing aspects are not accounted in this work.

**5. GPU Implementation of Neural Network**

GPUs are highly parallel instruction-based platforms which can be seen as a vastly extended multi-core processor but with a different architecture and instruction set. Historically developed for massive parallel manipulation of tasks related to computer graphics -such as pixel manipulation, three-dimensional graphics or filtering-, their architecture has been specifically optimized for high performance data-parallel workloads. After the advent of the programmable pipeline, usage of GPUs for general purpose applications affirmed as an active research field [69–71]. Furthermore, recognizing and exploiting synergistically the potential of both CPU and GPU architectures—e.g. in SoC devices- has showed to deliver even further computational gains [72,73].

The computational power and parallelism of GPUs has already been largely adopted for training DNNs, mainly targeting computer vision and other machine learning applications [8–10]. Besides, GPUs demonstrated to excel also for deployment of DNNs, with several tools arising for tackling the DNN inference challenge within dedicated embedded platforms [11,74].

For the developments presented along this paper, we exploit CUDA, a parallel computing platform and programming model created by NVIDIA [75]. In this environment and as is illustrated in Figure 6, heterogeneous programming is performed by having one host processor in charge of launching the execution of the parallel workloads on the GPU. Each program executed on the GPU is referred to as a kernel. Parallel execution of all threads in a kernel is structured as a grid of blocks, each block containing several threads, which are actually synchronously executed in several smaller sets called warps. Memory and communication among all the threads of a kernel are also structured following the same hierarchy (illustrated on the right side of Figure 6). Each thread has exclusive access to its registers but it is able to share its registers within all the threads of the same warp by means of shuffle operations. Within each block, threads can exchange information via shared memory, whereas the much slower global memory must be used for the wider scope of the grid level [75].



**Figure 6.** Illustration of the CUDA programming model: threads organization (left) and inter-thread communication mechanisms (right).

In this programming model, execution of the blocks is hardware dependent, with a block being allocated for execution on a given Streaming Multiprocessor (SM), as soon as its resource demand is satisfied. Therefore, the performance of such CUDA accelerated programs is strongly platform dependent.

For the sake of this work, we selected the affordable NVidia Tegra K1 SoC platform. The Tegra K1 features a GK20a GPU (Kepler architecture), including 192 SM3.2 CUDA cores clocked up to 852 MHz. The SoC is complemented with 4 ARM Cortex-A15 cores (ARMv7-A architecture) clocked at 2.0 GHz and an additional low power core at 1 GHz. It supports up to 8GB of (LP)DDR3 RAM. The warp size for the current architecture is fixed at 32, meaning that a program needs to have at least 12 warps running in order to theoretically maximize occupancy [25].

The remainder of the section describes the GPU implementation of the NN inference algorithm described in Section 4.3, focusing on the most important aspects necessary to unleash the full potential of the selected platform.

### 5.1. Multi-Kernel NN Inference

As a first step, we programmed—in C language- a single-thread CPU implementation of the NN inference algorithm described in 4.3 and then migrated it to GPU by means of CUDA. As depicted in Figure 7, the NN inference implies a sequence of different parallelizable tasks. Besides the values propagated between tasks, reading of constant parameters is also necessary for weights and biases (blocks to the left in Figure 7).

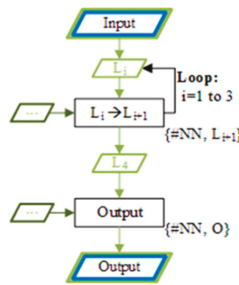


Figure 7. Flow chart of the fundamental NN inference approach on GPU.

The simplest implementation, referred as GM1, relies on Global Memory for both storage requirements. The size of the kernel launches -expressed in braces in the Figure 7 and given in Table 3- was set to be equal to the output layer dimensions. In GM1, each layer is implemented as an individual kernel launch. The profiling results -corresponding to the computation of a set of 512 NNs- confirmed the potential of using GPUs for NN evaluation, with a substantial speed-up of around 27x, w.r.t. the CPU performance, while also highlighting two main drawbacks: a suboptimal occupancy due to the chosen kernel sizes and a massive use of memory (and in particular of the texture units, needed for caching the reading operations).

Table 3. NN inference on GPU: GM1 performance for batch of 512 NNs with {16, 32|16|8, 4} topology.

Layer	Kernel Size {grid <sup>a</sup> , block <sup>b</sup> }	Occupancy Achieved (Theoretical) [%]	Registers per Thread	Comput. Time [μs]
L1	{512, 32}	24.1 (25)	34	0.160
L2	{512, 16}	24.2 (25)	37	0.174
L3	{512, 8}	23.6 (25)	34	0.104
O	{512; 4}	22.7 (25)	21	0.061
Total				0.499

<sup>a</sup> Number of thread blocks executed; <sup>b</sup> Number of threads per block.

We achieved a slightly better implementation, GM2, by tuning the kernel launch sizes, such to have larger thread blocks, each of them working on a fixed amount of NNs. The proposed modification resulted in a better usage of the GPU (higher occupancies), which is also reflected in a performance gain of a factor ~2x w.r.t. GM1. However, both GM1 and GM2 implementations share the drawback of abusing global memory for storing the intermediate data of the hidden layers -as depicted in Figure 7- and for reading the parameters -weights and offset values-.

### 5.2. Single-Kernel NN Inference: Shared and Global Memory

As a second step we developed the enhanced implementations -SM1 and SM2- by introducing the use of shared memory. We derived SM1 from GM2, using the same work partition strategy among the thread. However, we implemented the algorithm as a single kernel launch in order to enable the usage of shared memory for storing intermediate results between layers, thus avoiding the high latency corresponding to read/store operations from global memory. As shown in Table 4, the transition to shared memory provided an additional performance gain of about ~3x. However, it also resulted in reduced occupancy levels, because the limited amount of shared memory of each SM (configured as 48 KB for the used Kepler architecture) became the limiting factor, with a total of maximum 5 (rounding down 48/9) blocks per SM active, instead of the maximum of 16 allowed by the GPU.

Therefore, we pursued a more efficient usage of shared memory in SM2. This was achieved by buffering the storage and reading to shared memory in order to reuse the same buffer among the different layers. As reported in Table 4, SM2 offers a substantial performance gain of ~1.5x w.r.t.

SM1, in line with the increased occupancy of ~1.75x and the increased complexity added to prevent race conditions.

**Table 4.** Inference on GPU: GM2, SM1, SM2 and CM implementations for a batch of 512 NNs with a {16, 32|16|8, 4} topology.

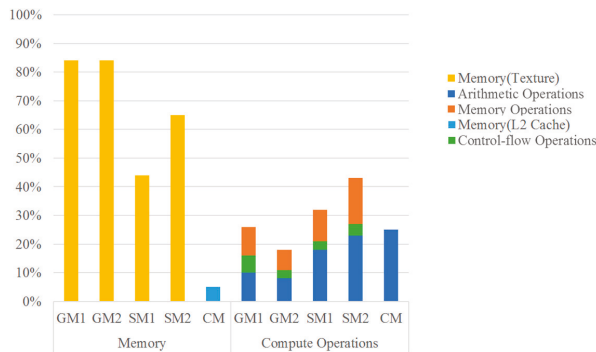
	Occupancy Achieved (Theoretical) [%]	Shared Memory (per Block)	Registers (per Thread)	Comput. Time [μs]
GM2-L <sub>1</sub> <sup>a</sup>	53.6 (62.5)	0 B	43	0.090
GM2-L <sub>2</sub> <sup>a</sup>	60.1 (75)	0 B	37	0.099
GM2-L <sub>3</sub> <sup>a</sup>	60.5 (75)	0 B	33	0.033
GM2-O <sup>a</sup>	76.8 (100)	0 B	21	0.006
GM2-Total	58.65 (71.69) <sup>b</sup>	0 B	38 <sup>b</sup>	0.237
SM1 <sup>a</sup>	26.7 (31.2)	9 KiB	50	0.089
SM2 <sup>a</sup>	46.6 (50.0)	4 KiB	52	0.064
<b>CM<sup>c</sup></b>	<b>24.9 (56.2)</b>	<b>0 B</b>	<b>52</b>	<b>0.052</b>

<sup>a</sup> Kernel Size: 16 blocks, each block of 128 threads; <sup>b</sup> Values normalized w.r.t overall duration; <sup>c</sup> Kernel Size: 4 blocks, each block of 128 threads.

### 5.3. Single-Kernel NN Inference: Registers + Constant Memory

In spite the notable net improvement w.r.t the CPU implementation, all the aforementioned implementations share the common bottleneck of reading the NN parameters (i.e., weights and bias of each layer) from global memory. As depicted in Figure 8, this resulted in overloading the texture memory buffer, with only marginal improvement achieved by the shared memory implementations.

The only feasible alternative for reading the constant values of the NN, is to exploit the GPU constant memory, which corresponds to a portion of the device memory accessed through constant cache. However, high reading performances from the constant memory are achieved only when all the threads access the same address simultaneously, in opposite to the concept of coalescent access pattern required for shared and global memory.



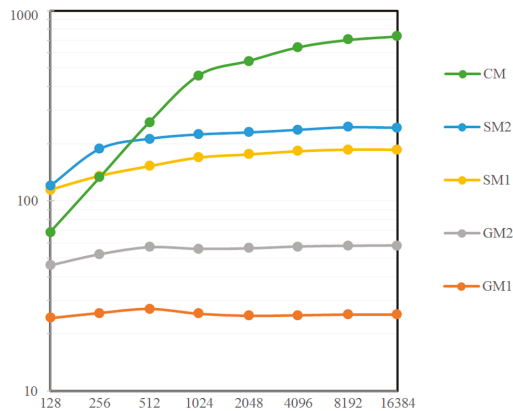
**Figure 8.** Distribution of memory usage and computing operations for different implementations.

To enable an effective use of constant memory, we implemented the algorithm illustrated in Figure 7 to dedicate each thread to the evaluation of a different NN, thus allowing the required perfectly coordinated access to constant memory. Moreover, being the data flow of each thread independent from the others, storage of the intermediate results was implemented on the local scope of each thread, avoiding the need of shared memory and the corresponding synchronization mechanisms required for preventing race conditions.

The resulting final implementation (referred as CM) achieved the best GPU performances, with an incremental performance up to  $\sim 3\times$  w.r.t the SM2 and an overall acceleration relative to a basic single-thread CPU implementation, ranging between  $\sim 68\times$  and  $\sim 730\times$  depending on the batch size, as showed in Figure 9. It can be observed how the batch size required to saturate the GPU depends on the implementation: while the computation time gains per NN for previous implementations saturate for batches beyond 256 or 512, the CM implementation takes 28 ns for 1024 evaluations, 20 ns for 4096 evaluations and as little as 18 ns for a batch of 8192 evaluations. Nevertheless, in spite of this remarkable maximum performance, it does not pay off for smaller batches, where the SM1 and SM2 implementations are faster.

This can be explained considering the theoretical occupancy of the CM version kernel. In fact, requiring 52 registers per thread (Table 4) and being launched in blocks of 128 threads, the GPU of the TK1 is saturated only when more than 9 blocks per SM are launched, which corresponds to a total of over 1152. Below that size, the size of the batch problem is not sufficient to compensate the latency of the required operations.

Additionally, the extended version of this NN with 24 inputs was also implemented. The computation time results of this are collected in Section 7.



**Figure 9.** Logarithmic representation relating the computation acceleration factor w.r.t. basic CPU implementation (vertical) for different batch sizes (horizontal) of the different GPU implementations of the NN with {16, 32 | 16 | 8, 4} topology.

## 6. FPGA Implementation of NN

FPGAs are versatile prefabricated silicon devices fundamentally consisting of generic logic cells, non-volatile embedded memory, digital signal processing blocks, input/output blocks and an interconnect structure. Basic FPGA structure is illustrated in Figure 10. Historically FPGAs arose as an alternative to ASICs (application specific integrated circuits) due to lower development and time-to-market expenses provided by their programmable character. Although, when compared to ASICs, FPGAs provide poorer cost/area, delay and consumption metrics, their flexible nature and performance turn them into convenient high performance devices for a diversity of solutions, especially in the field of signal processing applications with a relatively high throughput [76,77].

Internally, FPGA logic cells consist of a programmable combinational logic which feeds into fixed sequential logic in the form of D-type flip-flops (registers). This enables to pipeline a massive amount of data and to achieve parallel execution of algorithms. The principle of pipelining is illustrated in Figure 11. Essentially, in a fully pipelined solution, all functions (represented by the letter “F”) of the algorithm are executed concurrently and a new output is produced on every clock cycle once



the latency time—needed for the signals of the first instance to propagate through the system- has passed [76,78].

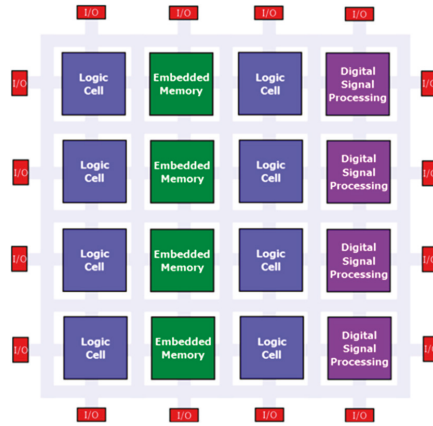


Figure 10. Simplified block diagram of a FPGA architecture.

Additionally, most modern FPGAs include DSP blocks, which usually ingrain multipliers and accumulators while improving power efficiency, chip size and timing for mathematical operations.

Regarding memory, embedded memory can be used as read-only-memory to store essential data but it can also be used for buffering. Memory is organized in blocks, which enable to use separate interfaces, meaning that they can be accessed concurrently [76,78].

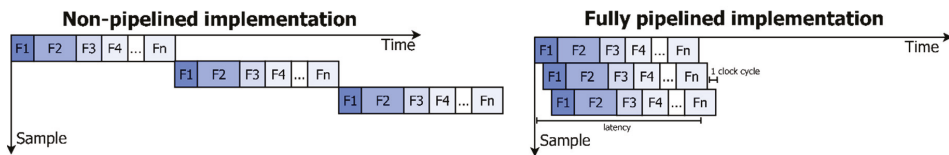


Figure 11. Graphical representation of the execution of functions on an FPGA.

The rise of more demanding algorithms—such as DNNs used for perception- is finding support in new features that vendors are introducing to FPGAs. For the implementation of algorithms using floating point variables on FPGAs, the usage of DSP blocks [7,79] and the reduction of numerical precision [80] are common solutions for the sake of performance and hardware resource efficiency.

Considerable research effort has been documented about NN implementations on FPGAs. Most of them focuses on convolutional NNs [12–15], whereas fewer works target smaller—or very small- feed-forward NNs [16–18]. Usually, the activation function of the NN is approximated using piece-wise-linear approximation or a LUT. Furthermore, for the sake of resource efficiency, floating point data is usually replaced with fixed point variables, thus reducing dynamic range [76,78].

For the sake of this work, we selected a Xilinx Zynq 7020 SoC platform. The Zynq 7020 features an Artix-7 FPGA combined with 2 ARM Cortex-A9 cores (ARMv7-A architecture) clocked at 666 MHz. The FPGA has 280 embedded memory blocks (4.9 Mb), 220 DSPs, 53200 LUTs (combinational logic) and 106400 registers (sequential logic). The processing system (ARM) and the FPGA are connected through multiple bidirectional interfaces, including support for direct memory access (DMA) and coherent and noncoherent access [24].

Although abstract schematics can be described effectively with hardware description languages such as VHDL and Verilog, this makes the implementations strongly hardware-specific thus limiting description reusability and maintainability. In view of a more solid applicability in the automotive

industrial context, this work exploits Xilinx SDx tools, a modern High Level Synthesis (HLS) solution provided by Xilinx in combination with its Software-Defined SoC workflow [81]. This toolchain addresses the SoC design in general, as it provides the connection between hardware accelerator and processor software while the HLS tools allows HDL and Verilog code generation from C or C++ code properly enriched with a set of certain directives.

The remainder of this section describes the implications and challenges faced to achieve an optimized NN implementation on the FPGA fabric.

### 6.1. Automatic Hardware Synthesis of NN Inference

We implemented the NN inference algorithms for the FPGA as a C++ object in accordance with the description in Section 4.3. Vivado HLS synthesizes this C++ code into RTL (register transfer level) description. The RTL generation flow is tuned by means of special directives to manage implementation aspects such as pipelining, loop unrolling, array instantiation and port specifications.

The generated hardware accelerator design is passed to the Xilinx SDSoC (Software Defined System on Chip) tool, which enables to select intercommunication interfaces, create the software layer and implement the data mover between the processor and the FPGA.

In this work, we used the FPGA master communications model, where data transactions are conducted by the FPGA master. The overall implementation architecture is shown in Figure 12. The Data Mover is implemented as DMA (direct memory access) controller, connecting the FIFO based accelerator interface with the ACP (accelerator coherency port) of the processor, thus supporting cache coherent transactions.

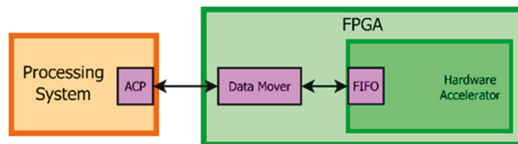


Figure 12. Block diagram of the interfacing mechanisms used by the master.

As the FPGA part of the Zynq does not include dedicated floating-point arithmetic, these operations need to be implemented with the available resources. This requires a series of compromise solutions depending on the resource usage of different implementations, which we had to bring into consideration using different data types and precisions for the sake of reducing resource usage and increasing speed.

### 6.2. Theoretical Performance and Resource Utilization

FPGAs excel at performance when a fully pipelined implementation is achieved, as previously discussed and illustrated in Figure 11. When fully pipelined and if data is continuously fed, a new output result -or results- will be generated at every clock cycle after the latency time has elapsed. For instance, in a fully pipelined solution with a latency of 1000 ns at 100 MHz, the first output would be available after this 1000 ns latency time but consecutive outputs would be available every 10 ns.

Following (3–7) the total count of mathematical operations is determined. These operations are implemented using deterministic functional blocks, thus it is possible to theoretically determine the resource utilization of a NN. Table 5 illustrates the amount of resources each mathematical operation requires for single precision data type when selecting implementations which exploit maximum hardware DSP block utilization [82].

**Table 5.** Resource utilization of single precision operations on Xilinx Zynq FPGA.

Instance	Resources		
	DSP	Registers	LUTs
Multiplier	3	143	321
Adder	2	205	390
Divider	0	761	944
Exponent	7	277	924

It must be noted that associating the total number of operations directly to the resource utilization corresponds to a fully pipelined implementation, which would provide a very fast throughput. This level of performance is determined to be above needed specifications and it would require excessive resources. By avoiding full pipelining, resource utilization can be reduced at the price of compromising speed. Nevertheless, for the following analysis, a smaller NN will be assumed, with a topology of {8, 16 | 12 | 8, 4}. Although this will still be too big for a fully pipelined implementation, it is adequate for different solutions of non-pipelined implementations.

Table 6 illustrates the theoretical amount of resources a fully pipelined implementation of this NN requires, indicating the percentage of available resources in relation to the Artix-7 FPGA which is integrated in the Zynq 7020. The table also shows the real values achieved through the synthesized HLS implementation. Furthermore, the resource utilization of HLS implementations of 32 bit and 16 bit fixed point versions is also provided, although in this case the theoretical calculation is not straight-forward due to low-level aspects such as bit-level optimizations.

**Table 6.** NN inference on FPGA: fully pipelined solutions for {8, 16 | 12 | 8, 4} topology.

Solution for NN	Resource Utilization								Latency [μs]
	BRAM		DSP		Registers		LUTs		
	Total	%	Total	%	Total	%	Total	%	
Single (theor.)	0	0.0%	2744	1247.3%	213,180	200.4%	425,412	799.6%	-
Single (HLS)	0	0.0%	2744	1247.3%	233,367	219.3%	427,469	803.5%	218
Fixed < 32 > (HLS)	0	0.0%	1148	521.8%	140,541	132.1%	216,208	406.4%	181
Fixed < 16 > (HLS)	0	0.0%	618	280.9%	73,890	69.4%	117,279	220.4%	134

The obtained numbers confirm that a fully pipelined implementation would still require far more hardware resources than the selected Zynq 7020 has available. Roughly 12x the available DSPs would be needed for the single precision version. For the 16 bit fixed point version this is reduced to roughly 3x available DSPs. Consequently, the implemented solution described in the following section is a compromise solution.

### 6.3. Resulting Hardware-based NN Inference

We have tried different NN implementation approaches, both regarding data type as well as regarding the usage of LUTs, as summed up in Table 7. LUTs of different data point count are used to replace the costly mathematical functions for the sigmoid of the activation function as in (4). This is meant to reduce resource usage -mostly regarding DSPs- and accelerate the computation but it must be noted that they cannot be pipelined.

Regarding constant data handling, we implemented weights and biases using registers, in order to improve their access speeds.

**Table 7.** NN inference on FPGA: implemented solution results for {8, 16 | 12 | 8, 4} topology.

Data Type	LUT	Resource Utilization								Normaliz. Mean Error	Computation Time [μs]
		BRAM		DSP		Registers		LUTs			
		Tot.	%	Tot.	%	Tot.	%	Tot.	%		
Single	No	0	0%	129	58%	23,919	22%	19,893	37%	~0%	5.53
Single	256	29	20%	98	44%	26,246	24%	21,782	40%	0.86%	4.92
Single	512	30	21%	98	44%	26,246	24%	21,873	41%	0.45%	5.04
Single	1024	30	21%	98	44%	26,246	24%	21,956	41%	0.29%	5.16
Fix32	No	0	0%	197	89%	23,051	21%	20,710	38%	0.04%	3.8
Fix32	256	27	19%	176	80%	18,739	17%	15,442	29%	0.89%	3.12
Fix32	512	27	19%	176	80%	18,739	17%	15,514	29%	0.46%	3.19
Fix32	1024	27	19%	176	80%	18,740	17%	15,738	29%	0.30%	3.25
Fix16	No	0	0%	86	39%	17,081	16%	16,867	31%	8.77%	2.02

The results illustrate that using LUTs accelerates the calculation and reduces the DSP usage, in exchange of using BRAM and some more registers and LUTs. The use of fixed-point data types reduces resource utilization, which permits to implement a faster solution, with more loop unrolling, which ultimately does consume more DSPs. Finally, the 16-bit precision solution outperforms the other solutions even without any loop unrolling, thus notably decreasing resource utilization but at the expense of excessive numerical error for the case of this implementation. Nevertheless, it must be noted that the error metrics should only be interpreted as indicative, as they would require further analysis, implementation optimization and training specifically for the specific data type.

As the previous intermediate results with the small NN have shown that acceptable performance and resource utilization can be achieved with fixed point data types, the originally targeted NN size is implemented for the 32 bit fixed-point implementation with the biggest LUT. The results for both 16 and 24 inputs versions are shown in Table 8.

**Table 8.** NN inference on FPGA: implemented solution results for a batch of 1024 NNs {16, 32 | 16 | 8, 4} and {24, 32 | 16 | 8, 4} topologies with 32 bit fixed point and 1024 value LUT solution.

NN Topology	Resource Utilization								Normalized Mean Error	Average Computation Time per NN [μs]
	BRAM		DSP		Registers		LUTs			
	Tot.	%	Tot.	%	Tot.	%	Tot.	%		
16, 32   16   8, 4	52	37%	184	83%	27,129	25%	40,787	76%	0.302%	4.20
24, 32   16   8, 4	56	40%	184	83%	30,872	29%	50,769	95%	0.318%	5.40

It must be noted that although avoiding the floating data type is a reasonable compromise which provides adequate results, it is expected that this compromise will eventually be less relevant in future FPGAs, as models with dedicated floating point functions are starting to appear even for low range families [83].

## 7. Results

Before proceeding to the core challenge discussed in this paper -which is the efficient implementation of the NN inference on the two automotive-suitable embedded platforms- Figure 13 shows the worst case prediction capabilities for the two selected NN topologies implemented in the targeted use cases: {16, 32 | 16 | 8, 4} and {24, 32 | 16 | 8, 4}. This particular plot is considered as worst case for two main reasons. Firstly, because with 50 ms it is providing a longer prediction horizon than really necessary. Secondly, because it is an extract of the most unfavourable situation among over 10 min of driving, subject to exceptionally strong fluctuations which result in notably worse accuracy and the appearance of a shift of up to roughly 15–20 ms.

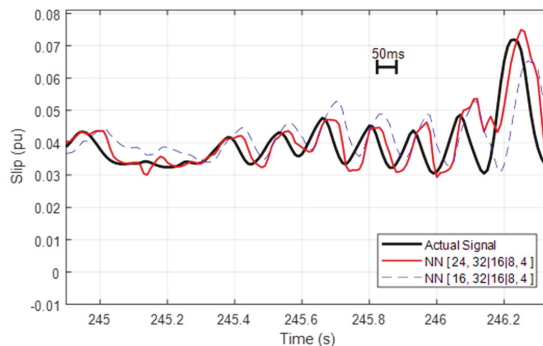
The bottom-line is that the achieved accuracy of the predictions is certainly satisfying and adequate for the application, with a mean average error (over the entire lap) of just 0.28% and 0.24% for the

NNs with 16 and 24 inputs respectively. For the especially difficult to predict situation illustrated in Figure 13, the values rise to 0.70% and 0.45% respectively. This illustrates that the NN provides sufficient capacity to adapt also to highly dynamic situations, including those of which only few fragments are present in the training dataset. It also shows the benefit of the extended inputs providing information about the derivatives of relevant signals.

Furthermore, several aspects must be emphasized in this respect. First, that the slip is strongly affected by random irregularities of the road surface, which are unknown and cannot be predicted. Second, that for the presented use-case, considering the lower time constant of the electric motors and the design of the presented Torque-Vectoring optimizing controller, the prediction can be adjusted to less challenging horizons under 50ms. Third, the said controller has been designed with far greater safety margins than the obtained accuracy, even for the worst case fragments.

In what respects to the setup and approach for the training and validation of these NNs, we generated a broad and diverse dataset, aiming to avoid overfitting and ensure good generalization, including a variety of road/track models (Nürburgring, Inta and different generic test tracks) and different driving styles (normal, aggressive, borderline and drifting) by different drivers. For this we relied on a high-fidelity multibody vehicle dynamics simulator (Dynacar, [84,85], also involved in real circuit tests in previous work for validation purposes [54,84]) integrated into an elaborate model-based development framework (using MatLab-Simulink for accelerated MiL tests as well as HiL validation [6,86]), together with models of other relevant components, subsystems and control units. Therefore, validation tests were performed using data generated by the same setup but driving differently. In particular, the final results were generated by one lap of mixed-style driving, including extreme manoeuvres on the Nürburgring, driving in opposite direction to the training data.

In conclusion this approach is highly representative to assess realistically the good learning and prediction capacity of the NNs and the outcome was successful even under the most challenging situations in what respects to dynamical behaviour of the predicted variables.



**Figure 13.** Worst case plot of a 50 ms horizon slip prediction on the front-left wheel with two NN topologies (extract of the most adverse situation among over 10min of driving).

Ultimately focusing on the principal technical challenge which is addressed in the implementation part of this paper, Table 9 provides a wrap-up of the intermediate results together with the final results for the embedded NN implementation on the two selected embedded devices, the GPU and the FPGA.

The execution time results show the remarkable computation capacity of the selected embedded GPU SoC, providing an average computation time of just 40 ns per NN for a batch of 1024 evaluations of the targeted NN with extended inputs and even below 20 ns for bigger batches with 16 inputs.

**Table 9.** Final summary of inference on GPU and FPGA for a batch of 1024 NNs with different NN topologies (selected solutions in bold).

Device	Implementation	Computation Time per NN [ $\mu$ s]		
		NN Topology		
		in: hidden: out:	{8, 16   12   8, 4}	{16, 32   16   8, 4}
GPU	GM1	-	0.509	0.567
GPU	GM2	-	0.232	0.264
GPU	SM1	-	0.077	0.087
GPU	SM2	-	0.058	0.071
<b>GPU</b>	<b>CM</b>	-	0.028	<b>0.040</b>
FPGA	Single (LUT-no)	5.534	-	-
FPGA	Single (LUT-1024)	5.092	-	-
FPGA	Fix32 (LUT-no)	3.832	-	-
<b>FPGA</b>	<b>Fix32 (LUT-1024)</b>	3.254	4.201	<b>5.403</b>
FPGA	Fix16 (LUT-no)	2.021	-	-

In contrast, the performance of the FPGA has suffered from the notable penalty of not being able to implement a fully pipelined solution, providing an average computation time per NN of 5.4  $\mu$ s for the extended inputs topology, which is 135x slower than the GPU. For a batch of 1024 evaluations for the conceived optimization algorithm, this is slower than required for the—quite demanding- 5 ms specification of the targeted application. Nevertheless, this also means that providing an efficient hardware interfacing solution, a suitable design is possible also for the FPGA if some adjustment of the use-case specification is made: reducing the batch size to 512 or alternatively either relaxing the cycle time to 10ms or slightly reducing the complexity and precision of the NN.

Another aspect is the scaling proportionality of the computation time, considering that the theoretical growth of computational complexity is 21% for extending the inputs from 16 to 24, as expressed in Table 2. For the FPGA implementation, the penalty grows to 29%. For the GPU, the effect clearly depends on the implementation: while the less efficient GM, GM2 and SM1 solutions suffer a penalty of just 11–14%, SM2 suffers an almost exactly proportional 22%. In contrast, the penalty for extending the inputs on the fine-tuned highly efficient CM version rises to 43%.

## 8. Conclusions

This work has successfully presented several NN implementations on parallel embedded platforms of different nature, FPGAs and GPUs, integrated in cost-sensitive and industry-suitable SoC devices. Consequently, it provides multiple enabler solutions for the innovative and challenging vehicle dynamics use-case proposed in this paper, as well as for further research also in other fields involving advanced control applications.

For the case of the targeted application, which required fast batch predictions of slip values for a real-time Torque-Vectoring optimization algorithm to control multi-motor electric vehicles, we assessed the technical feasibility of the developed implementations. These were conceived under consideration of the current technological context as well as the automotive domain constraints, which have been reviewed and discussed. Even when fulfilling the highly demanding design specifications defined in this particular use-case, the outcome met or even exceeded the expectations in what respects to both computational performance and accuracy.

The presented work has involved several notable particularities. One is restricting the embedded platforms to industry-suitable devices which due to a lower cost point, energy consumption and size, inevitably represent capacity and performance limitations with respect to typical non-embedded and high-end devices. Another is that the vehicle dynamics predictions corresponding to this application do not require complex NNs of massive dimensions as the ones used in other Machine Learning applications and in the Deep Learning domain. The additional challenge is instead brought by the

fact of having to calculate in under 5 ms, the effect of 1024 control actions (with 3 variables each) on 4 highly dynamic variables with a prediction horizon of up to 50 ms.

The GPU implementation has provided an outstanding performance, being two orders of magnitude faster than the FPGA thanks to its great parallelism. But this can only be achieved through very careful programming to maximize occupancy and avoid memory bottlenecks because, in spite of its high memory bandwidth, the high computational throughput can lead to data exchange inefficiencies causing a performance penalty of up to an order of magnitude.

The achieved FPGA implementation of the NN has showed one of the weaknesses of these kind of devices: if the algorithm is too big to be well pipelined, the performance dramatically drops. Consequently, in order to ensure sufficient performance, some compromise solutions need to be adopted in order to satisfy the demanding targeted specifications.

Our analysis also highlights that in cases involving smaller NNs or eventually bigger devices, aiming to achieve full pipelining, FPGAs do have the theoretical potential to provide similar -or even higher- throughput than the selected GPU. This would be beneficial especially for bigger batches of evaluations, to overcome the initial penalty of latency. In contrast, for the non-pipelined implementation shown in this work, the relative efficiency impact of the batch size is marginal for the FPGA, making it more suitable for small batches or for controllers which require a single evaluation per control cycle.

In conclusion this work highlights very clearly fundamental strengths and weaknesses of each platform type, depending on the application specification. Besides the pure performance aspects that have been discussed, while the FPGA is very space-constricted, the instruction-based paradigm of the GPU enables to implement greater functional diversity and to easily scale to bigger algorithms. On the other hand, while the GPU depends on the CPU to operate, the FPGA can run autonomously and could be directly interfaced with sensors and protocols, thus avoiding the initial penalty both platforms are subject to for interfacing with the processor. This would provide another clear benefit for use-cases with few evaluations per cycle on the FPGA.

Ultimately, both embedded platform types have shown their suitability and remarkable potential with the provided implementations. Furthermore, the obtained outcome is a very illustrative example of the fact that, when it comes to discussing the performance of complex highly parallel platforms based on such different execution paradigms, the dependency regarding to the application itself—as well as to the implementation approach- is very significant and needs to be analysed on a per-use-case basis.

## 9. Future Work

Having confirmed the relevance and feasibility of the application and enablers presented in this paper, future work will take this baseline and go into further depth regarding the application itself, as well as both embedded implementations.

Firstly, using the achieved implementations for very fast embedded NN execution, the benefits of the Torque-Vectoring optimization algorithm itself will be further pursued, paying special attention to the vehicle dynamics performance. Furthermore, the estimation capacities and implications of different NN designs will also be addressed. Besides, a more detailed analysis on the effect of precision loss due to data types and LUT utilization will be considered.

For both embedded implementations, an exhaustive analysis of the interfacing between the heterogeneous parts of the SoC must also be addressed, aiming to minimize the overhead caused by the data exchange. This will not only require careful programming of the interfacing mechanisms but it could also require to conceive solutions where some additional functionalities are integrated in the accelerator part, in order to further reduce the data exchange. For instance, the generation of the batch of values to be evaluated could be integrated. Furthermore, the selection of the best solutions for the optimization could also be accelerated.

Focusing on one platform, the FPGA implementation will be enhanced in different manners. Besides optimizations regarding aspects such as pipelining and data types, a major architectural

redesign will be approached. This is conceived to not simply implementing several network layers into the hardware but instead implementing a single cyclically reusable layer, with the capacity of a fast switching between weights and biases corresponding to different layers.

The FPGA will also be used for algorithms involving small batches and single evaluations as discussed in the conclusions, such as virtual sensor functions for current values (Figure 4, bottom left) or for prediction of values not subject to the real-time optimization of different control actions.

On the other hand, the performance of more powerful embedded GPUs for bigger batches and bigger NNs for different applications will also be addressed.

A final topic to be addressed with further depth is functional safety, analysing firstly the criticality of the application as a whole and secondly the criticality of the functionality of the NN. The distribution of functions and protection mechanisms between the processor and the accelerator will require exhaustive analysis, as well as carefully crafted validation approaches. Here, the benefits that the hardware-nature of the FPGA provides in this sense are to be discussed as well.

**Author Contributions:** Conceptualization, M.D.J. and F.C.; Funding acquisition, J.P.R. and V.G.-G.; Investigation, M.D.J., F.C. and R.N.; Project administration, M.D.J.; Supervision, J.P.R. and V.G.-G.; Visualization, M.D.J.; Writing—original draft, M.D.J., F.C. and R.N.; Writing—review & editing, J.P.R. and V.G.-G.

**Funding:** Some of the results presented in this work are related to activities within the 3Ccar project, which has received funding from ECSEL Joint Undertaking under grant agreement No. 662192. This Joint Undertaking received support from the European Union’s Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Romania, Belgium, United Kingdom, France, Netherlands, Latvia, Finland, Spain, Italy, Lithuania. This work was also partly supported by the project ENABLES3, which received funding from ECSEL Joint Undertaking under grant agreement No. 692455-2.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

- IEA. Electric Vehicles Initiative. Available online: [http://www.ertrac.org/uploads/documentsearch/id50/ERTRAC\\_ElectrificationRoadmap2017.pdf](http://www.ertrac.org/uploads/documentsearch/id50/ERTRAC_ElectrificationRoadmap2017.pdf) (accessed on 21 February 2019).
- ERTRAC. European Roadmap Electrification of Road Transport. Available online: [http://www.ertrac.org/uploads/documentsearch/id50/ERTRAC\\_ElectrificationRoadmap2017.pdf](http://www.ertrac.org/uploads/documentsearch/id50/ERTRAC_ElectrificationRoadmap2017.pdf) (accessed on 21 February 2019).
- Robinson, A. 2016 Acura NSX Dissected: Powertrain, Chassis and More—Feature. Available online: <http://www.caranddriver.com/features/2016-acura-nsx-dissected-powertrain-chassis-and-more-feature> (accessed on 24 February 2016).
- Louis, J.-P. *Control of Synchronous Motors*; John Wiley & Sons: New York, NY, USA, 2013; ISBN 978-1-118-60174-7.
- Isermann, R. *Engine Modeling and Control*; Springer: Berlin/Heidelberg, Germany, 2014; ISBN 978-3-642-39933-6.
- Dendaluce, M.; Allende, M.; Pérez, J.; Prieto, P.; Martin, A. Multi Motor Electric Powertrains: Technological Potential and Implementation of a Model Based Approach. In Proceedings of the IECON 2016—42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, Italy, 23–26 October 2016.
- Nurvitadhi, E.; Venkatesh, G.; Sim, J.; Marr, D.; Huang, R.; Ong Gee Hock, J.; Liew, Y.T.; Srivatsan, K.; Moss, D.; Subhaschandra, S.; et al. Can FPGAs Beat GPUs in Accelerating Next-Generation Deep Neural Networks? In Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, CA, USA, 22–24 February 2017; ACM: New York, NY, USA, 2017; pp. 5–14.
- Oskouei, S.S.L.; Golestani, H.; Kachuee, M.; Hashemi, M.; Mohammadzade, H.; Ghiasi, S. GPU-based Acceleration of Deep Convolutional Neural Networks on Mobile Platforms. *Distrib. Parallel Clust. Comput.* **2015**.
- Guzhva, A.; Dolenko, S.; Persiantsev, I. *Multifold Acceleration of Neural Network Computations Using GPU*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 373–380.
- Huqqani, A.A.; Schikuta, E.; Ye, S.; Chen, P. Multicore and GPU Parallelization of Neural Networks for Face Recognition. *Procedia Comput. Sci.* **2013**, *18*, 349–358. [[CrossRef](#)]
- NVIDIA Corporation. GPU-Based Deep Learning Inference: A Performance and Power Analysis. Available online: [https://www.nvidia.com/content/tegra/embedded-systems/pdf/jetson\\_tx1\\_whitepaper.pdf](https://www.nvidia.com/content/tegra/embedded-systems/pdf/jetson_tx1_whitepaper.pdf) (accessed on 21 February 2019).



12. Li, H.; Fan, X.; Jiao, L.; Cao, W.; Zhou, X.; Wang, L. A high performance FPGA-based accelerator for large-scale convolutional neural networks. In Proceedings of the 2016 26th International Conference on Field Programmable Logic and Applications (FPL), Lausanne, Switzerland, 29 August–2 September 2016; pp. 1–9.
13. Chakradhar, S.; Sankaradas, M.; Jakkula, V.; Cadambi, S. A dynamically configurable coprocessor for convolutional neural networks. In Proceedings of the ACM SIGARCH Computer Architecture News, Saint-Malo, France, 19–23 June 2010; ACM: New York, NY, USA, 2010; Volume 38, pp. 247–257.
14. Zhang, C.; Li, P.; Sun, G.; Guan, Y.; Xiao, B.; Cong, J. Optimizing fpga-based accelerator design for deep convolutional neural networks. In Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, CA, USA, 22–24 February 2015; ACM: New York, NY, USA, 2015; pp. 161–170.
15. Suda, N.; Chandra, V.; Dasika, G.; Mohanty, A.; Ma, Y.; Vrudhula, S.; Seo, J.; Cao, Y. *Throughput-Optimized OpenCL-Based FPGA Accelerator for Large-Scale Convolutional Neural Networks*; ACM Press: New York, NY, USA, 2016; pp. 16–25.
16. Hariprasath, S.; Prabakar, T.N. FPGA implementation of multilayer feed forward neural network architecture using VHDL. In Proceedings of the 2012 International Conference on Computing, Communication and Applications (ICCCA), Dindigul, Tamilnadu, India, 22–24 February 2012; pp. 1–6.
17. Youssef, A.; Mohammed, K.; Nasar, A. A Reconfigurable, Generic and Programmable Feed Forward Neural Network Implementation in FPGA. In Proceedings of the 2012 UKSim 14th International Conference on Computer Modelling and Simulation, Cambridge, UK, 28–30 March 2012; pp. 9–13.
18. Sahin, S.; Becerikli, Y.; Yazici, S. Neural network implementation in hardware using FPGAs. In Proceedings of the International Conference on Neural Information Processing, Hong Kong, China, 3–6 October 2006; pp. 1105–1112.
19. Macher, G.; Armengaud, E.; Kreiner, C. Integration of Heterogeneous Tools to a Seamless Automotive Toolchain. In *Systems, Software and Services Process Improvement*; O'Connor, R.V., Akkaya, M.U., Kemaneci, K., Yilmaz, M., Poth, A., Messnarz, R., Eds.; Communications in Computer and Information Science; Springer International Publishing: Berlin, Germany, 2015; pp. 51–62, ISBN 978-3-319-24646-8.
20. Pohl, K.; Honninger, H.; Achatz, R.; Broy, M. (Eds.) *Model-Based Engineering of Embedded Systems: The SPES 2020 Methodology*; Springer: Berlin, Germany; New York, NY, USA, 2012; ISBN 978-3-642-34613-2.
21. Trimberger, S.M. Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology. *Proc. IEEE* **2015**, *103*, 318–331. [[CrossRef](#)]
22. Strenski, D.; Sundararajan, P.; Wittig, R. The Expanding Floating-Point Performance Gap Between FPGAs and Microprocessors. Available online: [http://www.hpcwire.com/2010/11/22/the\\_expanding\\_floating-point\\_performance\\_gap\\_between\\_fpgas\\_and\\_microprocessors/](http://www.hpcwire.com/2010/11/22/the_expanding_floating-point_performance_gap_between_fpgas_and_microprocessors/) (accessed on 22 January 2015).
23. Altera. Cyclone V SoC Brochure. Available online: <http://www.altera.com/literature/br/br-soc-fpga.pdf> (accessed on 21 February 2019).
24. Xilinx Inc. *Xilinx Automotive Zynq-7000 All Programmable SoCs Brochure*; Xilinx Inc.: San Jose, CA, USA, 2014.
25. NVIDIA Tegra: The World's Fastest Mobile Processors. Available online: <http://www.nvidia.com/object/tegra.html> (accessed on 14 July 2017).
26. Che, S.; Li, J.; Sheaffer, J.W.; Skadron, K.; Lach, J. Accelerating Compute-Intensive Applications with GPUs and FPGAs. In Proceedings of the 2008 Symposium on Application Specific Processors, Anaheim, CA, USA, 8–9 June 2008; pp. 101–107.
27. Infineon Technologies AG Highly Integrated and Performance Optimized 32-Bit Microcontrollers for Automotive and Industrial Applications. Available online: <https://www.infineon.com/dgdl?fileId=5546d46153ac54890153c1a41ffe0000> (accessed on 21 February 2019).
28. Bernon-Enjalbert, V.; Blazy-Winning, M.; Gubian, R.; Lopez, D.; Meunier, J.P.; O'Donnell, M. Safety Integrated Hardware Solutions to Support ASIL D Applications. Available online: <http://www.nxp.com/docs/en/white-paper/FUNCSAFTASILDWP.pdf> (accessed on 21 February 2019).
29. Texas Instruments (Last) DSP—Overview—Processors. Available online: <http://www.ti.com/processors/digital-signal-processors/overview.html> (accessed on 31 December 2018).
30. Ward, B.C.; Herman, J.L.; Kenna, C.J.; Anderson, J.H. Outstanding Paper Award: Making Shared Caches More Predictable on Multicore Platforms. In Proceedings of the 2013 25th Euromicro Conference on Real-Time Systems, Paris, France, 9–12 July 2013; pp. 157–167.

31. Hemsoth, N. Latest FPGAs Show Big Gains in Floating Point Performance. Available online: [http://www.hpcwire.com/2012/04/16/latest\\_fpgas\\_show\\_big\\_gains\\_in\\_floating\\_point\\_performance/](http://www.hpcwire.com/2012/04/16/latest_fpgas_show_big_gains_in_floating_point_performance/) (accessed on 22 January 2015).
32. Strategy, M.I. A Machine Learning Landscape: Where AMD, Intel, NVIDIA, Qualcomm And Xilinx AI Engines Live. Available online: <http://www.forbes.com/sites/moorinsights/2017/03/03/a-machine-learning-landscape-where-amd-intel-nvidia-qualcomm-and-xilinx-ai-engines-live/> (accessed on 14 July 2017).
33. Zynq UltraScale+ MPSoC Data Sheet: Overview (DS891). Available online: [https://www.xilinx.com/support/documentation/data\\_sheets/ds891-zynq-ultrascale-plus-overview.pdf](https://www.xilinx.com/support/documentation/data_sheets/ds891-zynq-ultrascale-plus-overview.pdf) (accessed on 21 February 2019).
34. How to Reduce FPGA Logic Cell Usage by >x5 for Floating-Point FFTs. Available online: <https://www.design-reuse.com/articles/42344/hardwired-floating-point-fpga-based-ffts.html> (accessed on 11 July 2017).
35. Trader, T. Comparing Peak Floating Point Claims. Available online: <http://www.hpcwire.com/2014/06/17/comparing-peak-floating-point-claims/> (accessed on 22 January 2015).
36. ICCT The International Council on Clean Transportation—European Vehicle Market Statistics: Pocketbook 2014. Available online: [https://www.theicct.org/sites/default/files/publications/EU\\_pocketbook\\_2014.pdf](https://www.theicct.org/sites/default/files/publications/EU_pocketbook_2014.pdf) (accessed on 21 February 2019).
37. Gates, G. How Volkswagen’s ‘Defeat Devices’ Worked. *New York Times*, 10 December 2015.
38. Siampis, E.; Massaro, M.; Velenis, E. Electric rear axle torque vectoring for combined yaw stability and velocity control near the limit of handling. In Proceedings of the 52nd IEEE Conference on Decision and Control, Florence, Italy, 10–13 December 2013; pp. 1552–1557.
39. Parra, A.; Zubizarreta, A.; Pérez, J.; Dendaluze, M. Intelligent Torque Vectoring Approach for Electric Vehicles with Per-Wheel Motors. Available online: <https://www.hindawi.com/journals/complexity/2018/7030184/> (accessed on 1 October 2018).
40. Chevrolet Volt—Car and Driver. Available online: <http://www.caranddriver.com/chevrolet/volt> (accessed on 22 August 2017).
41. Gall, J. Car and Driver. Available online: <http://www.caranddriver.com/porsche/918> (accessed on 21 February 2019).
42. Mercedes-AMG Project One (2019): Vorschau—Über 1000 PS im AMG-Hypercar. Available online: <http://www.autobild.de/artikel/mercedes-amg-project-one-2019-vorschau-10641195.html> (accessed on 8 July 2017).
43. BMW i3—Car and Driver. Available online: <http://www.caranddriver.com/bmw/i3> (accessed on 22 August 2017).
44. Model S | Tesla. Available online: <https://www.tesla.com/models> (accessed on 22 August 2017).
45. Chilton, C. CAR Magazine UK. Available online: <http://www.carmagazine.co.uk/car-reviews/mercedes-benz/mercedes-sls-electric-drive-2013-review/> (accessed on 21 February 2019).
46. Concept\_One. Available online: [http://www.rimac-automobili.com/en/supercars/concept\\_one/](http://www.rimac-automobili.com/en/supercars/concept_one/) (accessed on 22 August 2017).
47. White, J. NextEV’s NIO EP9 is an Incredible Four-Wheel-Drive Electric Hypercar. Available online: <http://www.wired.co.uk/article/nextev-hypercar-nio-ep9> (accessed on 22 August 2017).
48. Roland Berger Strategy Consultants Consolidation in Vehicle Electronic Architectures. Available online: [https://www.rolandberger.com/media/pdf/Roland\\_Berger\\_TAB\\_Consolidation-in-vehicle-electronic-architectures\\_20150721.pdf](https://www.rolandberger.com/media/pdf/Roland_Berger_TAB_Consolidation-in-vehicle-electronic-architectures_20150721.pdf) (accessed on 21 February 2019).
49. PwC Automotive Perspective 2015. Available online: <https://www.pwc.com/gx/en/industries/automotive.html> (accessed on 21 February 2019).
50. ISO. ISO 26262-6:2018—Road Vehicles—Functional Safety—Part 6: Product Development at the Software Level. Available online: <https://www.iso.org/standard/68388.html> (accessed on 21 February 2019).
51. Hillenbrand, M.; Heinz, M.; Adler, N.; Matheis, J.; Müller-Glaser, K.D. Failure mode and effect analysis based on electric and electronic architectures of vehicles to support the safety lifecycle ISO/DIS 26262. In Proceedings of the 2010 21st IEEE International Symposium on Rapid System Prototyping, Fairfax, VA, USA, 8–11 June 2010; pp. 1–7.
52. The Mathworks Inc. Automotive Industry Standards—ISO 26262 Support in MATLAB and Simulink. Available online: <https://www.mathworks.com/solutions/automotive/standards/iso-26262.html> (accessed on 20 September 2017).

53. Dendaluce, M.; Gómez, V.; Irigoyen, E. Potential for applying Machine Learning in the context of Upcoming Automotive Technology. In Proceedings of the Symposium: XII Simposio CEA de Control Inteligente (SCI 2016), Gijón, Spain, 22–24 June 2016.
54. Dendaluce, M.; Iglesias, I.; Martin, A.; Prieto, P.; Peña, A. Race-Track Testing of a Torque Vectoring Algorithm on a Motor-in-Wheel Car Using a Model-Based Methodology with a HiL and multibody simulator setup. In Proceedings of the 19th International Conference on Intelligent Transportation Systems of the IEEE, Rio de Janeiro, Brazil, 1–4 November 2016.
55. Parra, A.; Dendaluce, M.; Zubizarreta, A.; Pérez, J. Novel Fuzzy Torque Vectoring Controller for Electric Vehicles with Per-Wheel Motors. In Proceedings of the Jornadas de Automática 2017, Gijón, Spain, 6–8 September 2017.
56. Heissing, B.; Ersoy, M. (Eds.) *Chassis Handbook: Fundamentals, Driving Dynamics, Components, Mechatronics, Perspectives*, ATZ, 1st ed.; Vieweg + Teubner: Wiesbaden, Germany, 2011; ISBN 978-3-8348-0994-0.
57. Pastorino, R.; Cosco, F.; Naets, F.; Desmet, W.; Cuadrado, J. Hard real-time multibody simulations using ARM-based embedded systems. *Multibody Syst. Dyn.* **2016**, *37*, 127–143. [[CrossRef](#)]
58. Connor, J.T.; Martin, R.D.; Atlas, L.E. Recurrent neural networks and robust time series prediction. *IEEE Trans. Neural Netw.* **1994**, *5*, 240–254. [[CrossRef](#)] [[PubMed](#)]
59. Sundermeyer, M.; Oparin, I.; Gauvain, J.-L.; Freiberger, B.; Schluter, R.; Ney, H. Comparison of feedforward and recurrent neural network language models. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 8430–8434.
60. Karim, M.N.; Rivera, S.L. Comparison of feed-forward and recurrent neural networks for bioprocess state estimation. *Comput. Chem. Eng.* **1992**, *16*, S369–S377. [[CrossRef](#)]
61. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)] [[PubMed](#)]
62. Tai, K.S.; Socher, R.; Manning, C.D. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. *arXiv*, 2015; arXiv:1503.00075 Cs.
63. Optimizing Recurrent Neural Networks in cuDNN 5. Available online: <https://devblogs.nvidia.com/optimizing-recurrent-neural-networks-cudnn-5/> (accessed on 28 September 2018).
64. Chui, C.K.; Chen, G. *Kalman Filtering: With Real-Time Applications*, 4th ed.; Springer: Berlin/Heidelberg, Germany, 2009; ISBN 978-3-642-09966-3.
65. Grewal, M.S. Kalman Filtering. In *International Encyclopedia of Statistical Science*; Springer: Berlin, Germany, 2011; pp. 705–708.
66. Doumiati, M.; Charara, A.; Victorino, A.; Lechner, D. *Vehicle Dynamics Estimation using Kalman Filtering: Experimental Validation*; John Wiley & Sons: Hoboken, NJ, USA, 2012; ISBN 978-1-118-57900-8.
67. Sammut, C.; Webb, G.I. *Encyclopedia of Machine Learning*; Springer: London, UK, 2010; ISBN 978-0-387-30768-8.
68. Alpaydin, E. *Introduction to Machine Learning*, 2nd ed.; The MIT Press: Cambridge, MA, USA, 2010; ISBN 978-0-262-01243-0.
69. Owens, J.D.; Houston, M.; Luebke, D.; Green, S.; Stone, J.E.; Phillips, J.C. GPU Computing. *Proc. IEEE* **2008**, *96*, 879–899. [[CrossRef](#)]
70. Owens, J.D.; Luebke, D.; Govindaraju, N.; Harris, M.; Krüger, J.; Lefohn, A.E.; Purcell, T.J. A Survey of General-Purpose Computation on Graphics Hardware. *Comput. Graph. Forum* **2007**, *26*, 80–113. [[CrossRef](#)]
71. Hu, L.; Che, X.; Zheng, S.-Q. A Closer Look at GPGPU. *ACM Comput. Surv.* **2016**, *48*, 60:1–60:20. [[CrossRef](#)]
72. Mittal, S.; Vetter, J.S. A Survey of CPU-GPU Heterogeneous Computing Techniques. *ACM Comput. Surv.* **2015**, *47*, 69:1–69:35. [[CrossRef](#)]
73. Youness, H.; Moness, M.; Shaaban, O.; Hussein, A.I. Accelerated Processing Unit (APU) potential: N-body simulation case study. In Proceedings of the 2016 11th International Conference on Computer Engineering Systems (ICCES), Cairo, Egypt, 20–21 December 2016; pp. 110–115.
74. Buonaiuti, N.; Louie, M.; Aarestad, J.; Mital, R.; Mateik, D.; Sivilli, R.; Bhopale, A.; Kief, C.; Zufelt, B. Satellite Identification Imaging for Small Satellites Using NVIDIA. In Proceedings of the AIAAUSU Conference Small Satell, Logan, UT, USA, 5–10 August 2017.
75. NVIDIA Corporation CUDA Toolkit Documentation. Available online: <http://docs.nvidia.com/cuda/> (accessed on 20 October 2017).
76. Kuon, I.; Tessier, R.; Rose, J. FPGA Architecture: Survey and Challenges. *Found. Trends® Electron. Des. Autom.* **2007**, *2*, 135–253.

77. GPU vs FPGA Performance Comparison. Available online: [http://www.bertendsp.com/pdf/whitepaper/BWP001\\_GPU\\_vs\\_FPGA\\_Performance\\_Comparison\\_v1.0.pdf](http://www.bertendsp.com/pdf/whitepaper/BWP001_GPU_vs_FPGA_Performance_Comparison_v1.0.pdf) (accessed on 21 February 2019).
78. Meyer-Baese, U. *Digital Signal Processing with Field Programmable Gate Arrays*; Signals and Communication Technology; Springer: Tallahassee, FL, USA, 2007; ISBN 978-1-4020-4817-3.
79. Ling, A.; Capalija, D.; Chiu, G. Accelerating Deep Learning with the OpenCL Platform and Intel Stratix 10 FPGAs. Available online: <https://builders.intel.com/docs/aibuilders/accelerating-deep-learning-with-the-openccl-platform-and-intel-stratix-10-fpgas.pdf> (accessed on 21 February 2019).
80. Fu, Y.; Wu, E.; Sirasao, A. *8-Bit Dot-Product Acceleration*; Xilinx Inc.: San Jose, CA, USA, 2017.
81. Xilinx Inc. Xilinx Design Tools. Available online: <https://www.xilinx.com/products/design-tools.html> (accessed on 10 October 2017).
82. Xilinx Inc. *LogiCORE IP Floating-Point Operator v6.1*; Xilinx Inc.: San Jose, CA, USA, 2012; p. 105.
83. Intel® Cyclone® 10 GX Device Overview—c10gx-51001.pdf. Available online: [https://www.altera.com/content/dam/altera-www/global/en\\_US/pdfs/literature/hb/cyclone-10/c10gx-51001.pdf](https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/cyclone-10/c10gx-51001.pdf) (accessed on 10 October 2017).
84. Pena, A.; Iglesias, I.; Valera, J.J.; Martin, A. Development and validation of Dynacar RT software, a new integrated solution for design of electric and hybrid vehicles. In Proceedings of the EVS26 International Battery, Hybrid and Fuel Cell Electric Vehicle Symposium, Los Angeles, CA, USA, 6–9 May 2012; pp. 1–7.
85. Dynacar by Tecnalia. Available online: <http://www.dynacar.es/en/home.php> (accessed on 20 July 2016).
86. Mathworks. Available online: <http://www.mathworks.com/> (accessed on 30 January 2015).



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# An Infinite-Norm Algorithm for Joystick Kinematic Control of Two-Wheeled Vehicles

Alejandro Said <sup>1,\*</sup>, Yasser Davizón <sup>2,3</sup>, Rogelio Soto <sup>4</sup>, Carlos Félix-Herrán <sup>5</sup>,  
Carlos Hernández-Santos <sup>6</sup> and Piero Espino-Román <sup>7</sup>

<sup>1</sup> Tecnológico de Monterrey, Campus Monterrey, Eugenio Garza Sada 2501, Monterrey 64849, Nuevo León, Mexico

<sup>2</sup> Department of Industrial Manufacturing and Systems Engineering, University of Texas, El Paso 500 West University Avenue, El Paso, TX 79968, USA; yadavizonca@miners.utep.edu

<sup>3</sup> Carretera a Pueblo Nuevo KM 3, Universidad Politécnica del Mar y la Sierra, Potrerillos Del Norte, Tayoltita 82740, Sinaloa, Mexico

<sup>4</sup> Tecnológico de Monterrey, Eugenio Garza Sada 2501, Monterrey 64849, Nuevo León, Mexico; rsoto@itesm.mx

<sup>5</sup> Escuela de Ingeniería y Ciencias, Instituto Tecnológico y de Estudios Superiores de Monterrey (ITESM), Enrique Mazón López 965, Hermosillo 83000, Sonora, Mexico; lcfelix@itesm.mx

<sup>6</sup> Department of Mechatronics, Instituto Tecnológico de Nuevo León, Eloy Cavazos 2001, Guadalupe 67170, Nuevo León, Mexico; carlos.hernandez@itnl.edu.mx

<sup>7</sup> Department of Mechatronics, Universidad Politécnica de Sinaloa (UPSIN), Carretera Municipal Libre Mazatlán Higuera Km 3, Mazatlán 82199, Sinaloa, Mexico; pespino@upsin.edu.mx

\* Correspondence: al.rodriguez.phd.mty@itesm.mx or alex.rsaid@gmail.com; Tel.: +52-1-811-599-2728

Received: 25 July 2018 ; Accepted: 23 August 2018 ; Published: 27 August 2018

**Abstract:** In this paper, we propose an algorithm based on the mathematical  $p$ -norm which has been applied to improve both the traction power and the trajectory smoothness of joystick-controlled two-wheeled vehicles. This algorithm can theoretically supply 100% of available power to each of the actuators if the infinity-norm is used, i.e., when the  $p$ -norm tends to infinity. Furthermore, a geometrical model using the radius of curvature has been developed to track the effect of the proposed algorithm on the vehicle's trajectory. Findings in this research work contribute to the kinematic control and path planning algorithms for vehicles actuated by two wheels, such as tanks and electric wheelchairs, both of vital importance for the security and health industry. Computer simulations and experiments with a real robot are performed to verify the results.

**Keywords:** joystick; two-wheeled; terrestrial vehicle; path planning; infinity norm;  $p$ -norm; kinematic control; navigation; actuation systems; maneuver algorithm

## 1. Introduction

In recent years, terrestrial vehicles and robots have gained research interest due to their potential use in military and rescue missions [1–3] as well as in medical applications to help the paraplegic people in their locomotion activities [4–10].

Although the terrestrial vehicles may have more than two wheels, in this work, our attention is directed to the ones which are actuated by only two wheels or caterpillars. Some examples are: the tanks used in the army and the electric wheelchairs, considered of great importance for the security and health fields, respectively.

Despite the use of simple levers as common control devices, lately, the attention has been turned to the use of the joystick as one of the preferred mechanisms to guide the trajectory of terrestrial vehicles. According to Hass [6], the dexterity of human hand allows for the joystick to be preferred as

an alternative ergonomic technology in tele-operation applications. The use of the joystick incorporates the separate functionality for each of the fingers at both hands, thus allowing multitasking and precision-demanding activities.

Evidently, controlling the traction power by using separate levers for each of the actuators, implying that the maximum available power of the wheels can be accessed by setting each of the levers to its maximum position. Furthermore, a turn trajectory can be achieved by setting the levers to different positions. However, if a joystick is used to guide the trajectory of two-wheeled vehicles, then, the power management relation is not straightforward; that is, the levers are now disposed in a perpendicular arrangement establishing a Cartesian configuration, so that not only the magnitude of each of the levers has an effect over the power supplied to the wheels, but also the angle which is formed with their X-Y displacement. In such a relation, a 100% power supply along with a smooth kinematic control can hardly be achieved. Prior state-of-the-art power management literature for vehicles can be found in [11–13].

In general, the kinematic motion control should be designed in such a way that it drives the vehicle up to its maximum possible speed and a rectilinear trajectory when the joystick is displaced to a maximum and pure forward position. In addition, the vehicle should execute a circular trajectory at its maximum speed by pivoting over one of its wheels when the joystick is directed completely towards a pure lateral position. The challenge here is to mathematically define the movement intention when the joystick is located somewhere else. At these locations, the wheels of the vehicle must rotate to the same direction but spin at different velocities.

The constraints imposed by the use of the joystick represent a critical situation because the vehicle's mobility could be negatively affected by the software when, theoretically, no mechanical limitations exist. This problem is solved in this research work by the use of the mathematical infinity norm which can, theoretically, provide for full traction power to each of the actuators. Furthermore, the proposed algorithm can also provide for a smooth trajectory modification in exchange for traction power. Additionally, a trajectory kinematic model based in the radius of curvature is developed, which can be used to analyze the smoothness performance when switching from a forward to a curved trajectory.

Being able to send all available power to each of the wheels can prevent a vehicle from getting stuck, potentially saving considerable amounts of money if human-assisted recovery is impossible, for example, in interplanetary operations. On the other hand, the ability to follow smooth trajectories is important in situations where expensive, dangerous or delicate loads are being transported.

Regarding the two-wheeled kinematics, Cooper [5] obtained a relation between the minimum and maximum radius of curvature and the target distance using the like-triangles geometry. Maulana [14] found the kinematic model for a two-wheeled differential drive robot, where the orientation of the robot is controlled using the speed differences between the two wheels.

In relation to joystick control algorithms, Fattouh [7] implemented a force-feedback approach to reduce collisions of people when using wheelchairs. Tsai [15] proposed a synchronized control scheme for the recovery of a dual-drive wheelchair trajectory when colliding with an obstacle or move across uneven surfaces. Rabhi used a Fuzzy-Logic controller [8] to provide for a smooth control of a wheelchair and react to obstacles. He also used neural networks [9] for joystick control of electric wheelchairs. Mrabet [16] used a recurrent neural network algorithm to design a controller that constantly corrects undesirable movements of the patient's hand and ensures a smooth and safe navigation for joystick-controlled wheelchairs.

With respect to infinity-norm solution, Yoon [17] proposed an algorithm using the infinity-norm to define the torque envelopes of the spacecraft reaction wheels. Hyun [18] introduced a framework for modeling the optimal path planning problem of rectangular robots for safe obstacle-avoiding paths by using the weighted  $L_p$  norm.

The procedures developed in this study may help to the development of mobility algorithms for odor-follower robots [19], commercial hoverboards and Segways [20]. They may become necessary in virtual reality applications [21], i.e., where a joystick is available but a physical robot is absent. They also may contribute to a better quality of life of people with severe mobility problems who can only move the some fingers, or even only the eyes [22]. In addition, if the joystick is substituted with a sensor network involving gyros, magnetometers or IMUs (disposed in a Cartesian configuration), then the algorithms presented here could be adapted for different applications such as drone and ship motion control for stabilization [23].

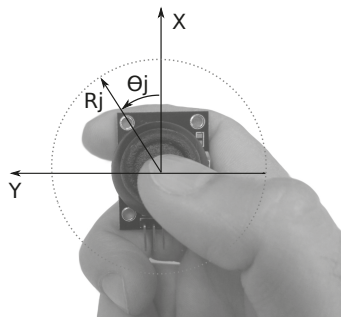
It is important to mention that, to simplify calculations, our study does not consider the backwards movement, which is left to future research. This fact prevents the two-wheeled model analyzed in this work from performing turns by pivoting at the center of the axis binding the two wheels. However, consider that the vehicle can always execute a  $180^\circ$  turn to go in the opposite direction when necessary.

The remainder of the manuscript is organized as follows: Section 2 presents background theory relevant to this study where the fundamentals of the joystick operation, the two-wheeled locomotion, and the mathematical norm are given. In Section 3, we present our proposed algorithm as well as the description of the experimentation platform. Section 4 illustrates the simulation and experimentation results. Section 5 points out some important conclusions, claims for the main contributions and encourages further research.

## 2. Background Theory

### 2.1. The Joystick Operation

Generally, joysticks consist of two potentiometers (pots), which are, in fact, variable resistors. Tilting the joystick along the vertical axis changes the resistance of a single pot. Tilting the joystick along the horizontal axis changes the resistance of the other pot [24]. To agree with the right-hand rule and the robot navigation convention followed by Rabhi [8,9], the X-axis and Y-axis directions are assumed to be directed as in Figure 1. Here,  $R_j$  and  $\theta_j$  refer to the displacement radius and the displacement angle of the joystick lever, respectively.



**Figure 1.** Geometrical parameters of the joystick.

It has been assumed that the signal given by the x-pot ranges from zero to one, while the signal given by the y-pot ranges from  $-1$  to  $1$ , i.e., the magnitude of joystick signals have been limited by software as in [4]. This means that the magnitude of vector  $R_j$  cannot be greater than unity, as indicated by the dotted circular line in Figure 1.

Because in this study the analysis of the motion is limited to be directed forwards (the backwards analysis is omitted), the x-component of  $R_j$  cannot take negative values. The direction for a positive increase in the variable  $\theta_j$  is indicated by the direction of the curved line with an arrow in Figure 1,

which starts at a value of  $\theta_j = 0$  when  $R_j$  is aligned with the positive direction of the unit vector of the X-axis.

2.2. Fundamentals of Two-Wheeled Terrestrial Locomotion

Commonly, by varying the velocity at each of the wheels, the vehicle is able to move forwards or backwards, as well as turning clockwise (CW) or counterclockwise (CCW). If the power at each of the wheels can be accessed individually by means of separate control levers, then the maximum available power for the vehicle can be reached by displacing each of the levers at a maximum forward position, thus making it possible for it to follow a forward trajectory at full speed (i.e., both of the wheels at a 100% setting). In contrast, if just one of the wheels is fed with 100% of the power and the other wheel is left powerless, then the vehicle follows a tight circular trajectory by pivoting over one of its wheels. The tightest circular trajectory is achieved when the vehicle performs a turn-in-place movement [25,26], i.e., when both wheels receive 100% of the power but they have opposite rotation directions. In this case, the vehicle performs a rotation by pivoting over the center of chassis along the axis which binds both of the wheels.

To achieve a smooth circular trajectory, an arbitrary power reference (different from zero) must be sent to each of the wheels, thus allowing for the vehicle to follow a wider circular trajectory.

Figure 2 shows a simplified diagram of a two-wheeled vehicle. Here,  $V_L$  and  $V_R$  refer to the magnitude of the tangential velocities of the left and right wheels, respectively; (i.e., the velocity of each of the wheels regarding the ground). The magnitude  $V$  is the tangential velocity of the center point between the two wheels attached to the chassis. In this study, the magnitude of the velocities are assumed to be normalized, and they are considered to be proportional to the power delivered to each of the wheels. Moreover, the constant  $r$  indicates the distance between the center of the chassis and any of the wheels. If the magnitude  $V$  is needed, it can be obtained by averaging  $V_L$  and  $V_R$ , as shown by Mulana [14].

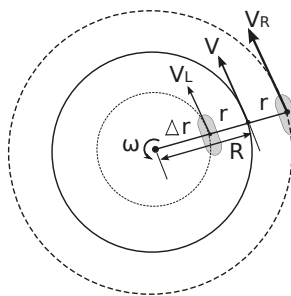


Figure 2. Geometrical disposal the variables used in the two-wheeled simplified model.

Figure 2 and Equation (1) show that the magnitude of the radius of curvature  $R$  comprises the length from the center of one the wheels to the center of the chassis  $r$ , with the distance from the rotation pivot to the center of the nearest wheel  $\Delta r$ . It is assumed that the angular velocity  $\omega$  and the radius of curvature  $R$  have a positive direction when the vehicle is turning CCW [14]. A particular case where the vehicle is turning CCW (using the left wheel as pivot) can be seen in Figure 3. Here, the radius of curvature  $R$  equals the distance  $r$ .



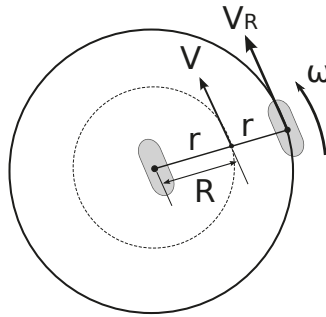


Figure 3. Performing a CCW turn with the left wheel as pivot.

Notice that variable  $\omega$  refers to the angular velocity of the vehicle regarding the ground, not the angular velocity of a wheel. In addition, consider that variables  $V_L$ ,  $V_R$ ,  $V$ ,  $R$ ,  $\Delta_r$  and  $\omega$  depend on the joystick coordinates provided by the user; thus, they are functions of time. Similar schemes can be found in [27,28]; however, the proposed model prioritize the radius of curvature analysis as demonstrated in the following sections.

$$R = r + \Delta_r \tag{1}$$

### 2.3. Fundamentals of the Mathematical Norm

The mathematical two-norm (also called Euclidean norm) is well-known in basic mathematics and it is commonly associated with the calculus of minimum distances; for example, it can be used for estimating the magnitude of the hypotenuse of a right triangle if the length of the perpendicular sides are known. However, there exist in robotics problems described by an infinite number of solutions, which require a particular one to be selected for some type of optimization criterion. One possibility employs a type of optimization which minimizes the maximum magnitude of the solutions. This is the *infinity-norm* solution, also known as the *minimum effort solution* [29]. The general form of the mathematical norm is called the *weighted  $L_p$  norm*, which has been used in robotics in [18]. When setting  $\sigma = 1$ , we have the non-weighted case; in addition, when  $p = 2$ , the Euclidean norm is obtained. If  $\sigma = 1$  and  $p$  approaches infinity, then the mathematical norm approaches the infinity norm which is also equal to the maximum absolute value within the elements of  $\mathbf{x}$ .

$$\|\mathbf{x}\|_{(p,\sigma)} = \left( \sum_{i=1}^n (|x_i|/\sigma_i)^p \right)^{1/p} \tag{2}$$

## 3. The Mathematical Model for Two-Wheeled Locomotion

### 3.1. An Infinity-Norm Approach Applied to Joystick Kinematic Control

In two-wheeled vehicle joystick control, the goal is to transform a Cartesian coordinate; (i.e., a joystick position) into a motor actuation for each of the wheels.

By using the weighted mathematical norm, we are able to minimize the maximum value of the joystick coordinates when  $p$  approaches infinity. In addition, by choosing another value of  $p$ , we can vary the behavior of the motor actuation. Here,  $\sigma$ , has been chosen to be a constant for a particular value of  $p$  in order to make the motor command to be normalized to be suitable of being implemented in a *Pulse Width Modulation Technique* [30,31].

Theoretically, the power loss can be reduced to zero by using a normalized formula if an infinity-norm is used in the numerator along with an infinity root in the denominator (recall that  $\sqrt[p]{a/b} = \sqrt[p]{a}/\sqrt[p]{b}$ ), as shown in Equations (3) and (4). However, it is important to consider that the

processor memory will be overloaded when using infinite exponentials; instead, they have to be replaced by large exponentials. Here,  $x$  and  $y$  refer to the joystick position coordinates, and  $V_R$  and  $V_L$  are velocity commands which have been normalized by the structure of the formulas. Because in this particular study  $x$  remains positive, the absolute value bars are not needed for this coordinate. The form of the velocity command which does not make use of the  $p$ -norm was selected so that both of the wheels have the same value in a pure forward trajectory.

$$\text{for } y \geq 0 \left\{ \begin{array}{l} V_R = \lim_{p \rightarrow \infty} \sqrt[p]{\frac{x^p + |y|^p}{2}} \\ V_L = \lim_{p \rightarrow \infty} \frac{x}{\sqrt[p]{2}} \end{array} \right. \tag{3}$$

$$\text{for } y < 0 \left\{ \begin{array}{l} V_R = \lim_{p \rightarrow \infty} \frac{x}{\sqrt[p]{2}} \\ V_L = \lim_{p \rightarrow \infty} \sqrt[p]{\frac{x^p + |y|^p}{2}} \end{array} \right. \tag{4}$$

Table 1 shows the maximum normalized velocities in the forward, CW and CCW trajectories for the left and right wheels as  $p$  is varied. In Table 1, it can be seen how the normalized velocity increases (and, consequently, the power) when  $p$  becomes greater. In the same table, it can be observed that, when using low-degree  $p$  exponentials, most of the power is lost by software. For example, when  $p = 1$ , only 50% of the power can be supplied to the wheels, whereas, when using  $p = 2$ , only 70.7% of the total power can be accessed. It can also be observed that, in the forward trajectory, both wheels receive the same amount of power, while, for the CW and CCW trajectories, the pivot wheel remains static, i.e., with zero velocity.

Table 1. Maximum normalized velocities.

$p$	Forward ( $x = 1, y = 0$ )		CW ( $x = 0, y = -1$ )		CCW ( $x = 0, y = 1$ )	
	$V_L$	$V_R$	$V_L$	$V_R$	$V_L$	$V_R$
1/2	0.25	0.25	0.25	0	0	0.25
1	0.5	0.5	0.5	0	0	0.5
2	0.707	0.707	0.707	0	0	0.707
3	0.793	0.793	0.793	0	0	0.793
5	0.87	0.87	0.87	0	0	0.87
8	0.917	0.917	0.917	0	0	0.917
20	0.965	0.965	0.965	0	0	0.965
100	0.993	0.993	0.993	0	0	0.993

### 3.2. Analysis of the Radius of Curvature

In this subsection, a mathematical model is developed to analyze the change in trajectory of two-wheel vehicles when they are commanded by a joystick. The method is based in the radius of curvature traced by the vehicle when it navigates.

In Figure 2, it can be seen that:

$$V_R = \omega(2r + \Delta_r) \tag{5}$$

and

$$V_L = \omega(\Delta_r) \tag{6}$$

Because  $\omega$  is equal for both wheels (including the chassis), we can substitute Equation (6) into Equation (5) to have Equation (7).

$$V_R = \frac{V_L}{\Delta r}(2r + \Delta r) \quad (7)$$

Substituting Equation (1) into Equation (7), we finally come to Equation (8), which provides the radius of curvature related to the left and right wheel velocities. As indicated by Chwa [32], because  $V_R$  and  $V_L$  are assumed to be normalized, their real values are not known beforehand, and thus, they must be estimated later in the design of the controller.

$$R = \frac{2rV_L}{V_R - V_L} + r \quad (8)$$

From Equation (8), it can be seen that, when both velocities are equal, the denominator in the first term tends to zero, making the equation to approach infinity. Here, according to Cooper [5], a circular trajectory with a infinite radius of curvature means that, in fact, a rectilinear trajectory is being followed. If the vehicle performs a counterclockwise turn using the left wheel as pivot, then  $V_L = 0$  resulting in  $R = r$ , i.e., the radius of curvature is positive and it has a magnitude equal to the distance between the center of chassis and the left wheel (as shown in Figure 3). On the other hand, if the right wheel is taken as pivot, then  $V_R = 0$  resulting in  $R = -r$ , that is, the vehicle is performing a clockwise turn with a radius of curvature equal to the magnitude of  $r$ .

According to Maulana [14], the velocity of the center of the chassis  $V$  can be obtained using Equation (9).

$$V = \frac{V_R + V_L}{2} \quad (9)$$

According to Nunes [10], Equation (10) gives the estimation of the rotation angle by integrating the angular velocity over time.

$$\theta = \int_t \omega dt \quad (10)$$

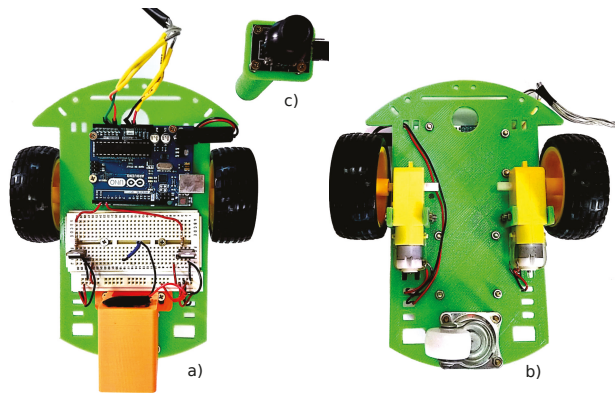
Substituting Equation (9) into Equation (10), we have Equation (11), which can be used by position and orientation estimation algorithms.

$$\theta = \int_t \frac{V}{R} dt \quad (11)$$

### 3.3. Experimentation Platform

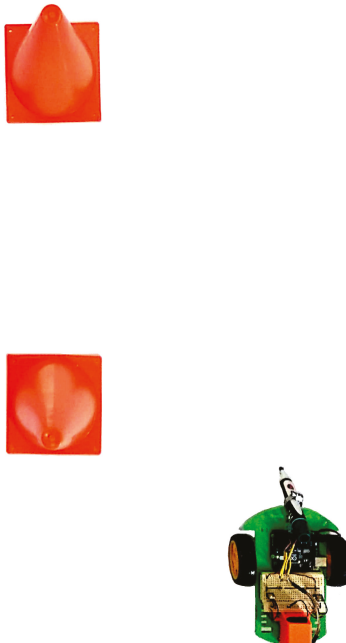
To carry out the generation of trajectories, a tow-wheeled actuated robot was created using the 3D-printing technology (see Figure 4). An un-actuated third free-wheel was included only to provide support. The robot consists of an Arduino UNO micro-controller card, a joystick, a 9Vdc power supply, two Tip-41 BJT transistors, two 330  $\Omega$  resistors, two DC motors and two diodes. The joystick is provided with Vcc and GND terminals and the x-pot and the y-pot are connected to the analog ports A0 and A1 respectively. The digital ports 2 and 3 are connected to the base of each of the transistors to provide for a PWM signal to the motors. The diodes are connected in an anti-parallel fashion along the motor terminals to serve as inductive snubbers.

Inside the program, the signals coming from the potentiometers of the joystick are normalized and the velocity commands are calculated using Equations (3) and (4). Then, a PWM signal with a switching frequency of 50 ms and a duty cycle proportional to the calculated velocity magnitudes is provided to the DC motors.



**Figure 4.** (a) Front; and (b) reverse sides of the real experimentation platform; and (c) the joystick controller.

The experimentation environment consists of two cones used as obstacles. The initial position of the robot is located at the lower right corner (see Figure 5), and the final position is located at the upper left corner. An erasable color marker is attached to the robot while it is guided by means of the joystick from the initial to the final position. This way, the trajectory is drawn over a white board.



**Figure 5.** Initial position of the robot at the experimentation environment.

## 4. Results

### 4.1. Simulation Results

Computer simulations were performed to know the behavior of the proposed joystick algorithm when used in the two-wheeled terrestrial locomotion; specifically, how the joystick coordinates affect the radius of curvature in Equations (3) and (4) as  $p$  increases. The numerical software package used in this work is *SciLab* [33].

Figure 6 shows a mapping of the radius of curvature when  $R_j = 1$  and  $\Theta_j$  moves from  $-90^\circ$  to  $90^\circ$  when considering different values of  $p$ . For simulation purposes the distance from one of the wheels to the center of the chassis was considered to be equal to 1 meter; i.e.,  $r = 1$ . Figure 6a–d shows  $p$  equal to one-half, one, two, and five, respectively.

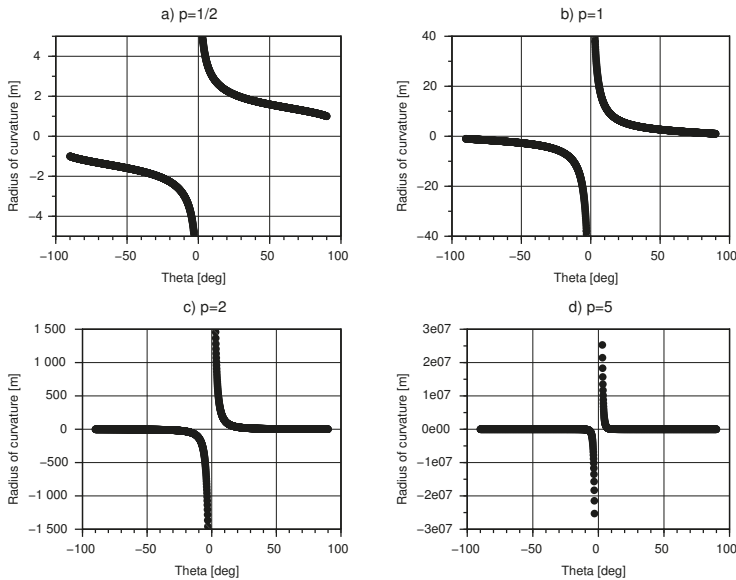


Figure 6. Effect of the increase of parameter  $p$  in the radius of curvature.

If the angle of the joystick is positive, then the radius of curvature is also positive because the vehicle is moving CCW. In contrast, if the angle of the joystick is negative, then the radius of curvature is also negative because the vehicle is moving CW.

Figure 6 shows that, when the  $\Theta_j$  approximates to zero degrees (i.e., the joystick displacement is aligned with the X-axis), the radius of curvature tends to infinity. Both positive and negative radii of curvature imply that a forward trajectory is being followed; however, in the case of an infinite negative radius of curvature, the theoretical pivot from which the radius of curvature is measured is located at the right side of the vehicle.

In Figure 6, it is important to notice that, despite the increase in traction power when  $p$  is incremented, the vehicle loses its capacity to follow a soft turn when switching from a forward trajectory to a circular one, i.e., either the vehicle tends to go in a forward direction (for values of  $\Theta_j$  close to  $0^\circ$ ), or to follow a pivoted turn. On the other hand, when  $p$  is decreased, then the ability to a perform smooth trajectory is incremented; however, the traction power capability decreases.

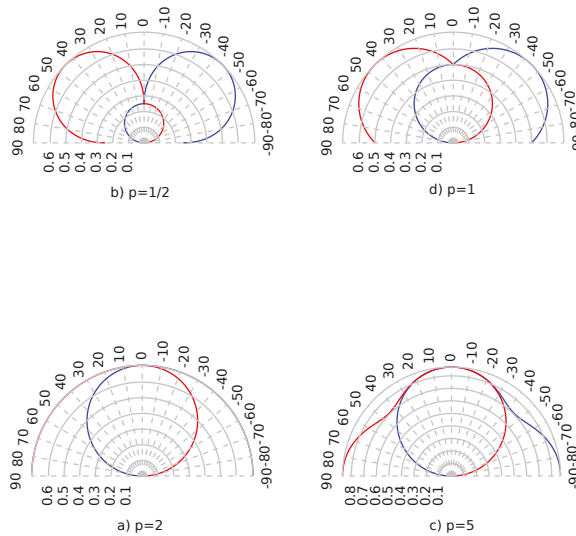
Table 2 shows the radii of curvature for different values of  $p$  when  $R_j = 1$ . In this table, it is easy to see that, if the angle of the joystick  $\Theta_j$  has a value of  $90^\circ$  (a CCW turn is commanded), then the radius

of curvature  $R$  has the same dimension as the distance from the left wheel to the center of the chassis, i.e.,  $R = r = 1$ . A similar situation arises when  $\Theta_j = -90$ , where the vehicle is turning clockwise and  $R = -1$ .

**Table 2.** Radius of curvature  $R$  for different joystick angles when  $R_j = 1$ .

$\Theta_j$	$p = 1/2$	$p = 1$	$p = 2$	$p = 5$
$-90^\circ$	-1	-1	-1	-1
$-60^\circ$	-1.45	-2.15	-3	-3.65
$-30^\circ$	-1.95	-4.46	-13.92	-160.83
$-3^\circ$	-4.91	-39.16	-1458.35	-25,294,220
$3^\circ$	4.91	39.16	1458.35	25,294,220
$30^\circ$	1.95	4.46	13.92	160.83
$60^\circ$	1.45	2.15	3	3.65
$90^\circ$	1	1	1	1

Figure 7 shows the polar plots of the normalized velocities of the left wheel  $V_L$  (in color blue) and the right wheel  $V_R$  (in color red) regarding the angle of the joystick  $\Theta_j$  when  $R_j = 1$ . The data in Table 1 can be corroborated in Figure 7. For example, it can be seen that, if the angle of the joystick is  $0^\circ$ , then the magnitude of the normalized velocities  $V_L$  and  $V_R$  for  $p = 1/2$  are 0.25; similarly, for  $p = 1$ ,  $V_L = V_R = 0.5$ .



**Figure 7.** Polar plots of the magnitude of  $V_L$  (in blue) and the magnitude of  $V_R$  (in red) regarding the angle  $\Theta_j$ .

#### 4.2. Experimental Results

Figure 8 shows the generated trajectories using different values for parameter  $p$ . It can be observed that, as predicted by Equations (3) and (4), a softer trajectory is produced for small values of  $p$ , i.e., the trajectory adjustments produced by the joystick can be finer. As parameter  $p$  increases, more power is delivered to the wheels, and thus, the acceleration is greater and longer traces are produced as a consequence.

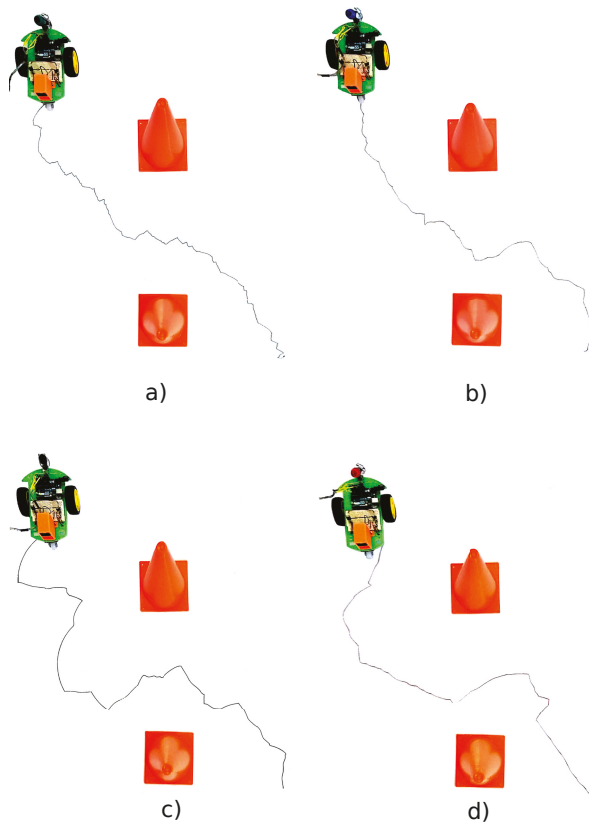


Figure 8. Generated trajectories using: (a)  $p = 1/2$ ; (b)  $p = 1$ ; (c)  $p = 2$ ; and (d)  $p = 5$ .

As shown in Figure 6, a trajectory can be considered soft when the robot is capable of following turn commands. If we assume real tangential velocities to be  $k \cdot V_R$  and  $k \cdot V_L$  for the right and the left wheels (where the constant  $k$  is an unknown proportional parameter), and if we substitute these values into Equation (8) instead of the terms  $V_R$  and  $V_L$ , then the parameter  $k$  is eliminated; thus, the radius of curvature described by Equation (8) only depends on the distance  $r$  and the normalized velocities  $V_R$  and  $V_L$ . Taking advantage of this fact and considering  $r = 6.5$  cm for the robot shown in Figure 8, Table 3 shows the radius of curvature for a joystick angle of  $\Theta_j = 45^\circ$  and a magnitude of  $R_j = 1$  (i.e.,  $x = 0.707$  and  $y = 0.707$ ). In this table, it can be observed that, despite the joystick intension for the robot to perform a turn, the robot tries to follow a rectilinear trajectory as the parameter  $p$  is increased because both velocities ( $V_R$  and  $V_L$ ) tend to be equal.

Table 3. Radius of curvature  $R$  when the angle and the magnitude of the joystick are  $\theta_j = 45^\circ$  and  $R_j = 1$ , respectively, when parameter  $p$  is increased.

$p$	$V_R$	$V_L$	$R$ [cm]
1/2	0.707	0.176	23.8080
1	0.707	0.353	32.4630
2	0.707	0.499	50.6870
5	0.707	0.615	106.402

Regarding the power management, Table 4 shows how the wheels receive more power as parameter  $p$  is increased. The voltage across the terminals of one of the wheels was measured when the joystick is completely displaced towards the positive direction of the X-axis. Because joystick measurements are normalized, in this case  $\Theta_j = 0$  and  $R_j = 1$ , i.e., both wheels receive the same amount of power. The current was measured in series with the battery, so by neglecting the power consumed by the microcontroller, the power measurement shown in last column of Table 4 corresponds to the two motors. Figure 9 shows the case when  $p = 1$ ; here, a voltage of 6.83 V times a current of 0.32 A results in 2.18 watts.

**Table 4.** Electric power (measured in watts) delivered to one of the wheels when the joystick is displaced completely in the positive direction of the X-axis.

$p$	Voltage [V]	Current [A]	Power [W]
1/2	5.50	0.28	1.54
1	6.83	0.32	2.18
2	7.20	0.34	2.44
5	8.00	0.37	2.96



**Figure 9.** Electric power delivered to one of the wheels when  $p = 1$  and the joystick is displaced completely in the positive direction of the X-axis.

## 5. Conclusions and Future Work

In the present study, we can conclude that a joystick algorithm may have an important effect in the trajectory smoothness and traction power of two-wheeled vehicles. In our particular case, despite the increment in traction power as  $p$  increased, the vehicle became less sensitive at executing smooth trajectory modifications. However, because the variable term  $p$  is not fixed, it can be set to a low degree to achieve smooth trajectory changes by ceding traction power.

In this work, a mathematical model based in the radius of curvature has been developed to analyze the effect of the proposed joystick algorithm regarding the two-wheeled locomotion. Computer simulations were performed to show the effect of varying the parameter  $p$ . An important



observation is that, for our particular case, the infinite radii of curvature, whether positive or negative are part of the same forward trajectory. Another consequence of this study is that, if a position and orientation analysis is needed, an integral analysis over the tangential velocity of the center of the chassis and the radius of curvature can be performed, provided that the left and the right tangential velocities of the wheels  $V_L$  and  $V_R$  are known, which is possible by using the algorithms proposed in this work. Regarding the experimentation results, it was confirmed that smaller values of parameter  $p$  produced finer trajectories but subtracted wheel power. In contrast, increasing values of  $p$  produced greater traction power, and, thus, larger robot acceleration but less maneuverability. However, this parameter could be adjusted according to specific required needs. Surely, circumstances such as wheel-to-ground slip, mechanical-part friction, power supply charge, human expertise, and robot architecture have a decisive impact over the experiment results.

As future work, the following opportunity areas remain: The study of the backwards movement and the inclusion of a regulatory controller (i.e., PID, optimal, state-feedback, sliding mode, etc.). The regulatory controller should correct the position and the orientation as well as work along with the proposed kinematic control. Here, it is believed that the knowledge we now have in advance about the vehicle's trajectory regarding the wheel velocities should minimize the manipulation action.

**Author Contributions:** A.S. conceived the idea, implemented the mathematical procedures, performed the simulations, and wrote the paper. Y.D. reviewed the mathematical form of the equations and looked for a suitable journal. R.S. and L.C.F. analyzed and reviewed the paper structure and validated the data. C.H.-S. and P.E.-R. carried out the literature review.

**Funding:** This research work received no funding.

**Acknowledgments:** The authors would like to thank the support for the open-access publishing costs covered by the Center for Robotics and Intelligent Systems (CRIS) and the National Laboratory for Robotics CONACyT-ITESM, located at Instituto Tecnológico y de Estudios Superiores de Monterrey (ITESM); Eugenio Garza Sada 2501, Monterrey, N.L., México, Zip Code 64849.

**Conflicts of Interest:** The authors declare no conflict of interests.

## References

1. Zhao, H.; Duan, X.; Yang, G. Kinematics and dynamics modeling of a small mobile robot with tracked locomotion mode. In Proceedings of the 2010 IEEE International Conference on Robotics and Biomimetics (ROBIO), Tianjin, China, 14–18 December 2010; pp. 1276–1280.
2. Fujiya, T.; Mikami, S.; Nakamura, T.; Hama, K. Locomotion method of a rescue robot with multi-legs and Omni-directional wheels. In Proceedings of the 2014 13th International Conference on Control Automation Robotics & Vision (ICARCV), Singapore, 10–12 December 2014; pp. 1627–1630.
3. Wang, X.; Wang, X.; Wang, H.; Lu, L.; Yu, H.; Vladareanu, L.; Melinte, D.O. Dynamic analysis for the leg mechanism of a wheel-leg hybrid rescue robot. In Proceedings of the 2014 UKACC International Conference on Control (CONTROL), Loughborough, UK, 9–11 July 2014; pp. 504–508.
4. Van Sickle, D.; Cooper, R.; Ster, J. Wheelchair virtual joystick interface. In Proceedings of the IEEE 17th Annual Conference on Engineering in Medicine and Biology Society, Montreal, QC, Canada, 20–23 September 1995; Volume 2, pp. 1175–1176.
5. Cooper, R.A.; Jones, D.K.; Fitzgerald, S.; Boninger, M.L.; Albright, S.J. Analysis of position and isometric joysticks for powered wheelchair driving. *IEEE Trans. Biomed. Eng.* **2000**, *47*, 902–910. [[CrossRef](#)] [[PubMed](#)]
6. Haas, E.C.; Kunze, M. The effect of a vehicle control device on driver performance in a simulated tank driving task. In Proceedings of the First International Driving Symposium on Human Factors in Driving Assessment, Training and Vehicle Design, Aspen, CO, USA, 14–17 August 2001; pp. 143–146.
7. Fattouh, A.; Sahnoun, M.; Bourhis, G. Force feedback joystick control of a powered wheelchair: Preliminary study. In Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics, The Hague, The Netherlands, 10–13 October 2004; Volume 3, pp. 2640–2645.
8. Rabhi, Y.; Mrabet, M.; Fnaiech, F.; Gorce, P. Intelligent joystick for controlling power wheelchair navigation. In Proceedings of the 2013 3rd International Conference on Systems and Control (ICSC), Algiers, Algeria, 29–31 October 2013; pp. 1020–1025.

9. Rabhi, Y.; Mrabet, M.; Fnaiech, F. Optimized joystick control interface for electric powered wheelchairs. In Proceedings of the 2015 16th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA), Monastir, Tunisia, 21–23 December 2015; pp. 201–206.
10. Nunes, W.R.B.M.; da Silva, N.; Covacic, M.R.; Junior, A.P.L. 3ph High Efficiency Induction Motors with IFOC Applied to a Wheelchair by Joystick. *IEEE Latin Am. Trans.* **2016**, *14*, 2041–2051. [[CrossRef](#)]
11. Xia, C.; Zhang, C. Power management strategy of hybrid electric vehicles based on quadratic performance index. *Energies* **2015**, *8*, 12458–12473. [[CrossRef](#)]
12. Ali, A.M.; Söffker, D. Towards Optimal Power Management of Hybrid Electric Vehicles in Real-Time: A Review on Methods, Challenges, and State-Of-The-Art Solutions. *Energies* **2018**, *11*, 476.
13. Zhang, X.; Mi, C. *Vehicle Power Management: Modeling, Control and Optimization*; Springer Science & Business Media: London, UK, 2011.
14. Maulana, E.; Muslim, M.A.; Zainuri, A. Inverse kinematics of a two-wheeled differential drive an autonomous mobile robot. In Proceedings of the Electrical Power, Electronics, Communications, Controls and Informatics Seminar (EECCIS), Malang, Indonesia, 27–28 August 2014; pp. 93–98.
15. Tsai, M.C.; Hsueh, P.W. Synchronized motion control for 2D joystick-based electric wheelchair driven by two wheel motors. In Proceedings of the 2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Kaohsiung, Taiwan, 11–14 July 2012; pp. 702–707.
16. Rabhi, Y.; Mrabet, M.; Fnaiech, F. Development of a New Intelligent Joystick for People with Reduced Mobility. *Appl. Bionics Biomech.* **2018**, *2018*. [[CrossRef](#)]
17. Yoon, H.; Seo, H.H.; Park, Y.W.; Choi, H.T. A new minimum infinity-norm solution: With application to capacity analysis of spacecraft reaction wheels. In Proceedings of the American Control Conference (ACC), Chicago, IL, USA, 1–3 July 2015; pp. 1241–1245.
18. Hyun, N.S.P.; Vela, P.A.; Verriest, E.I. A New Framework for Optimal Path Planning of Rectangular Robots Using a Weighted  $L_p$  Norm. *IEEE Robotics Autom. Lett.* **2017**, *2*, 1460–1465. [[CrossRef](#)]
19. Villarreal, B.L.; Gordillo, J.L. Directional aptitude analysis in odor source localization techniques for rescue robots applications. In Proceedings of the 2011 10th Mexican International Conference on Artificial Intelligence (MICAI), Aguascalientes, Mexico, 4–10 November 2011; pp. 109–114.
20. Babazadeh, R.; Khiabani, A.G.; Azmi, H. Optimal control of Segway personal transporter. In Proceedings of the 2016 4th International Conference on Control, Instrumentation, and Automation (ICCIA), Qazvin, Iran, 27–28 January 2016; pp. 18–22.
21. Li, W.; Liu, B.; Kosasih, P.B.; Zhang, X. A 2-DOF MR actuator joystick for virtual reality applications. *Sens. Actuators A Phys.* **2007**, *137*, 308–320. [[CrossRef](#)]
22. Nasif, S.; Khan, M.A.G. Wireless head gesture controlled wheel chair for disable persons. In Proceedings of the Humanitarian Technology Conference (R10-HTC), 2017 IEEE Region 10, Dhaka, Bangladesh, 21–23 December 2017; pp. 156–161.
23. Fortuna, L.; Muscato, G. A roll stabilization system for a monohull ship: modeling, identification, and adaptive control. *IEEE Trans. Control Syst. Technol.* **1996**, *4*, 18–28. [[CrossRef](#)]
24. Karvinen, T.; Karvinen, K.; Valtokari, V. *Make: Sensors. Projects and Experiments to Measure the World with Arduino and Raspberry Pi*; Maker Media: Sebastopol, CA, USA, 2014.
25. Abdolmaleki, A.; GhasemAghaee, N.; Reza, M.; Monadjemi, A. Robust humanoid turning-in-place using fourier series and genetic algorithm. In Proceedings of the 2011 4th International Conference on Modeling, Simulation and Applied Optimization (ICMSAO), Kuala Lumpur, Malaysia, 19–21 April 2011; pp. 1–5.
26. Said, A.; Rodriguez-Leal, E.; Soto, R.; Gordillo, J.; Garrido, L. Decoupled closed-form solution for humanoid lower limb kinematics. *Math. Prob. Eng.* **2015**, *2015*. [[CrossRef](#)]
27. Jia, Q.; Cao, X.; Sun, H.; Song, J. A novel design of a two-wheeled robot. In Proceedings of the Conference on Industrial Electronics and Applications, ICIEA 2007, Harbin, China, 23–25 May 2007; pp. 1226–1231.
28. Cho, S.K.; Jin, H.Z.; Lee, J.M.; Yao, B. Teleoperation of a mobile robot using a force-reflection joystick with sensing mechanism of rotating magnetic field. *IEEE/ASME Trans. Mech.* **2010**, *15*, 17–26.
29. Gravagne, I.A.; Walker, I.D. On the structure of minimum effort solutions with application to kinematic redundancy resolution. *IEEE Trans. Robot. Autom.* **2000**, *16*, 855–863. [[CrossRef](#)]
30. Mohan, N.; Undeland, T.M. *Power Electronics: Converters, Applications, and Design*; John Wiley & Sons: Hoboken, NJ, USA, 2007.

31. Said, A.; Davizón, Y.A.; Espino-Román, P.; Rodríguez-Said, R.; Hernández-Santos, C. Automatic Frequency Identification under Sample Loss in Sinusoidal Pulse Width Modulation Signals Using an Iterative Autocorrelation Algorithm. *Symmetry* **2016**, *8*, 78. [[CrossRef](#)]
32. Chwa, D. Robust Distance-Based Tracking Control of Wheeled Mobile Robots Using Vision Sensors in the Presence of Kinematic Disturbances. *IEEE Trans. Ind. Electron.* **2016**, *63*, 6172–6183. [[CrossRef](#)]
33. Group, E. SciLab [Numeric Calculus Software]. Available online: <http://www.scilab.org/> (accessed on 15 July 2018).



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# Coupled-Region Visual Tracking Formulation Based on a Discriminative Correlation Filter Bank

Jian Wei <sup>1,2</sup> and Feng Liu <sup>1,2,\*</sup>

<sup>1</sup> College of Education Science and Technology, Nanjing University of Posts and Telecommunications, Nanjing 210003, China; tdweijian@njupt.edu.cn

<sup>2</sup> Jiangsu Province Key Lab on Image Processing and Image Communications, Nanjing University of Posts and Telecommunications, Nanjing 210003, China

\* Correspondence: liuf@njupt.edu.cn; Tel.: +86-025-8586-6736

Received: 22 August 2018; Accepted: 6 October 2018; Published: 11 October 2018

**Abstract:** The visual tracking algorithm based on discriminative correlation filter (DCF) has shown excellent performance in recent years, especially as the higher tracking speed meets the real-time requirement of object tracking. However, when the target is partially occluded, the traditional single discriminative correlation filter will not be able to effectively learn information reliability, resulting in tracker drift and even failure. To address this issue, this paper proposes a novel tracking-by-detection framework, which uses multiple discriminative correlation filters called discriminative correlation filter bank (DCFB), corresponding to different target sub-regions and global region patches to combine and optimize the final correlation output in the frequency domain. In tracking, the sub-region patches are zero-padded to the same size as the global target region, which can effectively avoid noise aliasing during correlation operation, thereby improving the robustness of the discriminative correlation filter. Considering that the sub-region target motion model is constrained by the global target region, adding the global region appearance model to our framework will completely preserve the intrinsic structure of the target, thus effectively utilizing the discriminative information of the visible sub-region to mitigate tracker drift when partial occlusion occurs. In addition, an adaptive scale estimation scheme is incorporated into our algorithm to make the tracker more robust against potential challenging attributes. The experimental results from the OTB-2015 and VOT-2015 datasets demonstrate that our method performs favorably compared with several state-of-the-art trackers.

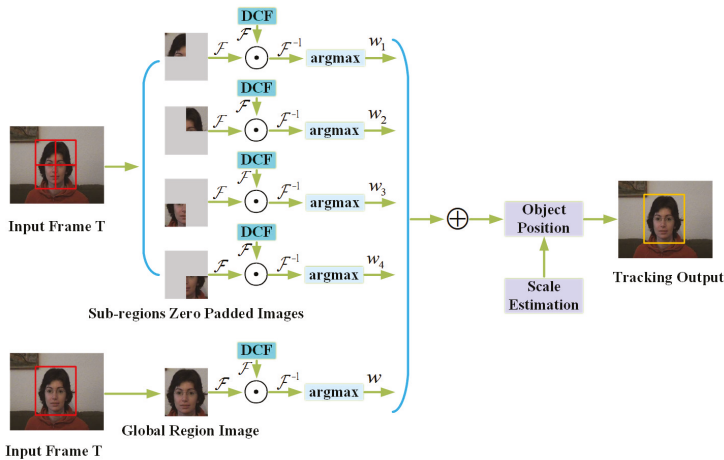
**Keywords:** visual tracking; discriminative correlation filter bank; occlusion; sub-region; global region

## 1. Introduction

Visual tracking plays an important role in computer vision, with numerous applications in areas such as robotics, human behavior analysis, intelligent traffic monitoring, and many more [1]. In recent years, numerous excellent tracking algorithms have emerged, but there are still some challenges that need to be addressed due to the practical complex background, such as illumination variation, scale variation, and occlusion. To solve the troubles caused by these challenges, the proposed trackers are generally divided into two categories: generative and discriminative methods. Generative trackers [2–4] perform tracking by searching for patches most similar to the target. Conversely, discriminative trackers [5–9] perform tracking by separating the target from the background.

Recently, the existing correlation filter tracking algorithms [10–30] have demonstrated superior performance in terms of speed and robustness. The main idea of the correlation filter-based tracking method is that the correlation output of each interested target is a correlation peak in the image sequence, while other background regions have a low correlation response, and thus the target is positioned in a new frame by the coordinates of the maximum correlation peak. According to the

convolution theorem, the correlation in the time domain corresponds to the element-wise multiplication in the frequency domain. Therefore, the essential idea of the high-speed correlation filter calculation is that it can be effectively calculated by fast Fourier transformation (FFT) and pointwise operations in the frequency domain. Thus, the time-consuming process of the convolution operation is effectively avoided. Based on this principle, the correlation filter tracking framework can meet the requirements of real-time tracking. Nevertheless, empirical experiments show that the sensitivity of the correlation filter when encountering challenging occlusion scenarios and the appearance of the target changes irregularly in tracking, which are easy to cause tracker drift. To deal with these issues, in this work, we formulate multiple discriminative correlation filters called discriminative correlation filter bank (DCFB) for visual tracking to solve the drifting problem caused by occlusion. Figure 1 demonstrates the overview of our formulation.



**Figure 1.** Overview of our algorithm. In the  $t$ -th frame, the sliding window obtains the sub-region and global region images, and the sub-region images are zero-padded to the same size as the global image. Subsequently, the correlation operation is performed with the trained DCFB in the frequency domain, and the position corresponding to the maximum correlation response is weighted to make the joint optimization the final target position. In addition, accurate scale estimation makes the tracking process more robust.

Our method combines the appearance models of multiple sub-regions and the global target region for the motion model, and not only takes the differences between sub-regions into account, but also effectively utilizes the constraint relationship between sub-regions and the global region to preserve the overall structure of the target. During tracking, the motion model of sub-regions and the global target region are basically consistent, and the sub-region patches are zero-padded to the same size as the global target region, which can effectively avoid noise aliasing during correlation operation, thereby improved the robustness of the discriminative correlation filter. Noise aliasing is an error that occurs when signal reconstruction, that is to say, information from high frequency is disguised as low frequency content. The advantage of the proposed DCFB tracking method is that the effective appearance of the remaining visible sub-region patches can still provide reliable cues for tracking when the target is partially occluded, since we can formulate multiple correlation filters corresponding to different sub-region patches simultaneously. Extensive experiments on the OTB-2015 [31] and VOT-2015 [32] datasets evidence the effectiveness of the proposed framework compared with several state-of-the-art trackers.

The contributions of this work are as follows. First, we formulate multiple discriminative correlation filters corresponding to different sub-region and global region patches simultaneously to combine and optimize the final tracking result. Second, we ensure that sub-region patches are zero-padded to the same size as the global target region to avoid noise aliasing during correlation operation, thereby improved the robustness of the discriminative correlation filter. Third, our proposed model not only exploits the constraint relationship between sub-regions and the global target region to learn multiple discriminative correlation filters jointly, but also preserves the overall structure of the target. Fourth, we validate our tracker by demonstrating that it performs favorably against state-of-the-art trackers, using the OTB-2015 [31] and VOT-2015 [32] as two benchmark datasets.

The remainder of the paper is arranged as follows. In Section 2, we review work related to ours. In Section 3, we describe the proposed tracking algorithm in detail. The experimental evaluations and analysis are reported in Section 4. Finally, we summarize this paper and point out the research direction of future work in Section 5.

## 2. Related Works

As a result of the annual visual object tracking (VOT) challenge, many excellent visual tracking algorithms have emerged one after another. To review these algorithms, readers can refer to References [32–35] for more details. In this section, we mainly review the literature related to our work, including correlation filter-based and part-based correlation filter tracking algorithms.

### 2.1. Correlation Filter Visual Tracking

The potential of correlation filters for visual tracking has attracted widespread attention, mainly because the correlation operation reduces the overhead time through fast Fourier transformation (FFT) in the frequency domain. Bolme et al. [10] the first used correlation filter to build a tracking framework by learning a minimum output sum of squared error (MOSSE) for appearance model. Its speed is several hundred frames per second, meeting the requirements of real-time tracking. The correlation filter of the circulant structure with kernel (CSK) [14] uses the kernel trick to learn the appearance model and further improve tracking performance. The KCF tracker [15] is an upgraded version of CSK. It uses the histogram of oriented gradients (HOG) feature instead of the original grayscale feature to represent the target, and shows an amazing speed on the OTB2013 dataset [36], but cannot achieve online scale estimation. The discriminative scale space tracking (DSST) [11] method consists of a translation correlation filter and a scale correlation filter to achieve target localization and target scale detection, respectively. The scale adaptive with multiple features (SAMF) [12] tracker solves online scale detection using KCF as a baseline. The fast DSST (fDSST) method [37] is an accelerated version of DSST. In addition to increasing speed, it is more accurate in scale estimation and tracking is more robust.

Recently, the strategy of reducing the boundary effects [25,26,38] has been integrated into the correlation filter model, which has greatly improved the quality of the tracking model. In Reference [21,27–29], the authors used depth features instead of hand-crafted features for visual tracking, which further enhances the robustness and accuracy of the tracker. The continuous convolution operators (C-COT) tracking algorithm [21] presents the best performance in VOT2016 [34], but it is a very complex model that cannot achieve real-time tracking. The efficient convolution operators (ECO) tracker [29] solved the speed problem of C-COT by optimizing model size, sample set size, and update strategy. In this work, we construct a robust discriminative correlation filter bank tracking framework to solve the tracker drift caused by occlusion scenarios, which is different from the existing correlation filter trackers.

### 2.2. Part-Based Correlation Filter Tracking Algorithm

The part-based strategy using a correlation filter effectively solves the occlusion problem for visual tracking owing to the fact that visible parts are still used when occlusion occurs. Liu et al. [39]

proposed using a discriminative part selection strategy to filtrate the most discriminative information parts from several candidate parts, and each part corresponds to a correlation output. Subsequently, all of the correlation outputs are combined to estimate the position of the target. In Reference [17], the authors proposed a reliable patch tracking algorithm to achieve target tracking by using reliable patches that can be effectively tracked throughout the tracking video. These reliable patches are calculated and selected by the trackable confidence function, and the trackable confidence and motion information are incorporated into the particle filter framework in order to estimate the position and scale of the target. In Reference [40], the authors proposed coupling interactions between local and global correlation filters for handling partial occlusion during tracking. First, using the local parts to estimate the initial position of the target based on the deformable model. Once a part is occluded or the appearance changes severely, the reliable parts provides new information to estimate a coarse prediction; then, the coarse result defines the search neighborhood, the final position of the target is estimated in conjunction with global filter; finally, the new prediction is provided to the part filters as the new reference position that is used in the deformable model to again estimate the position of the next frame target. Liu et al. [30] proposed to use the spatial constraints among local parts to preserve the structure of the target for the motion model that not only allows most parts to have similar motion, but also tolerate outlier parts of different motion directions. During tracking, the state of the part is predicted based on the maximum correlation response value of each part, and the location of the target is ultimately estimated by weighting average the state of all parts. Fan et al. [41] introduced a local-global correlation filter (LGCF) tracking method to solve the occlusion issue, which not only takes into account the constrained relationship of the local parts and global target, but also integrates the temporal consistencies of the local parts and global target to mitigate model drift, and then uses an occlusion detection model to exclude the occluded part to accurately estimate the location of the target. Wang et al. [42] developed a novel structured correlation filter model based on coupled interactions between a static model and a dynamic model to handle partial occlusion in tracking. The static model uses the star graph to model the spatial structure among parts to capture the spatial information of the parts and achieve the initial prediction of the target. The dynamic model uses this coarse initial prediction as a reference to estimate the final state of the target through Bayesian inference, and then the new target location is provided to the static model in order to update. However, the target response adaptive change tracking algorithm [24] exhibits superior performance when dealing with occlusion problems in scenarios, which utilizes the idea that the target response changes with frame changes. In Reference [23], the authors first considered the quality problem of the training samples in a joint optimization framework. The joint optimization function—consisting of the appearance model and the training sample weights—is used to purify the training sample set, thereby improving the tracking accuracy to counter occlusion challenges in a scene.

More relevant to our work is [41]. However, our method is different from [41] reflecting the following three aspects. First, we do not use the circulant structure of the training sample to learn correlation filters, empirical experiments show that these cyclic shift patches are only approximations of the actual samples, and are thus unreliable in the actual tracking occlusion scene. Second, in our method, the sub-region patches used for training the correlation filter are zero-padded to the same size as the global target region to avoid noise aliasing during the correlation operation. Third, we formulate multiple discriminative correlation filters instead of kernelized correlation filters corresponding to different sub-region and global region patches simultaneously to combine and optimize the final tracking output.

### 3. Our Tracking Framework

In this section, we describe the proposed tracking framework in detail. Starting with discussing the employed baseline tracker, we then introduce the proposed tracking framework. Finally, the proposed tracking algorithm is presented.

### 3.1. Baseline Approach

We adopted the DSST algorithm [11] as our baseline tracker. The DSST algorithm is a discriminative correlation filter tracker, which learns an appearance model on a single sample  $f$ , centered around the target with a  $d$ -dimension feature vector for calculation during tracking.  $f^l$  is used to represent the  $d$ -dimension feature vector of sample  $f$ , and  $l \in \{1, 2, \dots, d\}$ . The correlation filter  $h$ , consisting of one filter  $h^l$  per feature dimension, is optimized by minimizing the following objective function:

$$\varepsilon = \left\| \sum_{l=1}^d h^l * f^l - g \right\|^2 + \lambda \sum_{l=1}^d \|h^l\|^2, \tag{1}$$

where  $g$  is the Gaussian function label,  $f$  is the training example,  $\lambda$  is the regularization term coefficient, and  $*$  denotes circular correlation. The closed form expression of Equation (1) is as follows:

$$H^l = \frac{\overline{G}F^l}{\sum_{k=1}^d \overline{F^k}F^k + \lambda}, l = 1, 2, \dots, d, \tag{2}$$

where  $F, H$  and  $G$  denote the Fourier transforms of  $f, h$  and  $g$ , respectively.  $\overline{H}$  is the complex conjugate of  $H$ .

The updated plan is as follows:

$$\begin{aligned} A_t^l &= (1 - \eta)A_{t-1}^l + \eta\overline{G}_tF_t^l \\ B_t &= (1 - \eta)B_{t-1} + \eta \sum_{k=1}^d \overline{F_t^k}F_t^k, \end{aligned} \tag{3}$$

where  $\eta$  is a learning rate parameter.  $A_t^l$  and  $B_t$  are the numerator and denominator of the filter  $H_t^l$ .

We then estimated the new location of the target according to the maximum correlation score  $y_t$  on the candidate patch  $z_t$  in a new frame  $t$ . The maximum correlation score  $y_t$  is computed as:

$$y_t = \mathcal{F}^{-1} \left( \frac{\sum_{l=1}^d \overline{A_{t-1}^l}Z_t^l}{B_{t-1} + \lambda} \right). \tag{4}$$

Readers may refer to Reference [11] for more details.

### 3.2. The Proposed Tracking Model

Based on the baseline tracker's objective function Equation (1), we obtained the objective function of the sub-region and global model; see Equations (5) and (6), respectively.

$$\arg \min_{\{h_s\}_{s=1}^N} \sum_{s=1}^N \left( \left\| \sum_{l=1}^d h_s^l * f_s^l - g_s \right\|^2 + \lambda \sum_{l=1}^d \|h_s^l\|^2 \right) \tag{5}$$

$$\arg \min_{h_g} \left( \left\| \sum_{l=1}^d h_g^l * f_g^l - g_g \right\|^2 + \lambda \sum_{l=1}^d \|h_g^l\|^2 \right). \tag{6}$$

Here,  $N$  indicates that the target is divided into  $N$  sub-regions. Each sub-region is zero-padded to the same size as the global image and corresponds to a discriminative correlation filter.

In tracking, the sub-regions and global model of the target are difficult to keep consistent because of the target self-deformation and the interference of the occlusion scenarios. In order to preserve the overall structure of the target among the sub-regions and global region to mitigate the drift risk and



tolerate the outliers of the sub-regions model, the constraint between the sub-regions and the global model should be added and sparse. The constraint model [41] is represented by Equation (7):

$$h_s = h_g + v_s, \tag{7}$$

where  $h_s$  and  $h_g$  represent motion match models of the sub-region and global region, respectively, and  $v_s$  denotes the constraint between  $h_s$  and  $h_g$ .

In object tracking sequence, the target and background between consecutive frames are basically similar, so the target matching model  $h^{t-1}$  is consistent with  $h^t$ . This phenomenon is called temporal consistency [41]. Its mathematical model is shown as follows:

$$\begin{aligned} \arg \min_{h_s} \sum_{s=1}^N \|h_s^{t-1} - h_s^t\|^2 \\ \arg \min_{h_g} \|h_g^{t-1} - h_g^t\|^2. \end{aligned} \tag{8}$$

By combining the above points to construct our tracking model, it can effectively learn the correlation filter models of the sub-regions and global region through the following optimization:

$$\begin{aligned} \arg \min_{\{h_s\}_{s=1}^N, h_g^t} \left\{ \sum_{s=1}^N (\|h_s^t * f_s^t - g_s^t\|^2 + \lambda \|h_s^t\|^2) + (\|h_g^t * f_g^t - g_g^t\|^2 + \lambda \|h_g^t\|^2) + \gamma \sum_{s=1}^N \|v_s^t\|_1 + \frac{\zeta}{2} \|h_g^t - h_g^{t-1}\|^2 + \frac{\beta}{2} \sum_{s=1}^N \|h_s^t - h_s^{t-1}\|^2 \right\} \\ \text{s.t. } h_s^t = h_g^t + v_s^t, \end{aligned} \tag{9}$$

where  $\zeta$  and  $\beta$  denote trade-off coefficients,  $\gamma$  is the regularization term coefficient. The trade-off coefficients are used to control the strength of the regularization term and prevent it from becoming larger during the optimization process. In fact, the motion models between consecutive frames are basically similar, so the regularization terms formed by the differences of their motion models can't be too strong. Otherwise, if the target is occluded during the tracking process, it will lead to tracking failure or tracker drift, that is to say, the trade-off coefficients act as a role in guaranteeing the similarity of the motion models between consecutive frames.

### 3.3. Optimization Tracking Model

The optimization of Equation (9) is solved by constructing a Lagrangian function, which is an objective function formed by the augmented Lagrange multipliers being incorporated into the constraint condition. Then, the alternating direction method of multipliers (ADMM) [43] is used to implement an iterative update through a series of simple closed form operations. For details of the designed Lagrangian function, see Equation (10).

$$\begin{aligned} L(h_g^t, \{h_s^t, v_s^t, \varepsilon_s^t, \tau_s^t\}_{s=1}^N) = \sum_{s=1}^N (\|h_s^t * f_s^t - g_s^t\|^2 + \lambda \|h_s^t\|^2) + (\|h_g^t * f_g^t - g_g^t\|^2 + \lambda \|h_g^t\|^2) \\ + \gamma \sum_{s=1}^N \|v_s^t\|_1 + \frac{\zeta}{2} \|h_g^t - h_g^{t-1}\|^2 + \frac{\beta}{2} \sum_{s=1}^N \|h_s^t - h_s^{t-1}\|^2 \\ + \sum_{s=1}^N \left\{ (\varepsilon_s^t)^T (h_s^t - h_g^t - v_s^t) + \frac{\tau_s^t}{2} \|h_s^t - h_g^t - v_s^t\|^2 \right\}. \end{aligned} \tag{10}$$

Here,  $\varepsilon_s^t$  and  $\tau_s^t$  are the Lagrange multiplier and penalty parameter, respectively. However, the new objective function becomes Equation (11).

$$\arg \min_{h_g^t, \{h_s^t, v_s^t, \varepsilon_s^t, \tau_s^t\}_{s=1}^N} L(h_g^t, \{h_s^t, v_s^t, \varepsilon_s^t, \tau_s^t\}_{s=1}^N). \tag{11}$$

Next, each parameter is iteratively updated using the ADMM by minimizing Equation (11). When one of the parameters is updated, the other parameters remain fixed. The procedure for updating each parameter variable is as follows.

**Update  $h_s^t$ :** The  $h_s^t$  is updated by solving Equation (12) with the closed form solution, while the other parameters are fixed.

$$h_s^t = \arg \min_{h_s^t} \left\{ \left( \|h_s^t * f_s^t - g_s^t\|^2 + \lambda \|h_s^t\|^2 \right) + \frac{\zeta}{2} \|h_s^t - h_s^{t-1}\|^2 + \sum_{s=1}^N \left\{ -(\epsilon_s^t)^T h_s^t + \frac{\tau_s^t}{2} \|h_s^t - h_g^t - v_s^t\|^2 \right\} \right\}, \quad (12)$$

its closed solution gives Equation (13).

$$h_s^t = \mathcal{F}^{-1} \left( \frac{F_s^t \overline{G}_g^t + \frac{\zeta}{2} H_g^{t-1} + \sum_{s=1}^N \left\{ (\epsilon_s^t)^T + \frac{\tau_s^t}{2} (H_s^t - v_s^t) \right\}}{F_s^t \overline{F}_g^t + \left( \lambda + \frac{\zeta}{2} + \sum_{s=1}^N \frac{\tau_s^t}{2} \right) I} \right), \quad (13)$$

where  $F, H, G$  and  $v$  denote the discrete Fourier transforms (DFTs) of  $f, h, g$  and  $v$ , respectively.  $I$  is the identity matrix. The bar  $\overline{G}_g^t$  denotes a complex conjugation.

**Update  $h_s^t$ :** The  $s^{th}$  independent sub-problem  $h_s^t$  is updated by solving Equation (14) with the closed form solution, while the other parameters are fixed.

$$h_s^t = \arg \min_{h_s^t} \left\{ \|h_s^t * f_s^t - g_s^t\|^2 + \lambda \|h_s^t\|^2 + \frac{\beta}{2} \|h_s^t - h_s^{t-1}\|^2 + (\epsilon_s^t)^T h_s^t + \frac{\tau_s^t}{2} \|h_s^t - h_g^t - v_s^t\|^2 \right\}, \quad (14)$$

its closed solution gives Equation (15).

$$h_s^t = \mathcal{F}^{-1} \left( \frac{\overline{G}_s^t F_s^t + \frac{\beta}{2} H_s^{t-1} + \frac{\tau_s^t}{2} (H_g^t + v_s^t) - (\epsilon_s^t)^T}{\overline{F}_s^t F_s^t + \left( \lambda + \frac{\beta}{2} + \frac{\tau_s^t}{2} \right) I} \right). \quad (15)$$

**Update  $v_s^t$ :** The  $s^{th}$  independent sub-problem  $v_s^t$  is updated by solving Equation (16) with the closed form solution, while the other parameters are fixed.

$$v_s^t = \arg \min_{v_s^t} \left\{ \gamma \|v_s^t\|_1 - (\epsilon_s^t)^T v_s^t + \frac{\tau_s^t}{2} \|h_s^t - h_g^t - v_s^t\|^2 \right\}. \quad (16)$$

The solution of Equation (16) can be converted to the solution of Equation (17) according to Reference [43], and its closed solution gives Equation (18).

$$v_s^t = \arg \min_{v_s^t} \left\{ \frac{\gamma}{\tau_s^t} \|v_s^t\|_1 + \frac{1}{2} \left\| v_s^t - \left( h_s^t + \frac{\epsilon_s^t}{\tau_s^t} - h_g^t \right) \right\|^2 \right\}, \quad (17)$$

$$v_s^t = S_{\frac{\gamma}{\tau_s^t}} \left( h_s^t + \frac{\epsilon_s^t}{\tau_s^t} - h_g^t \right). \quad (18)$$

Here,

$$S_\theta(x_i) = \text{sign}(x_i) \max(0, |x_i| - \theta) \quad (19)$$

represents the soft threshold function of the vector  $x$ .

**Update  $\epsilon_s^t$  and  $\tau_s^t$ :** The Lagrange multiplier  $\epsilon_s^t$  and penalty parameter  $\tau_s^t$  are updated as in Equation (20).

$$\epsilon_s^t = \epsilon_s^t + \tau_s^t (h_s^t - h_g^t - v_s^t), \quad \tau_s^t = \varphi \tau_s^t \quad (20)$$

The solution of the objective function Equation (11) obtained through ADMM optimization is shown in Algorithm 1.

---

**Algorithm 1:** ADMM optimization for Equation (11).

---

**Input:**  $g_g^t, g_s^t, \lambda, \gamma, \zeta, \beta, \{h_s^t, h_s^{t-1}, \varepsilon_s^t, \tau_s^t\}_s^N, h_g^t, h_g^{t-1}$   
**Output:** Correlation filters  $h_g^t, \{h_s^t\}_{s=1}^N$

```

1 while not converged do
2   Update  $h_g^t$  according to Equation (13)
3   for  $s = 1$  to  $N$  do
4     Update  $h_s^t$  according to Equation (15)
5     Update  $v_s^t$  according to Equation (18)
6     Update  $\varepsilon_s^t$  and  $\tau_s^t$  according to Equation (20)
7   end
8 end

```

---

### 3.4. Tracking

#### 3.4.1. Position Estimation

In a new frame  $t$ , a sample patch  $z_t$  is extracted from the region centered around the previous frame target position. The HOG feature vector is then used to represent the sample patch  $z_t$ . Using the obtained sub-region and global region correlation filters, we can obtain the correlation responses of the sub-regions and global region in the frequency domain.

The correlation response  $y_g^t$  of global region is computed by:

$$y_g^t = \mathcal{F}^{-1} \left( \overline{H_g^{t-1}} \odot Z_g^t \right), \tag{21}$$

the correlation response  $y_s^t$  of the  $s^{th}$  sub-region is computed by:

$$y_s^t = \mathcal{F}^{-1} \left( \overline{H_s^{t-1}} \odot Z_s^t \right), \tag{22}$$

where the operator  $\odot$  is the Hadamard product, while  $H_g^{t-1}$  and  $H_s^{t-1}$  are the updated correlation filters of the global region and sub-regions in the previous frame, respectively.

The maximum correlation response value corresponds to the coordinate that indicates the location of the target, that is to say, the position  $p_g$  of the global region target is obtained by finding the maximum correlation response  $y_g^t$ , and the position  $p_s$  of the  $s^{th}$  sub-region target is obtained by finding the maximum correlation response  $y_s^t$ . The final target position  $P$  estimation depends on the global region target position  $p_g$  and the sub-region target position  $p_s$ , as follows:

$$P = \omega_g p_g + \sum_{s=1}^N \omega_s (p_s + \Delta_s), \tag{23}$$

where  $\omega_g$  and  $\omega_s$  denote the weights of the global region target position and the sub-regions target position, respectively.  $\Delta_s$  is the deformation vector [44] between the  $s^{th}$  sub-region and the object center. These weights are calculated based on their corresponding correlation response maximum values, as in Reference [45].

$$\omega_g = \frac{f(\max(y_g))}{f(\max(y_g)) + \sum_{s=1}^N f(\max(y_s))} \tag{24}$$

$$\omega_s = \frac{f(\max(y_s))}{f(\max(y_g)) + \sum_{s=1}^N f(\max(y_s))}, \tag{25}$$

where  $f(x) = \frac{1}{1+\exp(-x)}$ .

### 3.4.2. Scale Estimation

Resolving the scale change of the target is an important issue for visual tracking, and can make the tracking process more accurate. Existing correlation filter trackers [11,12,19,37] exhibit superior performance in estimating target scale change. These algorithms estimate the target’s scale by constructing a target pyramid, which is a different scale pool sampled around the estimated current target position and then correlated with the updated discriminative correlation filter. The maximum correlation response value corresponding to the scale level is the current target size. However, the scale of these filters does not change adaptively as the target scale changes, which leads to the inaccurate estimation of the target scale. Using the idea that the relative distance among the sub-regions and the target scale change is proportional, the filter scale can be adaptively changed to accurately estimate the target’s scale. This approach is described in References [40,41,45]. In this work, we use existing heuristics to estimate the target’s scale according to the method presented in Reference [45]. Specifically, we calculate the target’s scale in the  $t$ -frame as follows:

$$(w_t, h_t) = (w_{t-1}, h_{t-1}) \times \frac{1}{N(N-1)} \sum_{i=1, j=1}^N \frac{\|p_i^t - p_j^t\|}{\|p_i^{t-1} - p_j^{t-1}\|} \quad s.t. \quad i \neq j, \tag{26}$$

where  $w_t$  and  $h_t$  denote the width and height of the target in the  $t$ -frame, respectively.  $\|\cdot\|$  stands for the Euclidean metric.  $p_i^t$  indicates the position of the  $i$ -th sub-region in the  $t$ -th frame.

### 3.4.3. Model Update

During online tracking, the appearance model of the target may undergo severe changes. In order to solve these situations, after predicting a new target position in each frame, we have to update the sub-regions and global region correlation filters. To obtain a relatively good approximation, we used dynamic averaging to update the sub-regions and global region correlation filters as follows:

$$H_g^t = \eta H_g^t + (1 - \eta) H_g^{t-1} \tag{27}$$

$$H_s^t = \eta H_s^t + (1 - \eta) H_s^{t-1}, \tag{28}$$

where  $t$  and  $\eta$  denote the frame index and learning rate, respectively. The global region correlation filter  $h_g^t = \mathcal{F}^{-1}(H_g^t)$ , and the sub-regions correlation filter  $h_s^t = \mathcal{F}^{-1}(H_s^t)$ .

### 3.5. Proposed Tracking Algorithm

An overview of the proposed tracking algorithm is listed in Algorithm 2.

**Algorithm 2:** The proposed tracking algorithm.

---

```

Input: Image sequences  $\{f_i\}_1^t$ 
Output: Tracking results  $\{y_i\}_1^t$ 
1 for  $i = 1$  to end of sequence do
2   if  $i > 1$  then
3     Crop out the global region and sub-region at  $y_{i-1}$  from  $f_i$ 
4     Calculate global region position using Equation (21)
5     Calculate sub-region positions using Equation (22)
6     Calculate target position  $P_i$  using Equation (23)
7     Calculate target scale  $(w_i, h_i)$  using Equation (26)
8     Collect tracking result  $y_i = (P_i, w_i, h_i)$ 
9   end
10  Crop out the global region and sub-region at  $y_i$  from  $f_i$ 
11  Calculate correlation filters  $h_g^t, \{h_s^t\}_{s=1}^N$  using Algorithm 1
12  Update global region correlation filter using Equation (27)
13  Update sub-region correlation filters using Equation (28)
14  for  $s = 1$  to  $N$  do
15    if  $i > 1$  then
16      Update target template set for sub-region  $s$ 
17    else
18      Initialize target template set for sub-region  $s$ 
19    end
20  end
21 end

```

---

## 4. Experiment

In this section, the effectiveness of the proposed tracking algorithm is confirmed by comparing it with state-of-the-art trackers on two popular datasets: the OTB-2015 [31] and VOT-2015 [32] visual tracking benchmark datasets. In addition, we present the details of implementation and the ablation analysis in Sections 4.1 and 4.2, respectively. The experimental results are shown in Section 4.3, and the experimental analysis is reported in Section 4.4.

### 4.1. Implementation Details

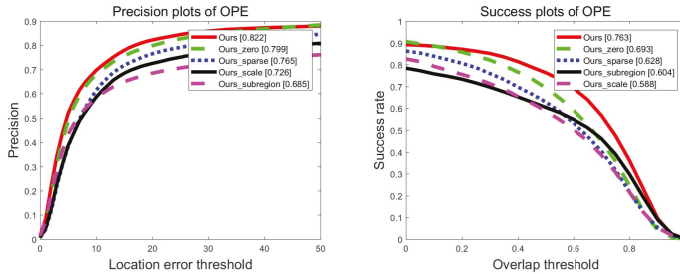
Our tracker was implemented using the MATLAB R2017a software platform. We set the same parameters during tracking, and ran at around 1.5 fps. The regularization term coefficient  $\lambda$  and the learning rate  $\eta$  were set to 0.01 and 0.025, respectively. The parameters  $\gamma, \zeta$  and  $\beta$  were all set to 0.01. We found that setting the number of sub-regions  $N$  to 4 was more suitable for the experiment. This is because too many sub-regions cause a low target resolution, resulting in less feature information for identifying the target, while too few sub-regions will reduce the feature information of the visual parts due to occlusion. We used the HOG feature for target representation.

### 4.2. Ablation Analysis

Our algorithm consists of four important components including zero-padding, scale estimation, sub-regions, and sparse constraint. In order to evaluate the effectiveness of each component in our tracking framework, we conduct ablation study on the OTB-2015 dataset by disabling each component one by one. The comparison results of the distance and overlap precision are shown in Figure 2.

As shown in Figure 2, without the zero-padding component, the tracking results are relatively good due to the accurate scale estimation, the coupling and constraints between the sub-regions and

the global region, all of which are attributed to our coupled-region tracking formulation. Without the sparse constraint component, in complex scenarios, due to the inability to tolerate the outliers of the subregion, our tracking model is difficult to completely preserve the internal structure of the target, which may cause the risk of tracker drift. Therefore, the scores of the distance and overlap precision are not the result of the promising. The scale estimation is an important role in our tracking framework. To evaluate the performance of our tracker, we disable the scale estimation component for tracking. Figure 2 shows that the value of the overlap precision is low. The main reason is that the tracker cannot adaptively change the scale through scale variation sequence.



**Figure 2.** Precision and success plots of disabling component tracker on the OTB-2015 dataset for the ablation analysis. In this plot, Ours\_subregion denotes Ours without using the subregion, and likewise Ours\_scale, Ours\_zero, and Ours\_sparse denotes Ours without using scale estimation, zero-padding, and sparse constraint, respectively.

Extensive evaluations demonstrate that coupling subregion tracking formulation is an effective strategy to solve occlusion problem. However, we disable the sub-region component for tracking, which is similar to the baseline tracker, while the baseline tracker does not solve the occlusion problem, so it is lower than the proposed tracker in terms of the value of the distance and overlap precision. In general, each component plays an important role in our tracking framework, and by jointly optimizing them, we receive the promised tracking performance.

#### 4.3. Experimental Results

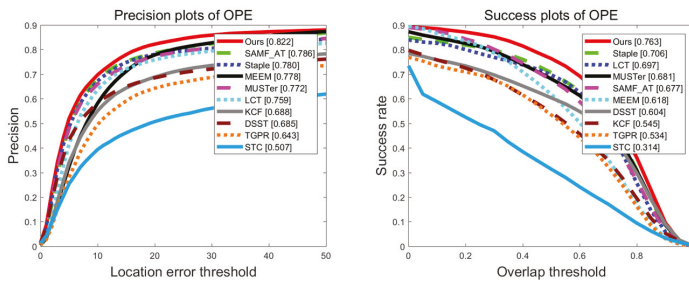
We performed comprehensive experiments on the OTB-2015 [31] and VOT-2015 [32] benchmark datasets to evaluate the performance of our tracker.

##### 4.3.1. Experiment on the OTB-2015 Dataset

The OTB-2015 benchmark dataset contains 100 fully annotated video sequences, which are divided into 11 different attributes such as: Illumination Variation (IV), Scale Variation (SV), Occlusion (OCC), Deformation (DEF), Motion Blur (MB), Fast Motion (FM), In-Plane Rotation (IPR), Out-of-Plane Rotation (OPR), Out-of-View (OV), Background Clutters (BC), and Low Resolution (LR). These attributes represent different challenging scenarios for visual tracking. Using this dataset to test the performance of our algorithm by comparing it with the other nine excellent trackers: TGPR [46], SAMF\_AT [24], STC [20], MUSTer [16], Staple [47], LCT [19], KCF [15], MEEM [7], DSST [11], and BACF [38]. We reported the comparison results through the one-pass evaluation (OPE) with precision and success plots. The precision plot shows the percentage of frames in which the center position error is smaller than a certain threshold; we used a threshold of 20 pixels for all comparison trackers. The success plot presents the percentage of successful frames where the overlap score between the tracking bounding box and the ground-truth bounding box was more than one threshold. The overlap score is defined as  $\frac{area(B_T \cap B_G)}{area(B_T \cup B_G)}$ , where  $B_T$  and  $B_G$  are the tracking bounding box and the ground-truth bounding box, respectively. We used a threshold of 0.5 to rank all comparison

trackers in the success plots. The precision and success plots demonstrate the mean results over the OTB-2015 dataset.

As shown in Figure 3, the comparison results of the precision and success plots show that our tracking algorithm outperforms other state-of-the-art trackers in terms of distance precision and overlap precision. We can see that our tracker achieved the ranking scores of 0.822 and 0.763 in distance precision and overlap precision, respectively. However, the distance precision and overlap precision ranking scores of the baseline tracker DSST [11] were 0.693 and 0.535, respectively. Obviously, our tracking algorithm was greatly improved in terms of distance and overlap precision. There are two main reasons. First, the motion model of the proposed method is completely different from the DSST. We use the idea of optimization and constraint to retain the internal structure of the target, while DSST has no optimization model. Second, the scale estimation method is different. We use the strategy of proportional to relative distance among sub-regions, whereas DSST uses the strategy of constructing target scale pyramid to estimate scale.



**Figure 3.** Precision and success plots of different trackers on the OTB-2015 dataset. Our tracker is better than other trackers.

To demonstrate the robustness of our tracker when faced with different challenging attributes, we present the comparison results of eight attributes (IV, SV, OCC, DEF, MB, FM, OPR and BC) in terms of distance and overlap precision. See Figures 4 and 5 for details. The results show that our tracker ranks second and first in the precision and success plot for sequences with deformation, respectively, while it ranks first in the precision and success plots for the other seven scenarios with challenging attributes. These results confirm that our approach has a very promising performance in dealing with such challenges, especially in scenarios with occlusion.

Both our algorithm and the BACF [38] tracker use the idea of zero-padding and ADMM iterative optimization, and both use the dynamic average strategy formulation for model update. Figure 6 shows the performance comparison of our method and BACF on the OTB-2015 dataset in terms of background clutters, occlusion and all sequences challenging attributes.

In the sequence of background clutters attribute, the results in Figure 6 show that our method and BACF are 0.85 and 0.83 in terms of distance precision, respectively, and the overlap precision is 0.786 and 0.796 respectively. In the sequence of occlusion, our approach outperforms BACF in terms of distance and overlap precision, this is the advantage of our coupled-region visual tracking formulation in solving occlusion problems. In all sequences, our approach outperforms BACF in terms of distance precision, whereas our method is slightly behind BACF in terms of overlap precision. In general, our approach and BACF have their own merits in performing tracking.

In order to more intuitively demonstrate the superior performance of our algorithm, we plotted the experimental results of 11 different challenge attribute sequences in OTB-2015 into Tables 1 and 2. By comparing distance and overlap precision with other state-of-the-art trackers, it can be seen at a glance that our tracker is superior to the other trackers apart from BACF in terms of overlap precision. However, in terms of distance precision, our tracker achieved the best results in six of the 11 attributes.

In the remaining attribute sequences (DEF, IPR, OV, FM, and LR), the BACF [38] and MEEM [7] performs better. Based on the results of Tables 1 and 2, we analyze the reasons for the advantages of our method in terms of partial challenging attributes.

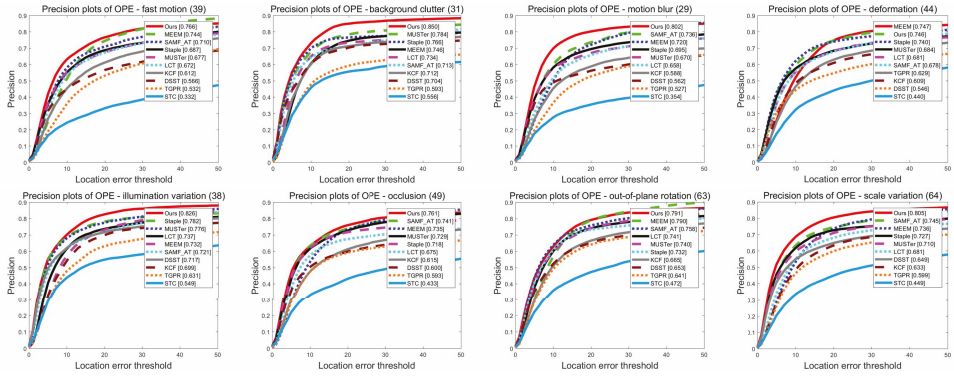


Figure 4. Performance evaluation of distance precision on eight challenging attributes (FM, BC, MB, DEF, IV, OCC, OPR and SV) of the OTB-2015 dataset.

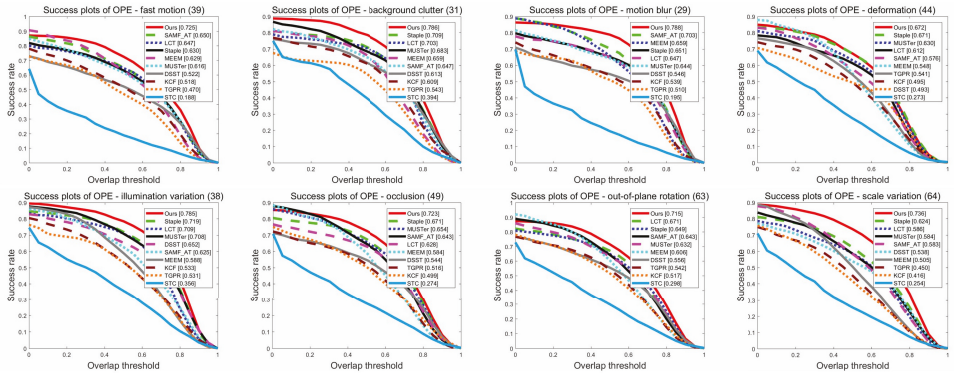


Figure 5. Performance evaluation of overlap precision on eight challenging attributes (FM, BC, MB, DEF, IV, OCC, OPR and SV) of the OTB-2015 dataset.

In the scene of the illumination variation(IV) attribute, the target appearance model will be seriously affected, often causing the tracker to drift. In our tracking framework, the HOG feature was used to represent the target and to some extent suppress the illumination variation. Together with our accurate scale estimation scheme, the tracker is more robust. In the actual tracking scene, the occlusion is usually accompanied by the occurrence of background clutters (BC). Our method solves the occlusion problem, which is naturally equivalent to solving the background clutters problem, which is attributed to the coupling formulation between the sub-regions and the global region. The internal structure of the target is preserved, and the outliers of the sub-region can be tolerated. In the out-of-plane rotation (OPR)scenarios, we use the idea that the relative distance among the sub-regions and the target scale change is proportional and the filter scale can be adaptively changed to accurately estimate the target’s scale, thereby lowering the risk of drift and tracking failure. In the low resolution (LR) sequences, our tracker does not perform as well as MEEM in terms of distance precision, because MEEM tracks the target with multiple appearance models. While in our tracking framework, the parts are small in size and low in resolution, and cannot contain enough target feature information, so that our algorithm



does not perform well in low-resolution scenes. However, in the range of successful frames tracked, since the scale of our correlation filter can be adaptively changed to accurately estimate the target's scale, our tracker performs well in terms of overlap precision.

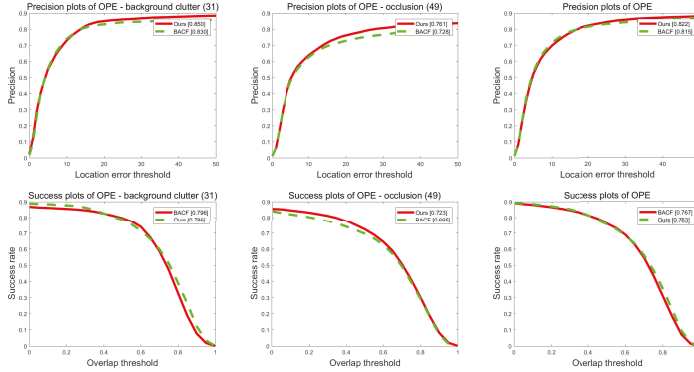


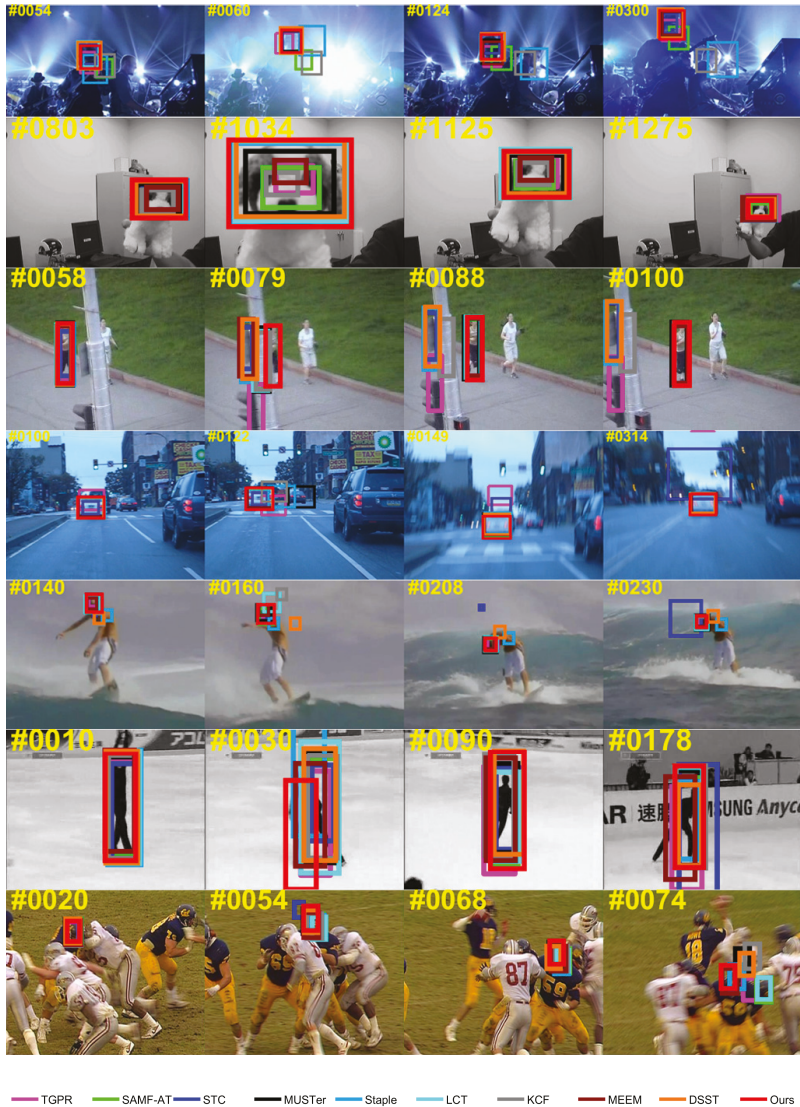
Figure 6. Performance evaluation of our method and BACF on the OTB-2015 dataset.

Table 1. Comparison of our tracker with other state-of-the-art trackers on 11 different attributes of the OTB-2015 dataset. Average precision scores (%) at a threshold of 20 pixels are presented. The optimal results are highlighted in bold.

Attribute	Ours	BACF	TGPR	SAMF_AT	STC	MUSTer	Staple	LCT	KCF	MEEM	DSST
IV	<b>82.6</b>	80.8	63.1	72.1	54.9	77.6	78.2	73.7	69.9	73.2	71.7
SV	<b>80.5</b>	77.4	59.9	74.5	44.9	71	72.7	68.1	63.3	73.6	64.9
OCC	<b>76.1</b>	72.8	59.3	74.1	43.3	72.9	71.8	67.5	61.5	73.5	60
DEF	74.6	<b>75.8</b>	62.9	67.8	44	68.4	74	68.1	60.9	74.7	54.6
MB	<b>80.2</b>	73.6	52.7	73.6	35.4	67	69.5	65.8	58.8	72	56.2
FM	76.6	<b>78.6</b>	53.2	71	33.2	67.7	68.7	67.2	61.2	74.4	56.6
IPR	77	77.8	65.8	77.5	47.6	76.8	76.3	77.5	68.6	<b>78.8</b>	69.4
OPR	<b>79.1</b>	77.3	64.1	75.8	47.2	74	73.2	74.1	66.5	79	65.3
OV	64.1	<b>76.5</b>	49.3	65	35	59.1	66.1	59.2	49.8	68.5	47.8
BC	<b>85</b>	83	59.3	71.3	55.6	78.4	76.6	73.4	71.2	74.6	70.4
LR	74.7	79.5	62.6	78.8	48.9	74.7	69.5	69.9	67.1	<b>80.8</b>	68.4
Overall	<b>82.2</b>	81.5	64.3	78.6	50.7	77.2	78	75.9	68.8	77.8	68.5

Next, the qualitative evaluation and analysis were carried out to further demonstrate that the performance of the proposed tracker is superior to other state-of-the-art trackers on the OTB-2015 [31] image sequence. Figure 7 shows the qualitative comparison results of our algorithm with nine state-of-the-art trackers (TGPR [46], SAMF\_AT [24], STC [20], MUSTer [16], Staple [47], LCT [19], KCF [15], MEEM [7], and DSST [11]) on seven sequences (*Shaking*, *Dog1*, *Jogging1*, *BlurCar3*, *Surfer*, *Skater2* and *Football1*). In *Shaking*, illumination variation is the most representative challenging attribute. The SAMF\_AT, Staple, and KCF trackers performed poorly due to noise image gradient effects. In our tracking framework, the HOG feature was used to represent the target and to some extent suppress the illumination variation. Compared to other trackers, our tracker showed better tracking results. In *Dog1*, scale variation is the most representative challenging attribute. Although there are significant scale variations between different frames, our tracking algorithm could accurately estimate the scale and position of the target. However, the MEEM, KCF, and TGPR trackers failed to address the challenges of scale variations. In the *Jogging1* sequence, occlusion is the most representative challenging attribute. When the target experienced partial and full occlusion, the proposed algorithm performed more robustly during the tracking process. This is because the remaining visible sub-region patches can still provide reliable cues for tracking. However, the tracking bounding box of these trackers (DSST, STC,

Staple, TGPR, and KCF) lost the target when the occlusion occurred, eventually resulting in tracking failure. In other sequences (*BlurCar3*, *Surfer*, *Skater2* and *Football1*), our tracker performed well in terms of scale and position estimation. However, the STC tracker did not perform well during the tracking process. The main reason for this was attributed to two aspects: first, the STC tracker uses image intensity as features to represent the appearance model of the target context. Second, the estimated scale depends on the response map of a single filter.



**Figure 7.** Qualitative evaluation of our algorithm and nine other state-of-the-art trackers on seven sequences (from top to bottom: *Shaking*, *Dog1*, *Jogging1*, *BlurCar3*, *Surfer*, *Skater2* and *Football1*). These sequences correspond to the attributes IV, SV, OCC, MB, FM, OPR and BC, respectively.

**Table 2.** Comparison of our tracker with other state-of-the-art trackers on 11 different attributes of the OTB-2015 dataset. Average success scores (%) at a threshold of 0.5 are presented. The optimal results are highlighted in bold.

Attribute	Ours	BACF	TGPR	SAMF_AT	STC	MUSTer	Staple	LCT	KCF	MEEM	DSST
IV	<b>78.5</b>	78.4	53.1	62.5	35.6	70.8	71.9	70.9	53.3	58.8	65.2
SV	<b>73.6</b>	70.4	45	58.3	25.4	58.4	62.4	58.6	41.6	50.5	53.8
OCC	<b>72.3</b>	69.5	51.6	64.3	27.4	65.4	67.1	62.8	49.9	58.4	54.4
DEF	67.2	<b>69.2</b>	54.1	57.6	27.3	63	67.1	61.2	49.5	54.8	49.3
MB	<b>78.8</b>	71.2	51	70.3	19.5	64.4	65.1	64.7	53.9	65.9	54.6
FM	72.5	<b>74.4</b>	47	65	18.8	61.6	63	64.7	51.8	62.9	52.2
IPR	69.2	<b>69.7</b>	55.4	64	30.8	64.3	66.7	68.7	54	61.9	58.9
OPR	<b>71.5</b>	70.8	54.2	64.3	29.8	63.2	64.9	67.1	51.7	60.6	55.6
OV	64.1	<b>69.4</b>	45.2	61	22.5	54.1	56	53.1	45.7	56.8	44.2
BC	78.6	<b>79.6</b>	54.3	64.7	39.4	68.3	70.9	70.3	60.9	65.9	61.3
LR	<b>67.9</b>	65	38.1	56.7	22.6	44.2	45.9	48.4	25.1	33	41.9
Overall	76.3	<b>76.7</b>	53.4	67.7	31.4	68.1	70.6	69.7	54.5	61.8	60.4

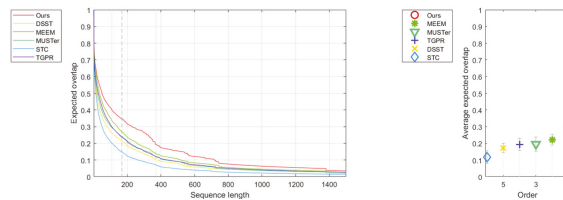
### 4.3.2. Experiment on the VOT-2015 Dataset

The VOT-2015 dataset [32] includes 60 video sequences. Using this dataset to test the performance of our algorithm by comparing it with the other five excellent trackers: TGPR [46], STC [20], MUSTer [16], MEEM [7], and DSST [11]. We reported the comparison results of average accuracy, robustness, and expected average overlap (EAO) to evaluate these trackers. Accuracy and robustness measures were based on the overlap ratio during successful tracking and the number of tracking failures per sequence, respectively. While the expected average overlap (EAO) is the new evaluation indicator for VOT-2015, this measure is based on empirical estimations of short-term sequence lengths. Table 3 presents the comparison results on the VOT-2015 in terms of accuracy, robustness, and expected average overlap. Our approach demonstrated a promising performance.

**Table 3.** Average ranks of accuracy, robustness, and expected average overlap under baseline experiments on the VOT-2015 dataset. The best three scores are highlighted in red, blue, and green, respectively.

Tracker	Acc. Rank	Rob. Rank	EAO
Ours	<b>1.35</b>	<b>1.43</b>	<b>0.2987</b>
MEEM	1.85	2.37	0.2212
MUSTer	1.67	2.20	0.1950
TGPR	2.12	2.67	0.1938
DSST	1.83	2.97	0.1719
STC	3.97	4.03	0.1179

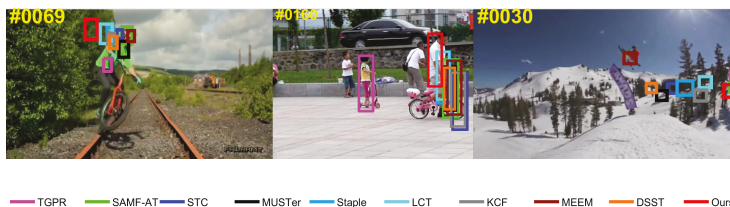
To further demonstrate that the performance of our tracker is superior to the five other state-of-the-art trackers on the VOT-2015 dataset, Figure 8 shows more intuitive comparison.



**Figure 8.** Expected average overlap curves and scores for the experiment baseline on the VOT-2015 dataset.

#### 4.4. Experimental Analysis

Our tracker achieved amazing results in many challenging scenarios. Especially in scenes where the target is partially occluded, the effective appearance of the remaining visible parts can still provide reliable cues for tracking. According to the coupling between the sub-regions and the global region, the complete structure of the target can be retained, and the outliers of the occluded sub-regions can be tolerated. This strategy can achieve effective tracking for solving the occlusion problem. However, the proposed algorithm did not perform well when faced with certain challenging attribute sequences (IPR, LR, and OV). In addition, when the sub-regions are completely occluded for long-term, our tracking framework did not effectively activate the tracker. The sampling frames for tracking failure are shown in Figure 9. There are three reasons for the flaws in our tracker. First, our algorithm does not solve the rotation problem of the target, so it cannot generate the rotated tracking bounding boxes for the IPR challenging attribute. Second, our framework lacks an occlusion re-identification scheme; therefore, when long-term occlusion occurs, the tracker cannot be active for a long time, causing the tracking to fail. The occlusion re-identification scheme incorporated into the tracking framework causes the tracker to skip the current occluded frame and calculate the tracking result from the next frame, which increases the adaptability of the discriminative correlation filter bank. Third, when a target in a low-resolution sequence is divided into multiple sub-regions, these sub-regions lack sufficient target feature information to train the robust discriminant correlation filter bank. Eventually, the tracking bounding boxes will not be able to effectively identify the target.



**Figure 9.** Failure cases on the OTB-2015 (from left to right: *Biker*, *Girl2*, and *Skiing*). In the *Biker* sequence, OV and LR are the most representative challenging attributes. The *Girl2* sequence contains long-term occlusion tracking challenges. In the *Skiing* sequence, the target undergoes LR and IPR during tracking.

## 5. Conclusions

In this paper, the discriminative correlation filter bank model is formed by combining multiple optimized correlation filters. We formulated multiple discriminative correlation filters corresponding to different sub-region and global patches simultaneously to achieve a robust tracking performance. By this means, the visible sub-regions can alleviate tracker drift when partial occlusion occurs. In addition, the sub-region patches used to train the correlation filters are zero-padded to the same size as the global target region to avoid noise aliasing during the correlation operation. Moreover, we used the ADMM optimization approach to iteratively train our correlation filters over time; this strategy will greatly improve the robustness of the tracker. Finally, we demonstrated the competitive accuracy and superior tracking performance of our method compared to state-of-the-art methods using the OTB-2015 and VOT-2015 datasets. In future work, we will study an effective occlusion detection model and incorporate this model into our tracking framework. When long-term occlusion occurs, the tracker can adaptively skip the occluded frame and calculate the tracking result from the next frame during the tracking process. Furthermore, the online adaptive update strategy will also be the focus of future work, because a real-time update tracker can greatly improve the accuracy of tracking for complex appearance changes.

**Author Contributions:** J.W. proposed the original idea, built the simulation model and completed the manuscript. F.L. modified and refined the manuscript.

**Funding:** This work was supported in part by the 1311 Talent Program of NJUPT, in part by the Natural Science Foundation of NJUPT under Grant NY214133, and in part by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Smeulders, A.W.M.; Chu, D.M.; Cucchiara, R.; Calderara, S.; Dehghan, A.; Shah, M. Visual Tracking: An Experimental Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 1442–1468. [[PubMed](#)]
2. He, S.; Yang, Q.; Lau, R.W.H.; Wang, J.; Yang, M.H. Visual Tracking via Locality Sensitive Histograms. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 2427–2434.
3. Jia, X.; Lu, H.; Yang, M.H. Visual tracking via adaptive structural local sparse appearance model. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 1822–1829.
4. Sevilla-Lara, L.; Learned-Miller, E. Distribution fields for tracking. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 1910–1917.
5. Babenko, B.; Yang, M.H.; Belongie, S. Visual tracking with online Multiple Instance Learning. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 983–990.
6. Kalal, Z.; Mikolajczyk, K.; Matas, J. Tracking-Learning-Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 1409–1422. [[CrossRef](#)] [[PubMed](#)]
7. Zhang, J.; Ma, S.; Sclaroff, S. MEEM: Robust tracking via multiple experts using entropy minimization. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 188–203.
8. Avidan, S. Support vector tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **2004**, *26*, 1064–1072. [[CrossRef](#)] [[PubMed](#)]
9. Avidan, S. Ensemble Tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 261–271. [[CrossRef](#)] [[PubMed](#)]
10. Bolme, D.S.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual object tracking using adaptive correlation filters. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 2544–2550.
11. Danelljan, M.; Häger, G.; Khan, F.; Felsberg, M. Accurate scale estimation for robust visual tracking. In Proceedings of the British Machine Vision Conference, Nottingham, UK, 1–5 September 2014; BMVA Press: Durham, UK, 2014.
12. Li, Y.; Zhu, J. A Scale Adaptive Kernel Correlation Filter Tracker with Feature Integration. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 254–265.
13. Danelljan, M.; Khan, F.S.; Felsberg, M.; Van de Weijer, J. Adaptive Color Attributes for Real-Time Visual Tracking. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1090–1097.
14. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. Exploiting the circulant structure of tracking-by-detection with kernels. In Proceedings of the European Conference on Computer Vision, Florence, Italy, 7–13 October 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 702–715.
15. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 583–596. [[CrossRef](#)] [[PubMed](#)]
16. Hong, Z.; Chen, Z.; Wang, C.; Mei, X.; Prokhorov, D.; Tao, D. MUlti-Store Tracker (MUSTer): A cognitive psychology inspired approach to object tracking. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 749–758.
17. Li, Y.; Zhu, J.; Hoi, S.C.H. Reliable Patch Trackers: Robust visual tracking by exploiting reliable patches. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 353–361.

18. Liu, T.; Wang, G.; Yang, Q. Real-time part-based visual tracking via adaptive correlation filters. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 4902–4912.
19. Ma, C.; Yang, X.; Zhang, C.; Yang, M.H. Long-term correlation tracking. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 5388–5396.
20. Zhang, K.; Zhang, L.; Liu, Q.; Zhang, D.; Yang, M.H. Fast visual tracking via dense spatio-temporal context learning. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 127–141.
21. Danelljan, M.; Robinson, A.; Khan, F.S.; Felsberg, M. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; Springer: Cham, Switzerland, 2016; pp. 472–488.
22. Galoogahi, H.K.; Sim, T.; Lucey, S. Multi-channel Correlation Filters. In Proceedings of the 2013 IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 3072–3079.
23. Danelljan, M.; Häger, G.; Khan, F.S.; Felsberg, M. Adaptive Decontamination of the Training Set: A Unified Formulation for Discriminative Visual Tracking. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 1430–1438.
24. Bibi, A.; Mueller, M.; Ghanem, B. Target response adaptation for correlation filter tracking. In Proceedings of the European Conference on Computer Vision, Seattle, WA, USA, 27–30 June 2016; Springer: Cham, Switzerland, 2016; pp. 419–433.
25. Danelljan, M.; Hager, G.; Khan, F.S.; Felsberg, M. Learning Spatially Regularized Correlation Filters for Visual Tracking. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 4310–4318.
26. Galoogahi, H.K.; Sim, T.; Lucey, S. Correlation filters with limited boundaries. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 4630–4638.
27. Ma, C.; Huang, J.B.; Yang, X.; Yang, M.H. Hierarchical Convolutional Features for Visual Tracking. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 3074–3082.
28. Danelljan, M.; Häger, G.; Khan, F.S.; Felsberg, M. Convolutional Features for Correlation Filter Based Visual Tracking. In Proceedings of the 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), Santiago, Chile, 7–13 December 2015; pp. 621–629.
29. Danelljan, M.; Bhat, G.; Khan, F.S.; Felsberg, M. ECO: Efficient Convolution Operators for Tracking. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6931–6939.
30. Liu, S.; Zhang, T.; Cao, X.; Xu, C. Structural Correlation Filter for Robust Visual Tracking. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 4312–4320.
31. Wu, Y.; Lim, J.; Yang, M.H. Object Tracking Benchmark. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1834–1848. [[CrossRef](#)] [[PubMed](#)]
32. Kristan, M.; Matas, J.; Leonardis, A.; Felsberg, M.; Cehovin, L.; Fernandez, G.; Vojir, T.; Hager, G.; Nebehay, G.; Pflugfelder, R. The Visual Object Tracking VOT2015 Challenge Results. In Proceedings of the 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), Santiago, Chile, 7–13 December 2015; pp. 564–586.
33. Kristan, M.; Roman, P.; Jiri, M.; Luka, Č.; Georg, N.; Tomáš, V.; Gustavo, F.; Alan, L.; Aleksandar, D.; Alfredo, P.; et al. The Visual Object Tracking VOT2014 Challenge Results. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 191–217.
34. Kristan, M.; Roman, P.; Jiri, M.; Luka, Č.; Georg, N.; Tomáš, V.; Gustavo, F.; Alan, L.; Aleksandar, D.; Alfredo, P.; et al. The Visual Object Tracking VOT2016 Challenge Results. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; Springer: Cham, Switzerland, 2016; pp. 777–823.

35. Kristan, M.; Matas, J.; Leonardis, A.; Felsberg, M.; Cehovin, L.; Fernandez, G.; Vojir, T.; Hager, G.; Nebehay, G.; Pflugfelder, R. The Visual Object Tracking VOT2017 Challenge Results. In Proceedings of the 2017 IEEE International Conference on Computer Vision Workshop (ICCVW), Venice, Italy, 22–29 October 2017; pp. 1949–1972.
36. Wu, Y.; Lim, J.; Yang, M.H. Online Object Tracking: A Benchmark. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 2411–2418.
37. Danelljan, M.; Häger, G.; Khan, F.S.; Felsberg, M. Discriminative Scale Space Tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1561–1575. [[CrossRef](#)] [[PubMed](#)]
38. Galoogahi, H.K.; Fagg, A.; Lucey, S. Learning Background-Aware Correlation Filters for Visual Tracking. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 1144–1152.
39. Liu, T.; Wang, G.; Yang, Q.; Wang, L. Part-based Tracking via Discriminative Correlation Filters. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, in press. [[CrossRef](#)]
40. Akin, O.; Erdem, E.; Erdem, A.; Mikolajczyk, K. Deformable part-based tracking by coupled global and local correlation filters. *J. Vis. Commun. Image Represent.* **2016**, *38*, 763–774. [[CrossRef](#)]
41. Fan, H.; Xiang, J. Local-Global Correlation Filter Tracking. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, in press.
42. Wang, S.; Wang, D.; Lu, H. Tracking with Static and Dynamic Structured Correlation Filters. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, in press. [[CrossRef](#)]
43. Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; Eckstein, J. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Found. Trends Mach. Learn.* **2011**, *3*, 1–22. [[CrossRef](#)]
44. Zhang, L.; van der Maaten, L. Preserving Structure in Model-Free Tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 756–769. [[CrossRef](#)] [[PubMed](#)]
45. Fan, H.; Xiang, J. Robust Visual Tracking via Local-Global Correlation Filter. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17), San Francisco, CA, USA, 4–9 February 2017; pp. 4025–4031.
46. Gao, J.; Ling, H.; Hu, W.; Xing, J. Transfer learning based visual tracking with gaussian processes regression. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 188–203.
47. Bertinetto, L.; Valmadre, J.; Golodetz, S.; Miksik, O.; Torr, P.H.S. Staple: Complementary Learners for Real-Time Tracking. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 1401–1409.



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# The Kernel Based Multiple Instances Learning Algorithm for Object Tracking

Tiwen Han <sup>†</sup>, Lijia Wang <sup>\*,†,‡</sup> and Binbin Wen <sup>†</sup>

Department of Intelligent Manufacture, Hebei College of Industry and Technology, Shijiazhuang 050091, China; hantiwen@163.com (T.H.); ge\_wenbinbin@163.com (B.W.)

\* Correspondence: wanglijia1981@hotmail.com; Tel.: +86-186-109-05016

<sup>†</sup> These authors contributed equally to this work.

<sup>‡</sup> Current address: Hebei College of Industry and Technology, Shijiazhuang 050091, China.

Received: 24 April 2018; Accepted: 13 June 2018; Published: 16 June 2018

**Abstract:** To realize real time object tracking in complex environments, a kernel based MIL (KMIL) algorithm is proposed. The KMIL employs the Gaussian kernel function to deal with the inner product used in the weighted MIL (WMIL) algorithm. The method avoids computing the pos-likely-hood and neg-likely-hood many times, which results in a much faster tracker. To track an object with different motion, the searching areas for cropping the instances are varied according to the object's size. Furthermore, an adaptive classifier updating strategy is presented to handle with the occlusion, pose variations and illumination changes. A similar score range is defined with respect to two given thresholds and a similar score from the second frame. Then, the learning rate will be set to be a small value when a similar score is out of the range. In contrast, a big learning rate is used. Finally, we compare its performance with that of the state-of-art algorithms on several classical videos. The experimental results show that the presented KMIL algorithm is faster and robust to the partial occlusion, pose variations and illumination changes.

**Keywords:** object tracking; kernel based MIL algorithm; Gaussian kernel; adaptive classifier updating

## 1. Introduction

Object tracking is a fundamental task in the fields of surveillance, robotics, human computer interaction, and so on. Recently, researchers have proposed many successful algorithms. However, the problem is still challenging due to factors such as occlusions, appearance variations, abrupt motion, and illumination changes [1].

The existing object tracking algorithms can be mainly classified into two groups: the generative method and the discriminative method [2]. The generative object tracking is one of the important problems, which learns an object model in the first frame and detects the area with the most similar appearance in the successive frames [3]. The MS tracker [4], IVT tracker [5], and VTD tracker [6] are the famous generative object tracking algorithms.

The discriminative tracking method learns and updates a binary classifier by using online training. The tracking-by-detection algorithm stems directly from the discriminative methods, which trains a classifier online and finds the most likely location from many candidate image patches as tracking evolves [7–11]. The Online-Boosting tracker updates a classifier by considering the tracked result as the positive sample [10]. However, it often fails when the tracked results drift from the real object location. Then, an improved algorithm (semi-supervised tracker) is proposed by Gabor [11]. The algorithm labels the positive samples in the first frame. However, the ambiguity problem exists in the tracking process. Zhang [3] describes a real-time CT tracker by utilizing the compressed features for training a Bayes classifier. In recent years, the researchers focus mainly on studying the real-time object tracking algorithms. The correlation filter-based tracking algorithms are proposed and have provided excellent



performance. The algorithms include MOSSE [12], CSK [13], KCF [14], DCF [15], CN [16], DAT [17], Staple [18], and DSST [19]. These trackers have shown the advantage of being computationally efficient, which is especially useful for real-time applications. The KCF [14] tracker generates samples by applying a circulate matrix, which speeds up the computing of matrixes. As a result, the algorithm runs at hundreds of FPS [14]. However, it often suffers from drift problem due to the occlusion, illumination changes, and pose variations. The deep learning algorithm has been widely studied in the fields of computer vision including object tracking. The DeepSRDCF [20], GOTURN [21], C-COT [22], and ECO [23] benefiting from big data for learning a net model are proposed for object tracking. The experimental results have addressed that the obtained convolution features have a powerful ability of representation. However, the processing of training networks is time consuming due to the complexity network. Normally, to realize real-time object tracking, the deep learning algorithm runs on GPU and can achieve about 100 FPS (e.g., the GOTURN tracker). However, the GOTURN operates at 2.7 FPS for the only CPU [21]. To overcome the ambiguity problem in the online-boosting algorithm, the MIL tracker-related algorithms are proposed to learn a strong classifier from multiple instances in the positive and negative bags [24,25]. The WMIL tracker [25] runs at about 14FPS on a single CPU. Moreover, the WMIL tracker performs well over the CF related trackers in terms of occlusion, illumination variations and pose changes.

In this paper, we propose a Kernel based MIL (KMIL) object tracking algorithm. To further reduce the computational cost, the Gaussian kernel function is presented for resolving the inner product used in the WMIL algorithm. The WMIL algorithm often fails to track the object with different speed. To deal with the problem, the searching areas for cropping the instances are varied according to the object's size. Finally, an adaptive classifier updating strategy is presented. The similar score of the tracking result in the second frame is remembered as a reference. Then, two thresholds are defined and a range is obtained with respect to the reference. As tracking evolves, the updating rate of the classifier is adjusted to suit for the appearance changes when the maximum similar score of the sample is out of the range at the current frame. At last, the proposed algorithm is compared with the state-of-art algorithms.

The paper is organized as follows. The Section 1 is the introduction of the paper. The WMIL tracker is detailed in the Section 2. The Section 3 addresses the KMIL tracker. The experimental results are shown in the Section 4. The Section 5 summaries the paper.

## 2. The WMIL Tracker

Babenko [24] proposed an online MIL Boosting method for visual tracking. It detects an object by maximizing the bag likelihood function. Consequently, it suffers from being time consuming because the bag probability and instance probability are computed many times before selecting a most discriminative weak classifier. To deal with this problem, Zhang [25] proposed an efficient online approach (WMIL algorithm) to approximately maximize the bag likelihood function. In the algorithm, "positive" and "negative" bags are extracted for training classifiers. The positive bag is constructed by using the instances extracted from a circle centered at the object's location, while the negative bag is obtained by cropping the instances from an annular region around the object's location. Then, a strong classifier is trained in the Online Boosting frame by using the positive and negative bags. In the successive frame, candidate samples are cropped around the object's position in the previous frame. At last, the trained classifier detects the candidate sample with the maximum score as the final result. Furthermore, to deal with the problem of occlusion, illumination changes, and pose variations, the classifier is updated by using the positive and negative bags constructed by cropping instances according to the tracked result in the current frame.

It is assumed that the location of each instance  $x$  is denoted as  $I_t(x)$  and the location of the object is denoted as  $I_t^*$  in the  $t^{th}$  frame. The instances for constructing a positive bag  $X^+$  are cropped as:  $X^+ = \{x : ||I_t(x) - I_t^*|| < r\}$ ,  $r$  is the searching radius centered as the location  $I_t^*$ , while the instances for constructing a negative bag  $X^-$  are cropped from an annular region  $X^- = \{x : r < ||I_t(x) - I_t^*|| < \beta\}$ ,

$r$  and  $\beta$  ( $r < \beta$ ) are the radius of the annular region. In the Online Boosting frame, the algorithm trains  $K$  weak classifiers  $\phi = \{h_1, h_2, \dots, h_K\}$ , which is defined as:

$$h_k(x) = \ln \frac{p(v_k(x)|y = 1)}{p(v_k(x)|y = 0)} \tag{1}$$

where  $v(x) = (v_1(x), \dots, v_k(x), \dots, v_K(x))^T$  is a feature vector function of the instances.  $y_i$  is the label of the bag  $X_i$ .  $y = 1$  means the bag is positive, while  $y = 0$  means the bag is negative.

Then,  $M$  discriminative weak classifiers are selected to generate a strong classifier:

$$h(\cdot) = \sum_{k=1}^M h_k(\cdot) \tag{2}$$

The learned strong classifier detects an area with the maximum similar score as the final tracking location :

$$l_i^* = l(\arg \max_{x \in X^s} p(y|x)) \tag{3}$$

After detecting an object, the weak classifiers are updated according to the new location with a constant learning rate:

$$\sigma_i^1 = \sqrt{\lambda(\sigma_i^1)^2 + (1-\lambda)\frac{1}{N} \sum_{j=0|y_i=1}^{N-1} (v_k(x_{ij}) - \bar{\mu})^2 + \lambda(1-\lambda)(\mu_i^1 - \bar{\mu})^2} \tag{4}$$

where  $\lambda$  is the learning rate.  $\bar{\mu} = \frac{1}{N} \sum_{j=0|y_i=1}^{N-1} (v_k(x_{ij}))$  is the average of the  $k$ th feature  $v_k(x_{ij})$  of the instances in the positive bag extracted around the tracking result at current frame.

### 3. The KMIL Object Tracking System

This section details the presented KMIL object tracking algorithm illustrated in Figure 1. Postive and negative bags are extracted around the object’s location. Then, a strong classifier is learned from the first frame by using the online boosting WMIL algorithm [25], and the object image is saved as a reference. In the successive frame, the classifier is used to detect the most similar sample as the tracking result. Finally, the classifiers are updated according to the reference frame and current tracking result.

In the WMIL algorithm, the inner product is presented for selecting a weak classifier with the most discriminative ability, which reduces the computational time by avoiding computing the bag probability and instance probability many times [25]. However, the inner product is also computed  $M$  times, which is also time consuming. Recently, the kernel based approaches have been proposed for real time object tracking [26]. Inspired by the ideas in the WMIL [25] and DLSSVM [26] algorithms, we present a kernel based inner product method to select the most discriminative weak classifiers to further reduce the computational complexity.

As tracking evolves, the sample with the maximum similar score in the candidate area is detected by using the learned classifier. Normally, we assume that the object moves with the same speed and appearances around the object location from the previous frame. Therefore, the candidate samples are extracted in a fixed circle around the previous tracking location. To account for the size of an object, we change the circle adaptively. After tracking an object, the weak classifiers are updated with the new cropped samples for handling the appearance changes. Normally, a constant learning rate is used, which may lead to “over-updating” or “less-updating”. To further handle these problems, an adaptive weak classifiers strategy are presented.

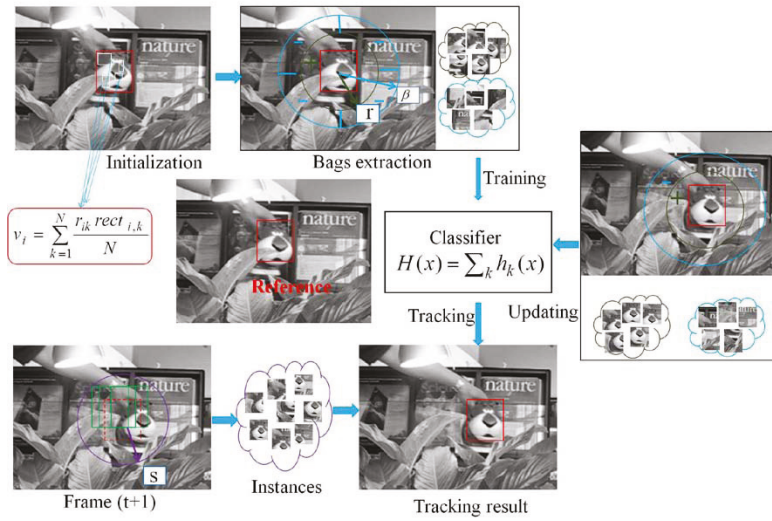


Figure 1. The flow chart of the Kernel based MIL algorithm.

### 3.1. The Kernel Based MIL Tracker

We assume that there are  $N$  instances in the positive bag and  $L$  instances in the negative bag. The tracking location at current frame is denoted as  $l_{10}$ . The positive bag and negative bag  $\{X^+, X^-\}$  are considered as the training data. Different from the method of computing bag probability used in the WMIL algorithm, we compute the bag probability with respect to the included instances' probability.

$$p(y = 1|X^+) = \frac{1}{N} \sum_{j=0}^{N-1} p(y = 1|x_{1j}) \tag{5}$$

The method means that each instance contributes equally to the bag probability according to the including instances probability. Then, the bag probability mainly depends on the instances with higher probability. Especially in the case of tracking drift, the instances near the real object but far away from the center of the previous location contribute more to the bag probability.

Similar to the positive bag, the probability of the negative bag is computed with respect to the including instances equally.

$$p(y = 1|X^-) = \frac{1}{L} \sum_{j=N}^{N+L-1} p(y = 0|x_{0j}) \tag{6}$$

Similar to the WMIL tracker, our KMIL tracker trains weak classifiers  $\phi = \{h_1, h_2, \dots, h_K\}$ , and selects the most discriminative weak classifiers by using an efficient criterion [25].

$$h_k = \arg \max_{h \in \phi} \langle h, \nabla \eta(H) \rangle |_{H=H_{k-1}} \tag{7}$$

where  $h$  is the weak classifier in the classifier pool and  $H$  is the learned strong classifier constructed by  $K - 1$  selected weak classifiers.

The inner product is computed as:

$$\langle h, \nabla \eta(H) \rangle = \frac{1}{N+L} \sum_{j=0}^{N+L-1} h(x_{ij} \nabla \eta(H)(x_{ij})). \tag{8}$$

$$\eta = \sum_{s=0}^1 (y_s \log(p(y = 1|X^+)) + (1 - y_s) \log(p(y = 0|X^-))) \tag{9}$$

$$\eta(H) = \sum_{s=0}^1 (y_s \log(\sum_{j=0}^{N-1} p(y = 1|x_{1j})) + (1 - y_s) \log(\sum_{j=N}^{N+L-1} (1 - p(y = 1|x_{0j})))) \tag{10}$$

$$\begin{aligned} \eta(H)(x_{ij}) &= y_i \frac{w_{j0} \sigma(H(x_{ij}))(1 - \sigma(H(x_{ij})))}{\sum_{m=0}^{N-1} w_{j0} \sigma(H(x_{im}))} \\ &= -(1 - y_i) \frac{\sigma(H(x_{ij}))(1 - \sigma(H(x_{ij})))}{\sum_{m=N}^{N+L-1} (1 - \sigma(H(x_{im})))} \end{aligned} \tag{11}$$

where  $\sigma(z) = \frac{1}{1+e^{-z}}$  is the sigmoid function and  $\sigma(H(x_{ij}))$  is the  $j$ th instance probability in the  $i$ th bag.

The results from the WMIL tracker [25] shown that the criterion is efficient because it can avoid computing the bag probability and instance probability  $K$  times before selecting a weak classifier. Therefore, it is more efficient than the log-likelihood function used in the MIL tracker [25].

However, there is higher dimension computing in the inner product  $\frac{1}{N+L} \sum_{j=0}^{N+L-1} h(x_{ij} \nabla \eta(H)(x_{ij}))$ , which is also time consuming. To handle with the problem, we use the kernel function. The inputs  $h$  and  $\nabla \eta(H)$  are mapped to the feature space by using  $\phi(h)$  and  $\phi(\nabla \eta(H))$ , where  $\phi(\cdot)$  is the Hilbert mapping of the inputs. The kernel is defined as:

$$k(h, \nabla \eta(H)) = \phi(h)^T \phi(\nabla \eta(H)) \tag{12}$$

In practice, we choose to use the Gaussian kernel [26]:

$$k(h, \nabla \eta(H)) = \exp\left(-\frac{\|h - \nabla \eta(H)\|^2}{\rho^2}\right) \tag{13}$$

where  $\rho$  is the bandwidth of the Gaussian function.

As new frames come, candidate samples are extracted around the tracking result:  $X^s = \{x : \|I_{t+1}(x) - I_t^*\| < s\}$ , where  $s(r < s < \beta)$  is the radius. Then, the learned strong classifier detects the sample with the maximum similar score as tracking result. In the WMIL, MIL, and CT trackers, it is assumed that the object moves around the previous tracking location. And the candidate samples are cropped in a fixed circle. These trackers have shown continuous performance in term of accuracy on tracking large object with continuous motion. However, they often fail to track a small object because of their fast motion. To deal with the problem, we present a method to vary the radius for extracting the candidate sample with respect to the target's size. The radius is set to be 25 if the object is big. On the contrary, the radius is 35 for tracking the small object.

### 3.2. The Classifiers Update Strategy

After detecting an object, the classifier is updated to deal with the problems of occlusion, pose variations, and illumination changes. Normally, a learning rate is set to make a balance between the previous frame and the current frame [25]. We have tried the updating method with a fixed learning rate and found that the experimental results are unstable when there are appearance variations. With a small learning rate, the parameters of the classifier will be updated mainly with the mean and variance of the new tracked location's features. As a result, the interference from background will be introduced to update the classifier, which results in "over-updating". If the learning rate is too large, the new tracked area will affect the parameters of the classifier rarely. Then, the classifier will be "less-updating" and can't deal with the illumination changes and pose variations.

To address the problem mentioned above, we present an adaptive classifier updating strategy. From experimental results of the WMIL tracker, we found that the similar scores of the tracking results vary frequently from the tracked locations in the beginning frames. Therefore, we define the similar score  $S_0$  of the tracking location in the second frame as a reference. Two thresholds  $H_1$  and  $H_2$  are also defined. Then, a similar score range ( $S_0 - H_1, S_0 + H_2$ ) is obtained with respect to the two thresholds and the reference. As tracking evolves, we consider the tracking results with the similar score outside the defined range as the appearance changes case. Then, a large learning rate is defined for updating the parameters of the classifier. If the similar score of tracking results are within the defined range, we use a small learning rate to update the classifier. The learning rate is defined as follows:

$$r = \begin{cases} 0.25 & \text{if } s \in (S_0 - H_1, S_0 + H_2) \\ 0.85 & \text{if } s \notin (S_0 - H_1, S_0 + H_2) \end{cases} \quad (14)$$

#### 4. Experiments

We compared KMIL tracker with the state-of-art object tracking algorithms, such as MIL [7], CT [3], WMIL [25], KCF [14], and DSST [19]. The binary code released by the authors are used for testing these trackers. The videos "Tiger2", "Lemming", "Shaking", "Deer", "Sylvester", "Faceocc1", "Tiger1", and "Football1" from the OTB25 database are downloaded from the Network for testing these trackers. There are illumination changes and pose variations in the "Tiger1", "Tiger2", "Sylvester" and "Lemming" sequences. The serious occlusions exist in the sequences "Occluded face", "Football1", "Shaking" and "Lemming" videos. The objects move fast and vary their pose frequently in the "Tiger2", "Tiger1", and "Lemming" sequences. All the algorithms are implemented in the MATLAB and run on a core 2 CPU, 2.33GHz and 2GB RAM computer.

##### 4.1. Parameters Setting

In these algorithms, the parameters  $r$ ,  $\alpha$ , and  $\beta$  determine the instances in the positive and negative bags. The  $s$  determines the area for cropping the candidate samples. The algorithms with bigger  $r$ ,  $\alpha$ ,  $\beta$ ,  $s$  can extract more instances which make the algorithms perform well to track an object, but result in time consuming. The  $K$  is the number of the weak classifiers, while  $M$  is that of the selected discriminative weak classifier for constructing a strong classifier. The classifier with bigger  $K$  and  $M$  can discriminate an object easily, which also lead to computing complexity. The parameters are set to be the same as the presented papers [3,7,19,25], which are illustrated in Table 1. The results of these algorithms have shown that these algorithms perform the best with the parameters. In the KMIL tracker, the radius for extracting candidate samples is set to be 25. In contrast, the radius is 35. Different from the fixed learning rate of the CT, MIL, WMIL trackers, the learning rate of the KMIL tracker is adapted according to the maximum score of the candidate sample at current frame. When the maximum score is in a given range, the learning rate is set to be 0.25, or else it is 0.85. The number of weak classifiers is 150, while that of the selected discriminative weak classifiers is 50. The KCF tracker employs the HOG feature and Gaussian kernel. In the first frame, a model is learned with the image patch centered at the initial position. In the successive frames, the position with the maximum value is detected over the candidate patch. Finally, a new model is trained at the new tracking position [14].

The CT, MIL, WMIL, KMIL, KCF, DSST trackers are implemented on the mentioned videos. After training the classifiers, the area with the maximum similar score is considered as the tracking result. Then, the classifiers are updated to overcome the drawbacks of occlusion, pose variations, and illumination variations.

**Table 1.** The parameters for the MIL, CT, WMIL and KMIL trackers.  $r$  is the radius for positive bag,  $\alpha$  and  $\beta$  are the radius for the negative bag,  $s$  is for the searching area,  $K$  is the number of the weak classifiers,  $M$  is the number of the selected most discriminative weak classifiers.  $\lambda$  is the learning rate.

Parameters	$r$	$\alpha$	$\beta$	$s$	$K$	$M$	$\lambda$
CT	4	8	30	20	/	/	0.85
MIL	4	/	50	35	250	50	0.85
WMIL	4	$\alpha = 2r$	$\beta = 1.5s$	25	150	15	0.85
KMIL(big object)	4	$\alpha = 2r$	$\beta = 1.5s$	25	150	15	0.25/0.85
KMIL(small object)	4	$\alpha = 2r$	$\beta = 1.5s$	35	150	50	0.25/0.85

#### 4.2. Tracking Object Location

Here, we detail the tracking object locations of the above trackers evaluated on the eight classical videos. The results are shown in Figure 2. The MIL tracker is time consuming because of its classifier selecting strategy. As a result, it often suffers from failure for the long time object tracking. The KCF tracker uses the circulate matrix and kernel function for completing real time object tracking. However, its searching area and learning rate are constant. Therefore, it often fails to track the object in the complex environment, especially when the object is small and moves fast. The WMIL and CT trackers update the classifier with a constant learning rate. Therefore, they result in tracking drift in the complex environment. Furthermore, the WMIL tracker computes the bag probability according to instances' distance. Consequently, the tracking drift will be aggregated. Benefiting from the constant learning rate, adaptive searching radius and the bag probability, the KMIL tracker can deal with the drift problem in the complex environment. The tracking object locations in Figure 2 demonstrate that the KMIL tracker performs well over the CT, MIL, WMIL, KCF, DSST trackers.



**Figure 2.** The illustration of the tracking locations on the sequences: “Deer”, “Tiger2”, “Faceocc2”, “Sylvester”, “Football1”, “Shaking”, “Tiger1”, “Lemming”.

#### 4.3. Quantitative Analysis

We use the precision curves to evaluate the performance of the proposed algorithm. The precision curves illustrate the percentage of correctly tracked frames for a range of distance thresholds [14]. The correctly tracked frame is the one with target center within a distance threshold of the ground truth. The tracker with a higher precision at low threshold is more accurate. Similar to the previous

works [7,13,27], we choose the 20 pixels as the threshold. The experimental results are shown in Figure 3. From the experimental results, we found that the precision of the KMIL tracker is higher than the other algorithms at 20 pixels for the “Tiger2”, “Lemming”, “Deer”, “Slyvery”, and “Faceoccl1” sequences. For the “Tiger1” sequence, the KMIL tracker achieves the second higher precision at the 20 pixels. The MIL tracker has the highest precision for tracking the “Football man” shown in the Figure 3h at 20 pixels. All of the algorithms can achieve 1 precision at 35 pixels. The experimental results have shown the effectiveness of the proposed KMIL tracker. Especially, the KMIL tracker is more efficient in term of precision than the popular kernel based trackers KCF and DSST.

Another performance criteria we chose for our evaluation is the success plot. In the tracking process, the tracking result is denoted as a bounding box (called  $a$ ) and the real position is the ground truth (called  $b$ ). Then, an overlap score is defined with respect to the two regions.

$$OS = \frac{|a \cap b|}{|a \cup b|} \tag{15}$$

where  $\cap$  and  $\cup$  mean the intersection and union, respectively.  $|\cdot|$  counts the number of the pixels. The tracked targets with the overlap score larger than the given threshold (0.5 is used) are considered as the successful results. The success curve is the ratios of the success frames to the whole frames. The experimental results are shown in Figure 4. The higher the success curve is, the stronger tracking ability the tracker has. It shows that the “red” line obtained by using the KMIL tracker on the sequences: “Tiger2”, “Lemming”, “Shaking”, “Deer”, “Sylvester”, “Faceoccl1”, and “Tiger1” is higher than other line. In the Figure 4h, the higher success curve is obtained by using the MIL tracker on the “Football1” sequences. However, it is time consuming, which will be detailed in the next section. Furthermore, the well known fast KCF and DSST trackers run at a low success rate especially for tracking the small object (e.g., on the “Tiger1” sequence).

#### 4.4. Computational Cost

This section details the computational cost of the above trackers. The average computing time processing an image is defined as:  $t_{avr} = \frac{t_{all}}{N_{fra}}$ .  $t_{all}$  is the total computing time processing all of the images in the whole video sequence.  $N_{fra}$  is the total number of frames.  $t_{avr}$  is the obtained average computing time. There are four factors lead to computational cost in the CT, MIL, WMIL, and KMIL trackers. The first factor is the total number of weak classifiers in the classifier pool. The number of the selected discriminative weak classifiers is the second factor influencing the computational time. The number of the instances due to big searching area also results in computational complexity. At last, the method for selecting discriminative weak classifiers which computes high dimension matrix is also time consuming. We did experiments by using the parameters in the Table 1. The Frames Per Second (FPS) of all the trackers are illustrated in Table 2. The KCF and DSST trackers run faster than the MIL, CT, WMIL, and KMIL trackers. However, it has low precision and success rate. Benefiting from the kernel function, the KMIL tracker avoids computing the  $(h(x_{ij}) \nabla \eta(H(x_{ij})))M$  times. As a result, it is with lower computational time than the MIL, WMIL trackers.

**Table 2.** The FPS (Frames Per Second) for different algorithms conducted on sequences “Tiger2”, “Lemming”, “Shaking”, “Deer”, “Sylvester”, “Faceoccl1”, “Tiger1”, “Football1”.

Video Clip	MIL	CT	WMIL	KCF	KMIL	DSST
Tiger2	3.52	10.14	7.34	54.33	9.96	260
Lemming	3.13	9.41	8.96	35.73	9.98	103
Shaking	3.38	13.03	14.69	30.12	18.14	279
Animal	3.52	11.40	8.87	28.76	10.25	479
Sylvester	3.65	13.32	7.98	42.31	14.37	137
Faceoccl2	3.46	13.21	13.85	38.28	17.70	260
Tiger1	3.02	10.4	7.93	10.94	9.22	265
Football1	3.73	13.59	8.77	224	13.04	500

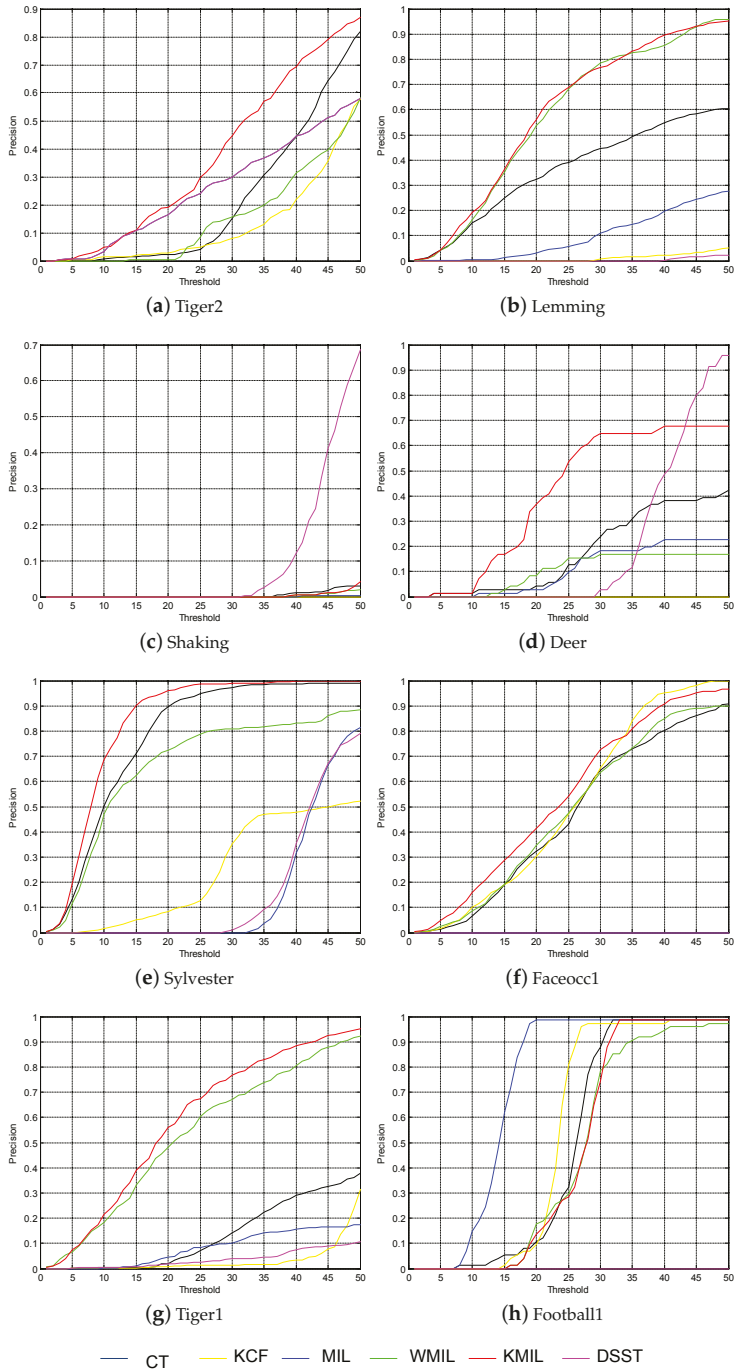


Figure 3. Precision plot for sequences: “Tiger2”, “Lemming”, “Shaking”, “Deer”, “Sylvester”, “Faceoccl”, “Tiger1”, “Football1”.



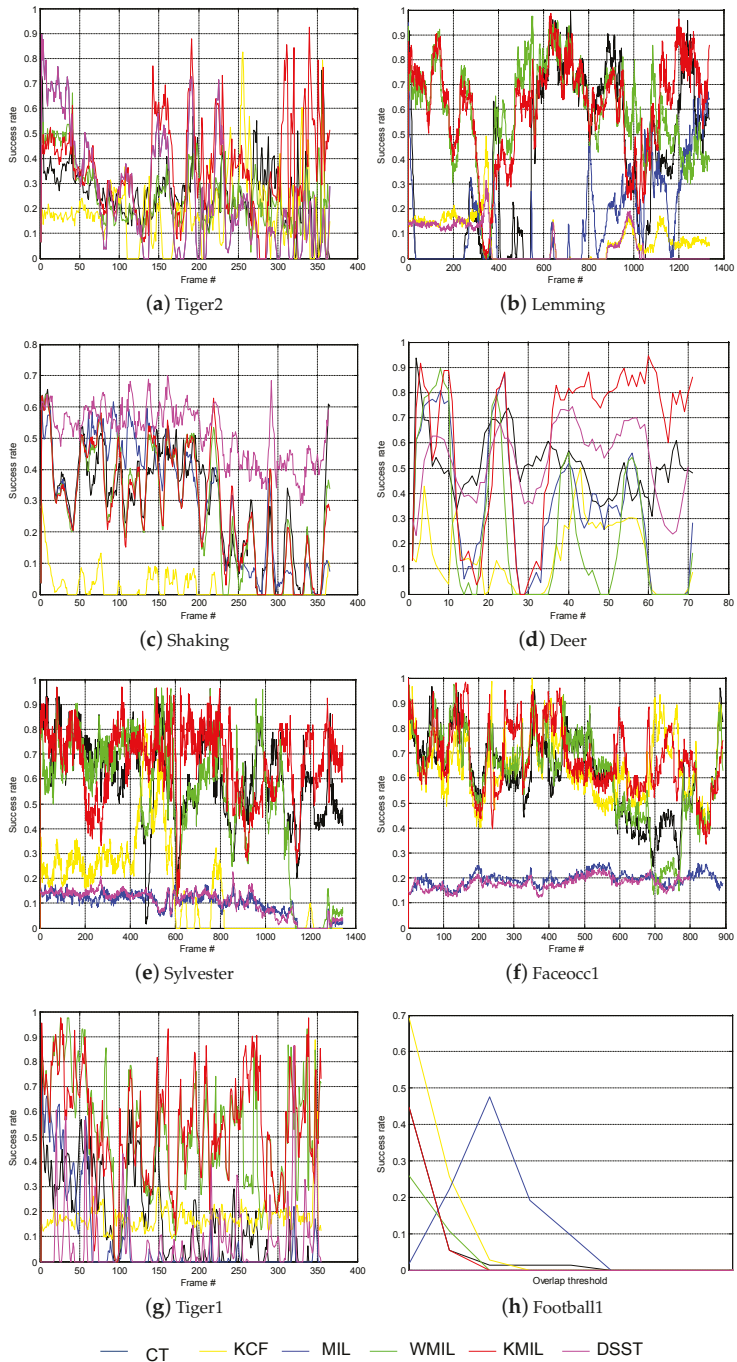


Figure 4. Success plot for sequences: “Deer”, “Tiger2”, “Faceocc2”, “Sylvester”, “Football1”, “Shaking”, “Tiger1”, “Lemming”.

## 5. Conclusions

In this paper, we revisit the core of the WMIL formulation to counter the issues of computation complexity and drift problem. We introduce a Gaussian kernel based multiple instance learning algorithm for real-time vision applications. We also suggest a simple yet effective searching circle update strategy that is especially suitable for small but moving fast objects. Lastly, we also present a classifier update method for handling the appearance changes with respect to two thresholds and reference similar score. The experiments conducted on several classical videos demonstrated that the KMIL tracker was efficient in terms of time-consumption and robustness.

**Author Contributions:** Conceptualization, T.H., L.W.; Methodology, L.W., T.H.; Software, B.W.; Validation, L.W., B.W.; Formal Analysis, B.W.; Investigation, B.W.; Resources, B.W.; Data Curation, B.W.; Preparation, L.W., B.W.; Writing, T.H., B.W.; Visualization, L.W.; Supervision, T.H.; Funding Acquisition, T.H.

**Funding:** This research was funded by [the key research and development program of the Hebei province science] grant number [18210329D]; [the Natural Science Foundation of Hebei Province] grand number [F2018205178]; [the Hebei College of Industry and Technology Natural Science Program] grand number [ZRY2017004].

**Acknowledgments:** Visual Tracker Benchmark [http://cvlab.hanyang.ac.kr/tracker\\_benchmark/datasets.html](http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html).

**Conflicts of Interest:** The authors declare no conflict of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

## Abbreviations

The following abbreviations are used in this manuscript:

MS	Mean Shift
IVT	Incremental Visual Tracking
VTD	Visual Tracking Decomposition
MOSSE	Minimum Output Sum of Squared Errors
CSK	Circulant Structure with Kernels
CN	Color Names
DCF	Discriminative Correlation Filter
DAT	Distractor Aware Tracking
C-COT	Continuous Convolution Operator
DeepSRDCF	Deep Spatially Regularized Discriminative Correlation Filter
DLSSVM	Dual Linear Structure Support Vector Machine
ECO	Efficient Convolution Operators
CT	Compressive Tracking
WMIL	Weighted Multiple Instance Learning
KMIL	Kernel based Weighted Multiple Instances Learning
KCF	Kernelized Correlation Filter
DSST	Discriminative Scale Space Tracking

## References

1. Yilmaz, A. Object tracking: A survey. *ACM Comput. Surv.* **2006**, *38*, 13. [[CrossRef](#)]
2. Gao, J.; Ling, H.; Hu, W.; Xing, J. Transfer Learning Based Visual Tracking with Gaussian Processes Regression. In Proceedings of the 13th European Conference, Zurich, Switzerland, 6–12 September 2014; pp. 188–203. [[CrossRef](#)]
3. Zhang, K.; Zhang, L.; Yang, M.H. Real-time compressive tracking. In Proceedings of the 12th European Conference on Computer Vision, Florence, Italy, 7–13 October 2012; pp. 864–877. [[CrossRef](#)]
4. Dadgostar, F.; Sarrafzadeh, A.; Overmyer, S.P. Face Tracking Using Mean-Shift Algorithm: A Fuzzy Approach for Boundary Detection. In Proceedings of the Affective Computing and Intelligent Interaction, First International Conference, (ACII 2005), Beijing, China, 22–24 October 2005; pp. 56–63. [[CrossRef](#)]

5. Ross, D.A.; Lim, J.; Lin, R.S. Incremental Learning for Robust Visual Tracking. *Int. J. Comput. Vis.* **2008**, *77*, 125–141. [[CrossRef](#)]
6. Kwon, J.; Lee, K.M. Visual tracking decomposition. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 1269–1276. [[CrossRef](#)]
7. Babenko, B.; Yang, M.H.; Belongie, S. Robust Object Tracking with Online Multiple Instance Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *33*, 1619–1632. [[CrossRef](#)] [[PubMed](#)]
8. Hare, S.; Saffari, A.; Torr, P.H.S. Struck: Structured output tracking with kernels. In Proceedings of the 2011 IEEE International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 263–270. [[CrossRef](#)]
9. Avidan, S. Support Vector Tracking. In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), Kauai, HI, USA, 8–14 December 2001; pp. 1184–1191. [[CrossRef](#)]
10. Grabner, H.; Grabner, M.; Bischof, H. Real-Time Tracking via On-line Boosting. In Proceedings of the British Machine Vision Conference 2006, Edinburgh, UK, 4–7 September 2006; pp. 47–56. [[CrossRef](#)]
11. Grabner, H.; Leistner, C.; Bischof, H. Semi-supervised On-Line Boosting for Robust Tracking. In Proceedings of the 10th European Conference on Computer Vision, Marseille, France, 12–18 October 2008; pp. 234–247. [[CrossRef](#)]
12. Bolme, D.S.; Beveridge, J.R.; Draper, B.A. Visual object tracking using adaptive correlation filters. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 2544–2550. [[CrossRef](#)]
13. Rui, C.; Martins, P.; Batista, J. Exploiting the circulant structure of tracking-by-detection with kernels. In Proceedings of the 12th European Conference on Computer Vision, Florence, Italy, 7–13 October 2012; pp. 702–715. [[CrossRef](#)]
14. Henriques, J.F.; Rui, C.; Martins, P. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *37*, 583–596. [[CrossRef](#)] [[PubMed](#)]
15. Danelljan, M.; Hager, G.; Khan, F.S. Learning Spatially Regularized Correlation Filters for Visual Tracking. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 4310–4318. [[CrossRef](#)]
16. Danelljan, M.; Khan, F.S.; Felsberg, M. Adaptive Color Attributes for Real-Time Visual Tracking. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1090–1097. [[CrossRef](#)]
17. Possegger, H.; Mauthner, T.; Bischof, H. In defense of color-based model-free tracking. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 2113–2120. [[CrossRef](#)]
18. Bertinetto, L.; Valmadre, J.; Golodetz, S. Staple: Complementary Learners for Real-Time Tracking. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; Volume 38, pp. 1401–1409. [[CrossRef](#)]
19. Danelljan, M.; Hager, G.; Khan, F.S. Discriminative Scale Space Tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1561–1575. [[CrossRef](#)] [[PubMed](#)]
20. Danelljan, M.; Hager, G.; Khan, F.S. Convolutional Features for Correlation filter-based Visual Tracking. In Proceedings of the 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), Santiago, Chile, 7–13 December 2015; pp. 621–629. [[CrossRef](#)]
21. Held, D.; Thrun, S.; Savarese, S. Learning to Track at 100 FPS with Deep Regression Networks. In Proceedings of the 14th European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 749–765. [[CrossRef](#)]
22. Danelljan, M.; Robinson, A.; Khan, F.S. Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking. In Proceedings of the 14th European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 472–488. [[CrossRef](#)]
23. Danelljan, M.; Bhat, G.; Khan, F. ECO: Efficient Convolution Operators for Tracking. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6931–6939. [[CrossRef](#)]

24. Babenko, B.; Yang, M.-H.; Belongi, S. Robust Object Tracking with Online Multiple Instance Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *33*, 1619–1632. [[CrossRef](#)] [[PubMed](#)]
25. Zhang, K.; Song, H. Real-time visual tracking via online weighted multiple instance learning. *Pattern Recognit.* **2013**, *46*, 397–411. [[CrossRef](#)]
26. Ning, J.; Yang, J.; Jiang, S.; Zhang, L.; Yang, M.-H. Object Tracking via Dual Linear Structured SVM and Explicit Feature Map. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 4266–4274. [[CrossRef](#)]
27. Wu, Y.; Lim, J.; Yang, M.-H. Online Object Tracking: A Benchmark. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 2411–2418. [[CrossRef](#)]



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# Predicting the Influence of Rain on LIDAR in ADAS

Christopher Goodin \*, Daniel Carruth, Matthew Doude and Christopher Hudson

Center for Advanced Vehicular Systems, Mississippi State University, Mississippi State, MS 39762, USA; dwc2@CAVS.MsState.Edu (D.C.); mdoude@CAVS.MsState.Edu (M.D.); chudson@cavs.msstate.edu (C.H.)

\* Correspondence: cgoodin@cavs.msstate.edu

Received: 19 December 2018; Accepted: 10 January 2019; Published: 15 January 2019

**Abstract:** While it is well known that rain may influence the performance of automotive LIDAR sensors commonly used in ADAS applications, there is a lack of quantitative analysis of this effect. In particular, there is very little published work on physically-based simulation of the influence of rain on terrestrial LIDAR performance. Additionally, there have been few quantitative studies on how rain-rate influences ADAS performance. In this work, we develop a mathematical model for the performance degradation of LIDAR as a function of rain-rate and incorporate this model into a simulation of an obstacle-detection system to show how it can be used to quantitatively predict the influence of rain on ADAS that use LIDAR.

**Keywords:** perception in challenging conditions; obstacle detection and classification; dynamic path-planning algorithms

## 1. Introduction

Among the many challenges involved in the development of safe, reliable advanced driver assist systems (ADAS), sensing and perception in adverse weather remains one of the most difficult problems. In fact, a recent article in Bloomberg magazine entitled “Self-Driving Cars Can Handle Neither Rain nor Sleet nor Snow” claimed that “The ultimate hurdle to the next phase of driver-less technology might not come from algorithms and artificial intelligence—it might be fog and rain [1]”.

The primary technical challenges associated with automated and autonomous driving in rain come from the influence of rain on the vehicles sensors such as cameras and LIDAR. Although the qualitative impacts of weather on these sensors has been studied for quite some time [2], there has been surprisingly little progress in quantitatively predicting the impact of rain on LIDAR sensors typically used in ADAS systems.

Such a model would be useful in both defining a performance envelope for ADAS systems and in the development of weather-aware ADAS algorithms. Ideally, the model would depend on simple environment parameters such as rain rate and simple sensor parameters such as laser power. While recent measurements and a resulting empirical model of the influence of heavy rain on a Hokuyo UTM-30LX-EW were published by [3], their work measured rain rates of 40.5–95.4 mm/h, whereas naturally occurring rain rarely exceeds 25 mm/h. Therefore, the empirical model is not directly applicable to other sensors at lower rain rates. It is more useful to have a physically-based model that is relevant for a variety of realistic sensors and rain-rates.

More recently, ref. [4] published experimental results quantifying the influence of rain on the reflected intensity of a Velodyne VLP-16 sensor. Even though the empirical results are again not generally applicable to all rain rates and sensors, the results are useful in constraining the analytical model developed in this work.

Perhaps the most detailed work on the influence of rain on LIDAR sensors is [5], which gave quantitative predictions for the LIDAR range reduction as a function of rain rate and compared these to laboratory measurements. However, this model requires a detailed measurement of LIDAR

specifications and in fact used a LIDAR sensor which is not currently commercially available. Because it is not often possible to easily acquire detailed internal specifications of a LIDAR sensor, a model is developed in this work that uses a simple parametrization of the LIDAR sensor to make predictions of the reflected intensity and range reduction caused by rain. The model is integrated into a physics-based simulator for autonomous driving and several simulated experiments are performed. An ADAS algorithm for obstacle detection is used to evaluate the performance reduction caused by rain in a realistic test environment.

The following sections will discuss the materials and methods used for the experiments, including a detailed description of the software (Section 2), followed by a presentation of the results of several simulated experiments (Section 3). Finally, the consequences of the results will be discussed (Section 4) followed by a brief conclusion to the paper (Section 5).

**2. Materials and Methods**

In the context of ADAS, light detection and ranging (LIDAR) refers to a broad category of sensors that use reflected light to measure the geometry of the environment near the vehicle. While operating principles can range from structured light [6] to amplitude modulation [7], the most commonly used type of LIDAR sensors in automotive ADAS are time-of-flight (TOF) systems [8], which calculate distance by accurately measuring the time it takes a reflected signal to return to the sensor. These TOF LIDAR sensors have greater range than other types of LIDAR, which is of critical importance in automotive ADAS. Therefore, in this work the influence of rain on TOF sensors will be modeled.

Operation of LIDAR in rain may have two consequences. First, the intensity of the signal reflected from the target may be reduced due to scattering from rain droplets. Secondly, back-scatter from the rain may result in a false positive detection from a rain droplet. Regarding false-positives, several studies have found that given geometrical considerations [9] and scattering properties of rain [10], false positives are very unlikely for modern automotive LIDAR sensors that are commonly used on ADAS systems. Therefore, in this work a model of the intensity reduction, and corresponding range reduction, caused by rain is developed.

*2.1. Lidar Theory*

The LIDAR equation for a target at a distance  $z$  from the sensor is [11]

$$P_r(z) = E_l \frac{c\rho(z)A_r}{2R^2} \tau_T \tau_R \exp\left(-2 \int_0^z \alpha(z') dz'\right) \tag{1}$$

where  $P_r$  [W] is the power received by the LIDAR sensor,  $E_l$  [J] is the laser pulse energy,  $c$  [m/s] is the speed of light,  $\rho(z)$  is the back-scattering coefficient of the target,  $\alpha(z')$  is the scattering coefficient of the rain along the path to the target,  $A_r$  [m<sup>2</sup>] is the effective receiver area, and  $\tau_T$  and  $\tau_R$  are the transmitter and receiver efficiencies, respectively. This equation can be simplified by considering the rainy atmosphere to be a homogenous uniformly scattering medium such that the integral in the exponent is reduced to a constant. Additionally, neglecting the spatial variation of the hard target and letting the sensor parameters be reduced to a single coefficient  $C_s = cE_l A_r \tau_T \tau_R / 2$ , the simplified LIDAR equation is then

$$P_r(z) = \frac{C_s \rho}{z^2} e^{-2\alpha z} \tag{2}$$

Finally, because  $C_s$  is a constant for a particular sensor, the relative sensor power  $P_n = P_r / C_s$  is given by

$$P_n(z) = \frac{\rho}{z^2} e^{-2\alpha z} \tag{3}$$

Most LIDAR specification sheets list the maximum range,  $z_{max}$ , of the LIDAR sensor in clear conditions ( $\alpha = 0.0$ ) for a 90% diffusely reflecting surface ( $\rho = 0.9/\pi$ ). In this case, the minimum detectable relative power is estimated as

$$P_n^{min} = \frac{0.9}{\pi z_{max}^2} \tag{4}$$

Equations (3) and (4) can be used to predict the intensity and range reduction for a TOF automotive LIDAR sensor with only two parameters, the rain scattering coefficient,  $\alpha$ , and the maximum range of the LIDAR sensor for a 90% reflective target in clear conditions,  $z_{max}$ . The  $z_{max}$  parameter is typically listed in LIDAR sensor specification sheets, but the rain scattering coefficient is not as easily measurable. Therefore, the relationship between rainfall rate and the scattering coefficient is preferable. Lewandowski et al. derive this relationship for optical and near-infrared (NIR) wavelengths (most automotive LIDAR operate at NIR) and find it should follow a power law [12].

$$\alpha = aR^b \tag{5}$$

where  $R$  is the rainfall rate in mm/h,  $a$  is the extinction coefficient, and  $a$  and  $b$  are empirical coefficients. While [12] find values for  $a$  and  $b$  by fitting measurements from an aerial LIDAR sensor, for this work it is more appropriate to use the measurements from terrestrial automotive LIDAR in the work of [4] to estimate the values of  $a$  and  $b$ . In particular, Filgueira et al. quantified the influence of rain on a VLP-16 sensor by measuring the reduction in reflected intensity as a function of rain rate. The fractional reduction,  $\delta$ , can be defined in terms of the relative intensity of the reflection as

$$\delta = (P - P_0)/P_0 \tag{6}$$

where  $P_0$  is the reflected intensity in the absence of rain. Substituting the model from Equations (3) and (5), the fractional reduction for a given surface at a distance  $z$  can then be modeled as

$$\delta = e^{-2aR^bz} - 1 \tag{7}$$

Comparing this model to the intensity reduction values provided in [4], values of  $a = 0.01$  and  $b = 0.6$  are found to give the best fit to the reported data. Therefore, the final model for the relative intensity returned by the LIDAR as a function of rainfall rate is.

$$P_n(z) = \frac{\rho}{z^2} e^{-0.02R^{0.6}z} \tag{8}$$

The work of [4] also showed that in addition to reducing the intensity of the LIDAR return, the presence of rain also introduces noise to the range measurement. Their measurements indicate that there is not a strong dependence on the rain rate, and that almost all the errors are less than 2%. Therefore, range errors are modeled by sampling from a normal distribution ( $\mathcal{N}$ ) around the true range with a standard deviation of  $\sigma = 0.02z(1 - e^{-R})^2$  to determine the modified range,  $z'$ .

$$z' = z + \mathcal{N}(0, 0.02z(1 - e^{-R})^2) \tag{9}$$

This noise equation has the property that the noise introduced by rain is zero when the rain rate is zero, and the variance increases to a maximum of 2% of the measured range as rain rate increases.

## 2.2. Integration into a 3D Simulator

The goal of this work is to provide a predictive tool for evaluating the influence of rain on LIDAR sensors in ADAS. In order to achieve this, the model developed in the previous section is integrated into a 3D autonomous vehicle simulator that includes a detailed physics-based LIDAR simulation.

This simulator is known as the Mississippi State University (MSU) Autonomous Vehicle Simulator (MAVS), and previous work has shown that it accurately captures the physics of automotive LIDAR by using ray-tracing in geometrically detailed environments and oversampling the laser beam to capture divergence effects [13].

The rain attenuation model was integrated into MAVS using the following procedure (Figure 1):

1. The MAVS LIDAR simulation is used to calculate the returned range and intensity in the absence of rain.
2. Equation (9) is used to calculate the new range rain-induced with error.
3. Equation (7) is used to calculate the reduced intensity value.
4. If the reduced intensity falls below the threshold value defined by Equation (4), the point is removed from the point cloud.

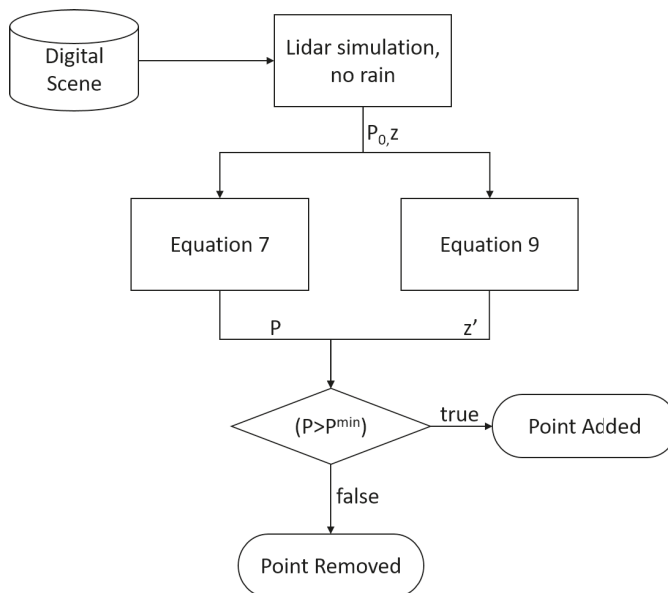


Figure 1. Procedure for modifying LIDAR distances and intensities based on rain rate.

The MAVS was used to evaluate the LIDAR-rain interaction model in both simple test scenarios and in more detailed outdoor environments, both of which are discussed in Section 3.

### 2.3. Maximum Range Experiments

In order to evaluate the validity of the proposed model in a controlled experiment, a simulated test scene was created with large cylinders placed in an arc of increasing radius around the vehicle. This setup provided near-continuous coverage of distance values from 10 m to 80 m, increasing in the clockwise direction around the sensor. In the range experiment, a single scan of a Velodyne VLP-16 [14] sensor was simulated, and the maximum returned range was recorded. The rain rate was increased in the simulation and the experiment was repeated, developing a correlation between rain rate and maximum range for this sensor. The results are presented in Section 3.

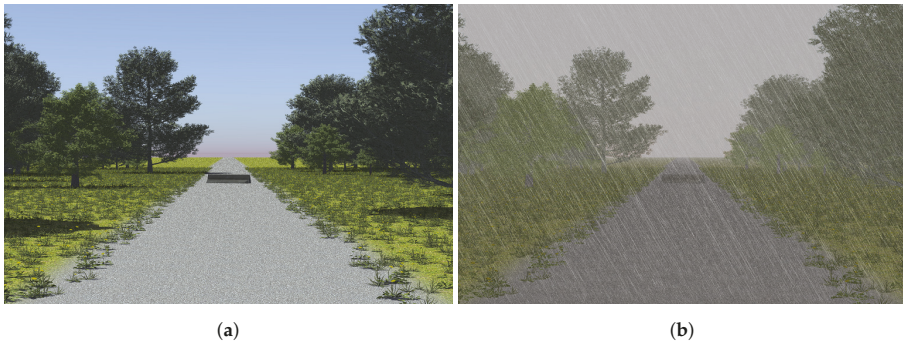
### 2.4. Obstacle Detection in a Realistic Scenario

Simulating the maximum returned range of a LIDAR sensor in a rainy environment provides valuable information, but this information alone is not sufficient to predict the influence of rain on



ADAS. Many environmental factors such as the reflectance properties of surfaces in the scene and the geometric complexity of the scene also influence the performance of ADAS, and all of these factors must be considered simultaneously in order to estimate how rain will affect the performance of ADAS for a given environment and safety function.

A common safety function of ADAS is to provide obstacle detection and avoidance (ODOA), and LIDAR sensors have proved to be a commonly used sensor in ODOA algorithms [15]. In order to evaluate the influence of rain on LIDAR-based ODOA, a simple scenario was created in which an obstacle was placed on a paved rural road. The obstacle was a concrete barrier about 1 m tall and 2 m wide. The scenario is depicted in Figure 2. Note that the rendering method of the rain in MAVS is similar to the method described in [16].



**Figure 2.** (a) The ODOA test scenario in clear weather, rendered by MAVS (b) When raining ( $R = 17$  mm/h).

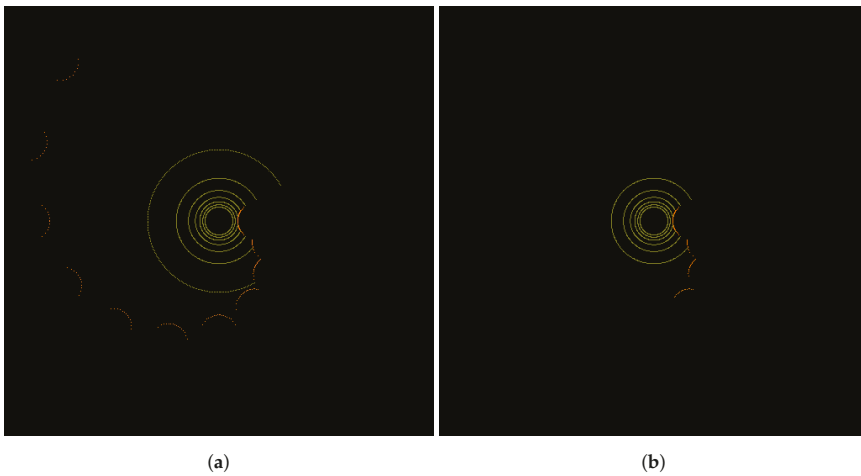
In this scenario, a vehicle with a top-mounted Velodyne HDL-64E [17] sensor drove toward the obstacle, and a segmentation algorithm was used to calculate the location of all detected obstacles in each scan. The segmentation algorithm was the Euclidean cluster extraction algorithm [18] provided with the Point Cloud Library [19]. In the algorithm, the leaf size was set to 0.1 m, the cluster tolerance was set to 1.0 m, and the minimum cluster size was set to two points. While this parametrization tended to create a high number of small clusters, it also gave the highest likelihood of detecting the obstacle at longer ranges.

The farthest distance in which the obstacle could be detected was recorded, and the experiment was repeated with increasing rain rates. The results of this experiment are shown in Section 3.

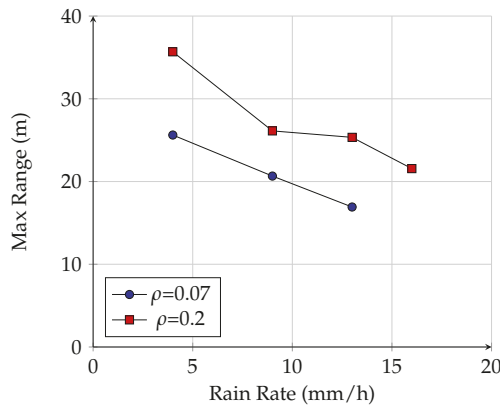
### 3. Results

#### 3.1. Maximum Range Experiments

The results of the maximum range experiments are shown in Figures 3 and 4. Two different target reflectance values were measured for several different rain rates in a manner similar to the experiment presented in [5]. Although the Velodyne VLP-16 sensor used in the simulations does not exactly match the specifications of the sensor used in [5], comparison of Figure 4 to Figure 9 from [5] shows very good qualitative agreement, demonstrating that the model derived in the previous section accurately captures the salient aspects of LIDAR performance in rain as a function of rain rate. In particular, the power-law reduction in reflected intensity (and corresponding reduction in range) is accurately predicted by the model, as well as the overall magnitude of the measured range.



**Figure 3.** (a) Top down view of the LIDAR point cloud in the range test in clear conditions (b) When raining ( $R = 17$  mm/h).



**Figure 4.** Decrease in max range for a simulated LIDAR as a function of rain rate.

### 3.2. Obstacle Detection in a Realistic Scenario

The results of the ODOA scenario are shown in Figures 5–7. Figure 5 shows the expected result that the number of points returned for a scan decreases as the rain rate increases, and Figure 7 gives a visualization of how the decreasing range of the LIDAR with increased rain rate results in an overall reduction in the number of points in the scan. Clearly, the resulting point cloud is drastically affected by the increasing rain rate. However, as Figure 6 shows, the LIDAR range reduction does not have as strong of an impact on the detection range for the obstacle.

The capability of the sensor to detect the obstacle is clearly dependent on factors other than the rain. This includes the resolution of the sensor in the horizontal and vertical directions, the ability of the Euclidean cluster extraction algorithm to distinguish the obstacle returns from the ground, and the reflectance properties of the obstacle. It is only when rain rates become quite heavy (21 mm/h) that the range reduction caused by the rain becomes an important limiting factor in the ADAS algorithm.

Figure 6 shows that the ADAS algorithm is only affected after the rain rate reaches 17 mm/h, which is a rather heavy rain that would not frequently be encountered in most environments. Additionally, the variation with rain rate is small, with only a 6-m reduction at a rain rate of 45 mm/h. To put this

range reduction in more operational terms, a typical passenger vehicle operating on wet pavement has braking distance,  $d_b$ , that varies as the square root of the vehicle speed,  $v_s$  [20].

$$d_b = 3\sqrt{v_s} \tag{10}$$

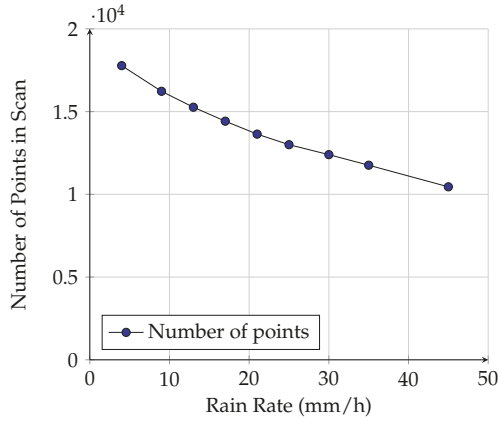


Figure 5. Number of points in a single scan of the LIDAR as a function of rain rate.

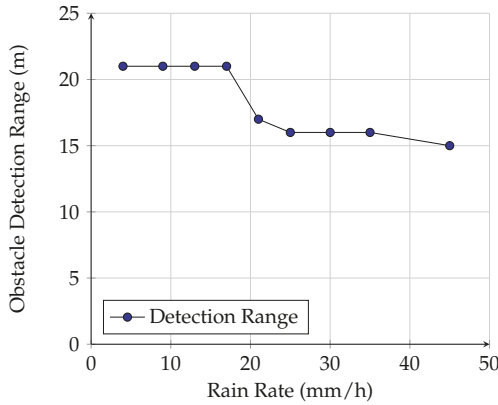


Figure 6. Obstacle detection range as a function of rain rate.

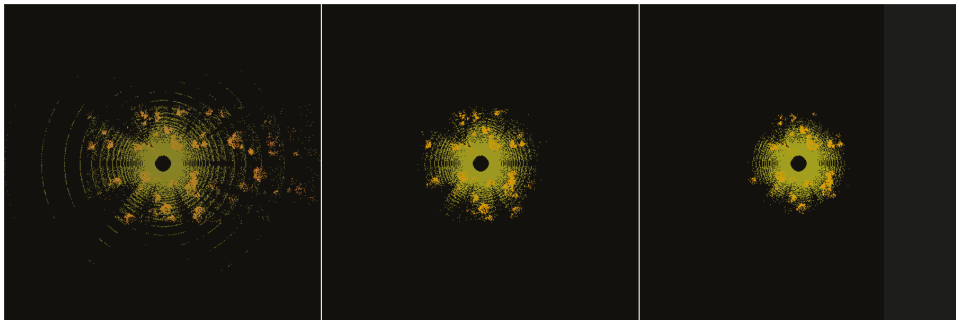


Figure 7. LIDAR range decrease with increasing rain rate from 0 mm/h on the far left, 9 mm/h in the middle, and 17 mm/h on the right.

This equation indicates that the maximum safe operating speed on wet pavement is about 14 m/s in clear conditions and about 12 m/s in the heaviest rain (45 mm/h). Therefore, for this simple ADAS algorithm using roof mounted LIDAR, heavy rain does not prove to be a particularly important factor in the system performance.

#### 4. Discussion

The previous sections focused on obstacle detection with LIDAR. However, most ADAS systems use a combination of complimentary sensors including cameras, stereo vision, and automotive RADAR to detect obstacles in the vehicles environment [21]. In particular, many vision systems use convolutional neural networks to detect and classify obstacles and other features of the environment [22]. Nevertheless, because ODOA requires accurate estimation of the obstacle position, some studies have concluded that LIDAR is of primary importance in ODOA algorithms [21,23].

Because the rain model presented in previous sections is applicable to cameras, LIDAR, and RADAR, future studies examining the influence of rain on multi-sensor ADAS algorithms are possible. However, it is also highly desirable to perform single-sensor analyses like the one presented in this work in order to understand the relative error contributions of the different sensor modalities.

#### 5. Conclusions

There are several important conclusions that may be drawn from this work. First, quantitative predictions of LIDAR performance in rain, and the impact of the sensor performance on ADAS, can be made using a simplified version of the LIDAR equation derived in this work. This model uses only the LIDAR maximum range and rain rate as parameters, allowing the user to determine the quantitative relationship between rain rate and ADAS algorithm performance for a given sensor and environment. Most importantly, using simulation allows the rain-rate to be easily controlled in order to develop a quantitative relationship between rain-rate and ADAS performance.

Second, although the degradation in LIDAR performance caused by rain is well known, the actual impact to ADAS algorithm performance may not be clear cut, and integrated, closed-loop simulations like the one presented in this work are necessary to determine how rain may or may not limit the ADAS.

Finally, even for multi-sensor systems, analyzing the performance of each sensor in a rainy environment, as well as the performance of the combined sensor package, is necessary to fully understand the influence of rain on the ADAS performance.

**Author Contributions:** C.G. provided conceptualization, software, and writing; M.D., C.H. and D.C. provided conceptualization, reviewing, and editing.

**Funding:** Funding for this research was provided by the Center for Advanced Vehicular Systems, Mississippi State University. This research received no external funding.

**Conflicts of Interest:** The authors declare no conflict of interest.

#### References

1. Stock, K. Self-Driving Cars Can Handle Neither Rain nor Sleet nor Snow. *Bloomberg Businessweek*, 17 September 2018.
2. Rashedi, R.; Gresser, K. Automotive radar and lidar systems for next generation driver assistance functions. *Adv. Radio Sci.* **2005**, *3*, 205–209. [[CrossRef](#)]
3. Hasirlioglu, S.; Doric, I.; Lauerer, C.; Brandmeier, T. Modeling and simulation of rain for the test of automotive sensor systems. In Proceedings of the 2016 IEEE Intelligent Vehicles Symposium (IV), Gothenburg, Sweden, 19–22 June 2016; pp. 286–291.
4. Filgueira, A.; González-Jorge, H.; Lagüela, S.; Díaz-Vilariño, L.; Arias, P. Quantifying the influence of rain in LiDAR performance. *Measurement* **2017**, *95*, 143–148. [[CrossRef](#)]

5. Rasshofer, R.H.; Spies, M.; Spies, H. Influences of weather phenomena on automotive laser radar systems. *Adv. Radio Sci.* **2011**, *9*, 49–60. [[CrossRef](#)]
6. Fofi, D.; Sliwa, T.; Voisin, Y. A comparative survey on invisible structured light. In *Proceedings of the Machine Vision Applications in Industrial Inspection XII*; International Society for Optics and Photonics: Bellingham, WA, USA, 2004; Volume 5303, pp. 90–99.
7. Okubo, Y.; Ye, C.; Borenstein, J. Characterization of the Hokuyo URG-04LX laser rangefinder for mobile robot obstacle negotiation. In *Proceedings of the Unmanned Systems Technology XI*; International Society for Optics and Photonics: Bellingham, WA, USA, 2009; Volume 7332, p. 733212.
8. SICK AG. *LMS200/211/221/291 Laser Measurement Systems, Technical Description*; SICK AG: Reute, Germany, 2006.
9. Fersch, T.; Buhmann, A.; Koelpin, A.; Weigel, R. The influence of rain on small aperture LiDAR sensors. In *Proceedings of the 2016 German Microwave Conference (GeMiC)*, Bochum, Germany, 14–16 March 2016; pp. 84–87.
10. Wang, B.; Lin, J.X. Monte Carlo simulation of laser beam scattering by water droplets. In *Proceedings of the International Symposium on Photoelectronic Detection and Imaging 2013: Laser Sensing and Imaging and Applications*; International Society for Optics and Photonics: Bellingham, WA, USA, 2013; Volume 8905, p. 89052.
11. Dannheim, C.; Icking, C.; Mäder, M.; Sallis, P. Weather Detection in Vehicles by Means of Camera and LIDAR Systems. In *Proceedings of the 2014 Sixth International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN)*, Tetova, Macedonia, 27–29 May 2014; pp. 186–191.
12. Lewandowski, P.A.; Eichinger, W.E.; Kruger, A.; Krajewski, W.F. Lidar-based estimation of small-scale rainfall: Empirical evidence. *J. Atmos. Ocean. Technol.* **2009**, *26*, 656–664. [[CrossRef](#)]
13. Goodin, C.; Doude, M.; Hudson, C.; Carruth, D. Enabling Off-Road Autonomous Navigation-Simulation of LIDAR in Dense Vegetation. *Electronics* **2018**, *7*, 154. [[CrossRef](#)]
14. Velodyne Acoustics, Inc. *VLP-16 User's Manual and Programming Guide*; Velodyne Acoustics, Inc.: Morgan Hill, CA, USA, 2016.
15. Schafer, H.; Hach, A.; Proetzsch, M.; Berns, K. 3D obstacle detection and avoidance in vegetated off-road terrain. In *Proceedings of the 2008 IEEE International Conference on Robotics and Automation (ICRA 2008)*, Pasadena, CA, USA, 19–23 May 2008; pp. 923–928.
16. Starik, S.; Werman, M. Simulation of rain in videos. In *Proceedings of the 2003 Texture Workshop*, Nice, France, 17 October 2003; Volume 2, pp. 406–409.
17. Velodyne Acoustics, Inc. *HDL-64E User's Manual, Rev. D*; Velodyne Acoustics, Inc.: Morgan Hill, CA, USA, 2008.
18. Rusu, R.B. Semantic 3D object maps for everyday manipulation in human living environments. *Künstliche Intelligenz* **2010**, *24*, 345–348. [[CrossRef](#)]
19. Rusu, R.B.; Cousins, S. 3D is here: Point cloud library (PCL). In *Proceedings of the 2011 IEEE International Conference on Robotics and automation (ICRA)*, Shanghai, China, 9–13 May 2011; pp. 1–4.
20. Queensland Government. *Stopping Distances on Wet and Dry Roads*; Queensland Government: Queensland, Australia, 2016.
21. Jiménez, F.; Naranjo, J.E. Improving the obstacle detection and identification algorithms of a laserscanner-based collision avoidance system. *Transport. Res. Part C Emerg. Technol.* **2011**, *19*, 658–672. [[CrossRef](#)]
22. Aarhi, R.; Harini, S. A Survey of Deep Convolutional Neural Network Applications in Image Processing. *Int. J. Pure Appl. Math.* **2018**, *118*, 185–190.
23. Ferguson, D.; Darms, M.; Urmson, C.; Kolski, S. Detection, prediction, and avoidance of dynamic obstacles in urban environments. In *Proceedings of the 2008 IEEE Intelligent Vehicles Symposium*, Eindhoven, The Netherlands, 4–6 June 2008; pp. 1149–1154.



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# Enabling Off-Road Autonomous Navigation-Simulation of LIDAR in Dense Vegetation

Christopher Goodin \*, Matthew Doude, Christopher R. Hudson and Daniel W. Carruth

Center for Advanced Vehicular Systems, Mississippi State University, Starkville, MS 39759, USA; mdoude@CAVS.MsState.Edu (M.D.); chudson@cavs.msstate.edu (C.R.H.); dwc2@CAVS.MsState.Edu (D.W.C.)

\* Correspondence: cgoodin@cavs.msstate.edu

Received: 18 July 2018; Accepted: 16 August 2018; Published: 21 August 2018

**Abstract:** Machine learning techniques have accelerated the development of autonomous navigation algorithms in recent years, especially algorithms for on-road autonomous navigation. However, off-road navigation in unstructured environments continues to challenge autonomous ground vehicles. Many off-road navigation systems rely on LIDAR to sense and classify the environment, but LIDAR sensors often fail to distinguish navigable vegetation from non-navigable solid obstacles. While other areas of autonomy have benefited from the use of simulation, there has not been a real-time LIDAR simulator that accounted for LIDAR–vegetation interaction. In this work, we outline the development of a real-time, physics-based LIDAR simulator for densely vegetated environments that can be used in the development of LIDAR processing algorithms for off-road autonomous navigation. We present a multi-step qualitative validation of the simulator, which includes the development of an improved statistical model for the range distribution of LIDAR returns in grass. As a demonstration of the simulator’s capability, we show an example of the simulator being used to evaluate autonomous navigation through vegetation. The results demonstrate the potential for using the simulation in the development and testing of algorithms for autonomous off-road navigation.

**Keywords:** perception in challenging conditions; obstacle detection and classification; dynamic path-planning algorithms

---

## 1. Introduction

Laser ranging sensors, commonly referred to as LIDAR, are ubiquitous in off-road autonomous navigation because they provide a direct measurement of the geometry of the operating environment of the robot [1]. One of the ongoing issues with LIDAR perception is the inability of the sensor to distinguish between navigable obstacles like grass and non-navigable solid obstacles. This problem is stated clearly by [2]:

Among the more pervasive and demanding requirements for operations in vegetation is the discrimination of terrain from vegetation-of rocks from bushes... Failure to make the distinction leads to frustrating behaviors including unnecessary detours (of a timid system) around benign vegetation or collisions (of an aggressive system) with rocks misclassified as vegetation.

While there has been progress over the last decade in addressing the perception issues associated with LIDAR [3,4], the mitigating techniques have primarily been developed and refined experimentally. Recent advances in simulation for robotics have demonstrated that autonomy algorithms can be developed and tested in simulation [5–7]. However, up until now simulations have either lacked

the fidelity to realistically capture LIDAR-vegetation interaction or been computationally slow and difficult to integrate with existing autonomy algorithms [8].

In this work, the development, validation, and demonstration of a realistic LIDAR simulator that can be used to develop and test LIDAR processing algorithms for autonomous ground vehicles is presented. It accurately captures the interaction between LIDAR and vegetation while still maintaining real-time performance. Real-time performance is maintained by making some simplifying approximations and by using Embree, an open-source ray-tracing engine from Intel [9], for ray-tracing calculations, making the simulator useful for integration into “in-the-loop” simulations of autonomous systems.

### *LIDAR Basics*

While there are several different types of LIDAR sensors, this paper focuses on incoherent micro-pulse LIDAR sensors, commonly referred to as Time-of-Flight (TOF) sensors, as this type of LIDAR is by far the most common type used in outdoor autonomous navigation. The operating principle of TOF LIDAR is to measure the time between the initial pulse of the LIDAR and the arrival of the reflected pulse, and divide this time by the speed of light to derive a distance [10].

Most LIDAR sensors used in autonomous navigation also feature a rotating scanning mechanism that allows the laser to sample multiple locations. For example, the SICK sensor scans by having a stationary laser reflect from a rotating mirror, producing a planar “slice” of the environment, while the Velodyne sensors feature an array of 16, 32, or 64 laser-sensor pairs in a rotating housing, producing a 3D point cloud with each full rotation of the sensor.

There are several different sources of error in LIDAR range measurements. The finite duration of the laser pulse, discretization of the reflected signal for processing, and atmospheric scattering properties all contribute to the error. However, this work focuses on the divergence of the laser beam and how the diverging beam’s interaction with the fine scale detail of vegetation results in error in the LIDAR range measurement.

In the following sections, the paper briefly outlines the materials and methods of the simulated experiments (Section 2), describes the results of simulations (Section 3) and presents discussion and conclusions based on these results (Sections 4 and 5).

## **2. Materials and Methods**

This work describes the development of a high-fidelity, physics-based LIDAR simulator for off-road autonomous vehicles. The method of validation of the simulator is qualitative comparison to previously published experiments on LIDAR interaction with vegetation and other extended objects.

The LIDAR simulation has several key objectives that guided development decisions. First, it must simulate common robotic LIDAR sensors such as the Velodyne HDL-64E operating in highly-vegetated outdoor environments in real-time. Second, it must realistically capture the salient characteristics of laser-beam interaction with blades of grass and leaves. Third, the simulation must be generic enough to simulate a variety of different LIDAR sensors with only the parameters obtained from specification sheets and other freely available data. With these requirements in mind, it is noted that the LIDAR simulator presented here was not developed to support the design and development of LIDAR sensors. Rather, the simulator was developed to be used in robotics applications where real-time LIDAR simulation is a requirement.

### *2.1. Software*

The simulator is written in C++ with MPI bindings. The compiler used was the Intel C++ compiler. Third party libraries used in the development of the simulator are given in Table 1.

**Table 1.** Third-party software libraries used by the simulator.

Software	Available at
Cimg	<a href="http://cimg.eu">cimg.eu</a>
TinyObjLoader	<a href="https://github.com/syoyo/tinyobjloader">https://github.com/syoyo/tinyobjloader</a>
OpenGL Mathematics (GLM)	<a href="https://github.com/g-truc/glm">https://github.com/g-truc/glm</a>
Rapidjson	<a href="https://github.com/Tencent/rapidjson">https://github.com/Tencent/rapidjson</a>
HosekWilkie Sky Model	<a href="http://cgg.mff.cuni.cz/projects/SkylightModelling/">http://cgg.mff.cuni.cz/projects/SkylightModelling/</a>
Rinex	<a href="https://www.ngs.noaa.gov/gps-toolbox/rinex.htm">https://www.ngs.noaa.gov/gps-toolbox/rinex.htm</a>
a-star	<a href="https://github.com/hjweide/a-star">https://github.com/hjweide/a-star</a>
Embree	<a href="https://github.com/embree">https://github.com/embree</a>

## 2.2. Hardware

Two different computers were used for the simulations in this work. The LMS-291 simulations were run on a Linux workstation using four Intel Xeon E5 2.9 GHz CPUs. The Velodyne simulations were run with 40 Intel Ivy Bridge processors on Mississippi State University's High Performance Computer, Shadow [11].

## 3. Results

This section describes the results of simulated LIDAR experiments and their comparison to previously published data. This section also outlines the development of an analytical range model for LIDAR penetration into grass for comparison to simulated results.

### 3.1. Simulation Parameters

The parameters used by the LIDAR model are listed in Table 2, all of which are typically found in sensor specification sheets. As an illustrative example, the parameters used in the simulation for a common LIDAR sensor, the Velodyne HDL-32E [12], are also listed.

**Table 2.** LIDAR simulation parameters.

Parameter	Units	HDL-32E
Min & max horizontal angle	degrees	[−180, 180]
Horizontal resolution	degrees	0.16
Min & max vertical angle	degrees	[−30.6623, 10.67]
Vertical resolution	degrees	1.3333
Min range	m	1.0
Max range	m	70
Beam spot shape	circular, rectangular, or elliptical	rectangular
Horizontal divergence	rad	0.0033
Vertical divergence	rad	0.0007
Signal cutoff	m	1.0
Mode	first, last, strongest, strongest & last	strongest or last

### 3.2. Physics of Laser-Vegetation Interaction

There are three processes which need to be simulated in order to realistically capture the salient characteristics of the laser-beam interaction with vegetation. First, the divergence and shape of the laser beam should be taken into account. Second, the scattering properties of the leaves and vegetation in the environment must be simulated. Finally, the on-board processing of the LIDAR sensor also influences the result.



### 3.2.1. Beam Divergence

In order to simulate beam divergence, the simulator uses the Embree ray-tracing kernel [9,13] to over-sample each laser pulse and estimate a return signal. The signal pulse is then processed according to the specified mode of the LIDAR, which can be first, strongest, or last return, or both strongest and last. Ideally, each beam would be randomly sampled to build up a representative reflected pulse. However, in order to maintain the real-time requirement, the simulator uses a fixed 9-point stencil for each of the three beam spot shapes. The three stencils are shown in Figure 1a–c.

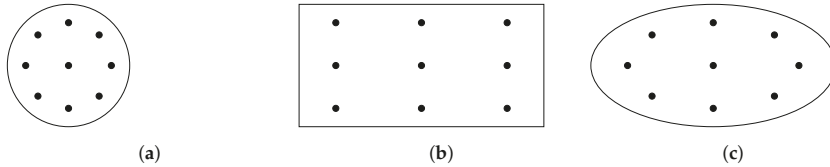


Figure 1. Stencils used to sample the beam spots for (a) circular, (b) rectangular, and (c) elliptical beams.

In the simulation, nine rays are traced from the sensor origin through the stencil points for each beam pulse. The location of the stencil points is defined by the divergence values specified for the sensor at 1 m of range. For example, for a sensor located at the origin, oriented with the z-axis up, having a circular beam spot with divergence  $\gamma$  and beam oriented along the x-axis, the stencil points would lie in a circle in the y – z plane with radius given by

$$\text{radius} = \sqrt{(2) \tan(\gamma/2)/4}$$

and centered around the x-axis at  $x = 1$ .

### 3.2.2. Scattering from Leaves and Vegetation

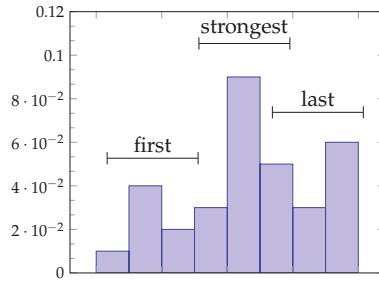
The environment is modeled as a collection of triangular meshes, with each triangle attributed with a reflectance value. All materials in the scene are assumed to be diffuse reflectors, so if a ray intersects a triangle, the intensity of the retro-reflected ray is given by

$$I = \rho I_0 \cos(\theta) \tag{1}$$

where  $\rho$  is the surface reflectance (with possible values ranging from 0 to 1),  $\theta$  is the angle between the surface normal and the ray, and  $I_0$  is the intensity of the laser. In the simulator,  $I_0 = 1$  for all sensors and perform all calculations in relative intensity. Multiply scattered rays are not considered in the simulation. Although multiply-scattered rays can introduce anomalies in environments with highly reflective materials, natural materials tend to reflect diffusely with values of  $\rho < 0.5$ , in which case the intensity of multiply scattered rays typically falls below the detection threshold of the receiver.

### 3.2.3. Signal Processing

For each pulse, all reflected rays are stored in a reflected signal. The range is then extracted from the signal according to the mode of the LIDAR. If the mode is “strongest”, the ray with the most intense reflection is used to calculate the distance. If the mode is “last”, the ray with the furthest distance (and longest time) is returned. It is also possible to return both the strongest and last signals. Finally, if the mode is set to “first”, all signals between the closest reflection (in time and distance) and those within the “signal cutoff” distance of the closest return are averaged. The signal cutoff parameter accounts for the fact that LIDAR sensors typically do not average over the time window of the entire pulse. The value of the signal cutoff parameter can be inferred from laboratory measurements [14]. Figure 2 illustrates how the signal cutoff parameter is used to process the signals in the simulation.

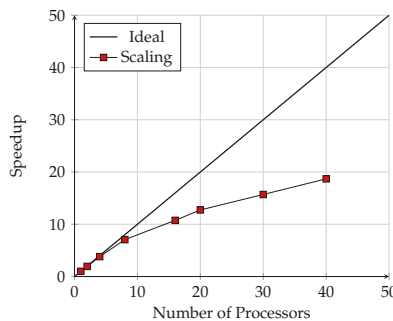


**Figure 2.** Example of pulse processing. Each laser beam in the simulation is sampled by 9 rays. An averaging window (depicted by the horizontal bars) is defined by the “Signal cutoff” parameter, and the location of the window depends on the mode.

### 3.3. Real-Time Implementation

In the example simulations shown in the following sections, the simulated environment contains over 193 million triangles. A sensor like the Velodyne HDL-64E returns about  $10^6$  points per second, and the simulator traces 9 rays per pulse. This means that the simulation must perform  $\mathcal{O}(10^{15})$  ray-triangle intersections per second of simulation time. Even with a ray-tracing kernel like Embree that uses bounding-volume hierarchy acceleration, ray-tracing simulations on this scale require additional parallelization in order to maintain real-time speed.

The Message Passing Interface (MPI) [15] is used to parallelize each scan by laser pulse so that each bundle of 9 rays can be simulated independently. This method is embarrassingly parallel and scales well with the addition of processors. Figure 3 shows the scaling of the code with the number of processors used.



**Figure 3.** Scaling of the LIDAR simulation of a Velodyne HDL-64E in a scene with 191,245,324 triangles.

The Sick LMS-291 and the Velodyne HDL-64E were used for computational performance bench-marking. The LMS-291 simulation was run on 4 processors at approximately 30% faster than real time (6.8 s of simulated time in 4.7 s of wall time). The HDL-64E was run on 40 processors at approximately 30% faster than real time (6.8 s of simulated time in 4.7 s of wall time). Details about the hardware used to run the simulations are given in Section 2.2. These examples demonstrate that the implementation is capable of achieving real-time performance, even for very complex LIDAR sensors and environment geometries.

### 3.4. Simulation Validation

The requirements for validation of physics-based simulation can vary in rigor depending on the application. Accuracy requirements must be defined by the user, and this creates well-known

difficulties when generating synthetic sensor data for autonomous vehicle simulations [16]. In the sections above, the development of a generalized LIDAR simulation that captures the interaction between LIDAR and vegetation was presented. The simulation is not optimized for a particular model of LIDAR sensor or type of environment. Therefore, in order to validate the simulation, is of primary importance to qualitatively reproducing well-known LIDAR-vegetation interaction phenomenon. To this end, three validation cases are presented. In the first case, the results of the simulation are compared to an analytical model. Next, simulation results are compared to previously published laboratory experiments. Finally, the simulation is compared to previously published controlled field experiments. All three cases show good qualitative agreement of the simulation with the expected results.

### 3.4.1. Comparison to Analytical Model

The statistics of LIDAR range returns in grass have been studied for nearly two decades. In particular, Macedo, Manduchi, and Matthies [3] presented an analytical model for the range distribution of LIDAR returns in grass that was based on the exponential distribution function, and this model has been cited frequently in subsequent works [4,14,17,18]. Accounting for the beam divergence, Ref. [3] present the following model for the probability distribution of the LIDAR interacting with a stand of grass, modeled as randomly placed, uniform diameter cylinders.

$$p(r) = \lambda d(1 + ar)e^{-\lambda d(r-D)(1+a(r+D)^2/2)}U(r - D) \tag{2}$$

where  $\lambda$  is the number density of the grass per square meter,  $d$  is the average grass stem diameter (meters),  $D$  is the distance between the laser and the edge of the grass-stand (meters),  $r$  is the measured range (meters), and  $a$  is related to the laser beam divergence,  $\gamma$  (radians) by

$$a = 2 \tan(\gamma/2) \tag{3}$$

Although the details of the derivation of Equation (2) are not presented in [3], it appears that the model was developed by assuming that the LIDAR’s returned distance would be the nearest distance encountered along the beam spot profile. However, the exponential model has the unsatisfactory result that the most likely returned distance is  $D$ , which runs counter to simple geometrical considerations. Additionally, as noted in [3], TOF LIDAR sensors will measure an average over the spatial resolution of the beam spot. The exponential model does not properly account for this averaging over the entire width of the returned pulse. Therefore, an improved analytical model is needed.

Noting that the distribution of the average of multiple samples of an exponentially distributed random variable is a gamma distribution, an improved analytical model based on the gamma distribution is proposed. In the case of a laser beam, the integration is continuous over the width of the beam. Noting that the beam spot factor is proportional to  $aD^2 \approx \gamma D^2$  for small values of  $\gamma$  [3], the continuous variable of integration is

$$\alpha = 1 + \gamma D^2. \tag{4}$$

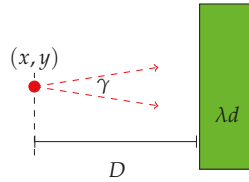
The range probability equation is then given by the gamma distribution

$$p(r) = \frac{(r - D)^{\alpha-1} (\alpha \lambda d)^\alpha}{\Gamma(\alpha)} e^{-\alpha \lambda d(r-D)} \tag{5}$$

where  $\Gamma$  is the gamma function. This choice of  $\alpha$  gives the desirable property that when the divergence is zero, the model reproduces the exponential model while tending towards a normal distribution as  $\alpha$  gets large.

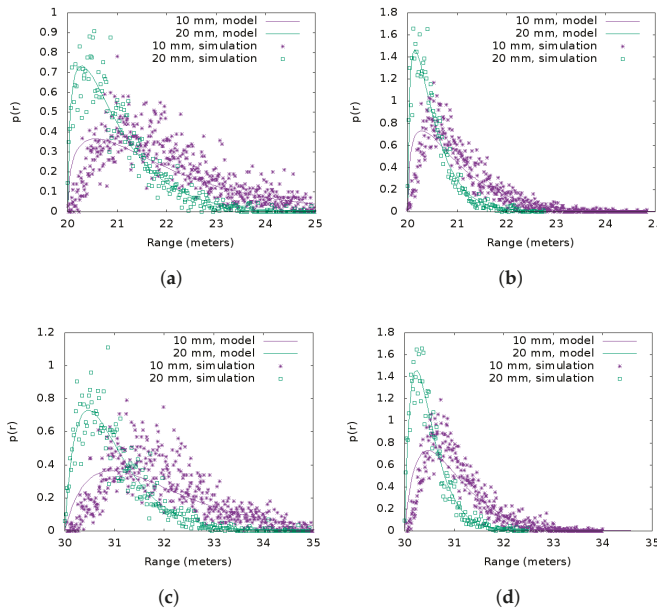
In order to compare the simulation to this model, a single point LIDAR with a divergence of 1 mrad aimed at a stand of randomly placed grass, modeled as uniform vertical cylinders, was simulated.

For purposes of comparison, the signal cutoff parameter was set to a value of 100 m, simply because the analytical model does not account for signal cutoff. The grass stand was 10 m wide and 5 m deep. Stem densities of  $\lambda = 50$  and  $\lambda = 100$  were simulated, corresponding to 2500 and 5000 stems, respectively, in the 50 m<sup>2</sup> area. The simulation setup is shown in Figure 4.



**Figure 4.** Setup of simulated experiment. The red dashed lines represent the diverging laser beam at from position  $(x, y)$ . In the simulated experiments,  $y$  varied randomly and  $x = [20, 30]$ .

Simulation results for distances of 20 and 30 m are shown in Figure 5a–d. Several interesting features of the model and simulation are noted. First, increasing the range,  $D$ , shifted the entire distribution to the right, as predicted by the model. Second, the most likely value in the simulated distribution matches the model fairly well, although better for the larger grass diameters. Lastly, gamma distribution and the simulation match well qualitatively, indicating that for a diverging beam the gamma distribution is indeed a better predictor of the range statistics than the exponential distribution. The qualitative agreement between the simulation and the analytical model strongly indicates that the simulation is valid for LIDAR-grass interaction for a range of distances and grass properties.



**Figure 5.** Comparison of the simulated experiments to the improved analytical model for stem diameters of 10 mm and 20 mm. (a)  $D = 20$ ,  $\lambda = 50$ , (b)  $D = 20$ ,  $\lambda = 100$ , (c)  $D = 30$ ,  $\lambda = 50$ , (d)  $D = 30$ ,  $\lambda = 100$ .

### 3.4.2. Comparison to Laboratory Experiment

There have been a number of attempts to experimentally quantify the occurrence of mixed pixels in LIDAR measurements. One experiment presented laboratory tests for mixed pixel effects using cylindrical rods arranged in front of a flat background [14]. In the experiment, 8–9 cylindrical rods of various diameters were placed vertically in front of a flat plywood background. While the exact locations and diameters of the rods are not given in the reference, it is possible to infer the approximate arrangement from the figures given in the paper. Figure 6 shows the arrangement used for the simulations. Nine cylindrical rods were arranged in even 12.7 cm spacings in a row parallel to the background. All rods had diameter of 25 mm, except for one which had a diameter of 75 mm. The sensor was placed 80 cm away from the row of rods, and the background was moved to an offset of either 60 cm or 2 m, to match the results given in Figure 4 of [14].

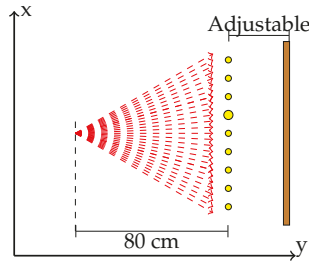
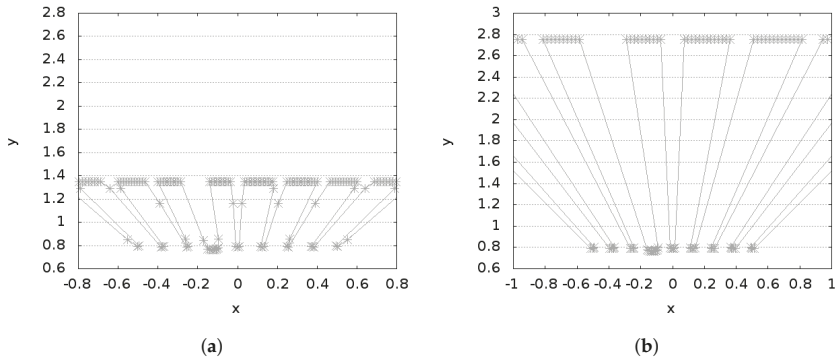


Figure 6. Setup of the mixed pixel experiment from [14].

To compare the results of the laboratory simulations, shown in Figure 7a,b, the LMS-291 with settings listed in Table 3 was used. The LMS-291, although originally designed for manufacturing applications, was quite common in field robotics for nearly a decade. However, due to a relatively high divergence ( $>10$  mrad) the sensor is especially prone to mixed pixels. The primary conclusion of the original experiment was that mixed pixels were present when the offset between the background and the cylinders is less than 1.6 m, but at larger offsets the mixed pixels are not present. This is due to the timing resolution and signal processing of the sensor. Figure 7a,b present the results of our simulation, and show clearly that results from the original laboratory experiments are qualitatively reproduced by our simulation. This indicates that our simulated beam divergence and signal processing correctly reproduces the real sensor.

Table 3. LIDAR simulation parameters for the LMS-291.

Parameter	LMS-291S05
Horizontal angles	[−50,50]
Horizontal resolution	0.5, 1.0
Vertical angles	-
Vertical resolution	-
Min range	-
Max range	80
Beam spot shape	circular
Horizontal & vertical divergence	0.0129
Signal cutoff	1.6
Mode	first return

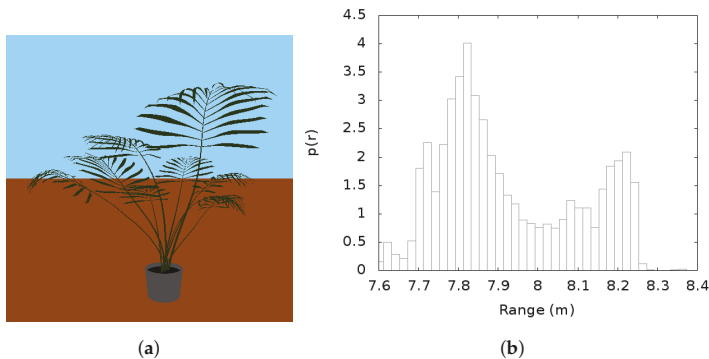


**Figure 7.** Simulation of the experiment from [14]; compare to Figure 4 from [14]. (a) Scan with the background 60 cm behind targets. Note the presence of mixed-pixels. (b) Scan with background 2 m behind targets. Note the absence of mixed-pixels.

### 3.4.3. Comparison to Controlled Field Test

Velodyne LIDAR sensors have become ubiquitous in field robotics in the last decade [1]. In particular, the availability of accurate, spatially dense point clouds provided with the introduction of the Velodyne HDL-64E enabled tremendous advances in LIDAR-based autonomous navigation [19–22]. The Velodyne sensors have a more focused beam than the SICK scanners and are thus less prone to mixed pixels. This lead to a higher degree of “penetration” into extended objects such as vegetation. This feature has been exploited to help distinguish vegetation from solid objects in 3D point clouds generated by the Velodyne sensor [20].

In this section, it is shown that our simulation accurately reproduces sensor-vegetation interaction for the Velodyne HDL-64E. Statistical techniques for quantifying the range penetration properties of LIDAR into vegetation were presented in [23], who showed results for range variability when scanning vegetation with a Velodyne HDL-64E LIDAR. In their experiment, a small potted shrub was placed at a distance of 8 m from the sensor and a histogram of the returned distances from approximately 1500 range measurements was presented. While the experiment cannot be exactly reproduced because the geometric detail about the shrub used in the experiment is not available, the experiment has been reproduced using a shrub model that appears to be a similar size and shape to the one used in [23]. The shrub model used in the experiment is shown in Figure 8a.



**Figure 8.** Simulated comparison to experiment from [23]. (a) Rendering of the shrub that was scanned. (b) Range histogram for multiple scans of the shrub at a distance of 8 m. Compare to Figure 1 from [23].

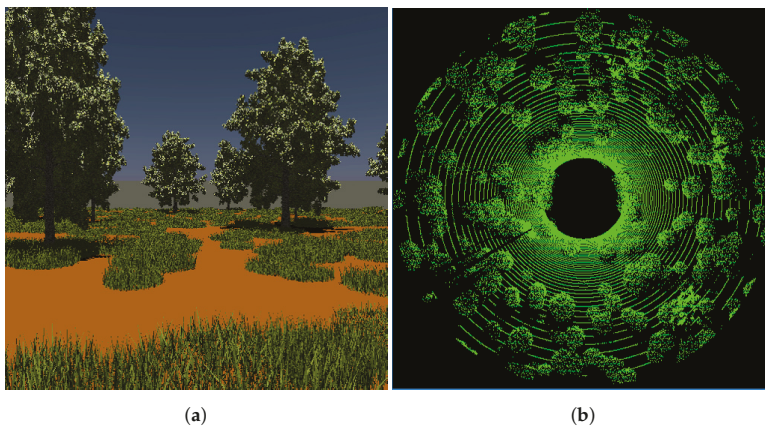
In our simulated experiment, the sensor was placed at a distance of 8 m from the shrub shown in Figure 8a and extracted points from one rotation of the sensor on the 5 Hz setting. The sensor was moved along a quarter-circle arc at a distance of 8 m in 1 degree increments, for a total of 91 measurement locations. All distance measurements which returned from the shrub were binned into a histogram, which is shown in Figure 8b. In the original experiments, two main features of the range distribution were observed [23]. First, the broadening of the distribution due to the extended nature of the object, and second the bi-modal nature of the distribution due to some returns from the trunk and others from the foliage. Comparing our Figure 8b to Figure 4 from [23], it is clear that the simulation reproduces both of these features of the distance distribution. This qualitative agreement provides indication that our simulation is valid for Velodyne sensors interacting with vegetation.

### 3.5. Simulation Demonstration

In this section, the results of an example demonstration in a realistic digital environment are presented. The goal of the demonstration is to show how the simulator could be used to develop and test autonomy algorithms and highlight the LIDAR-vegetation interaction. While the navigation algorithms presented in the demonstration are not novel, the demonstration highlights the efficacy of the sensor simulation for evaluating the navigation algorithm performance and the capability of the simulation to reveal the influence of sensor-environment interaction on autonomous navigation.

#### 3.5.1. Digital Scene

A “courtyard” scene was created by placing vegetation in a  $300 \times 300$  m square, surrounded by a 10 m high wall. One tree model and one grass model were used with random orientations, scales, and locations throughout the courtyard. The resulting scene is shown in Figure 9a. The scene contained nearly 2 million blades of grass, 50 trees, and 191,245,324 triangles. Embree’s instancing feature was used to reduce the memory required to load the scene into the simulation.



**Figure 9.** Digital scene used in the example simulation. (a) Rendering of the digital scene. (b) Point cloud from HDL-64E, top-down view.

#### 3.5.2. Simulating the Velodyne HDL-64E

The Velodyne HDL-64E presents several unique considerations for simulation. First, the sensors consists of two arrays of aligned laser-receiver pairs, with each array having 32 lasers. The lower block of lasers has only one-fourth the horizontal angular resolution of the upper block. The blocks are reported from the sensor in “packets”, and there are three packets from the upper block and then one

from the lower block. Finally, because the repetition rate of the laser array is constant, the horizontal resolution of the sensor depends on the rotation rate of the sensor, which can vary from 5–15 Hz.

In order to reproduce these features in the simulation, the upper and lower blocks of the HDL-64E are simulated as two separate LIDAR sensors and combined in the software into a single result. In the simulation, the variable rotation rate,  $\omega$ , and horizontal resolution ( $\delta^{horizontal}$ ) are related by the equations

$$\delta_{lower}^{horizontal} = \frac{360}{\frac{250000}{32\omega} - 1} \tag{6}$$

$$\delta_{upper}^{horizontal} = \frac{360}{\frac{250000}{8\omega} - 1} \tag{7}$$

In the simulation, the point cloud is assembled from three sets of 32 points from the upper block. The fourth block is discarded from the upper block and replaced by 32 points from the lower block. In this way, the readout and resolution of the real sensor is maintained in the simulation. Table 4 shows how the sensor is parameterized for the simulation. A point cloud from a single simulated scan is shown in Figure 9b.

**Table 4.** LIDAR simulation parameters for the Velodyne HDL-64E.

Parameter	Lower Block	Upper Block
Horizontal angles	[−180,180]	[−180, 180]
Horizontal resolution	Equation (6)	Equation (7)
Vertical angles	[−24.8, −11.6127]	[−11.1873, 2.0]
Vertical resolution	0.41875	0.41875
Min range	1	1
Max range	100	100
Beam spot shape	rectangular	rectangular
Horizontal divergence	0.0033	0.0033
Vertical divergence	0.0007	0.0007
Signal cutoff	1.0	1.0
Mode	strongest	strongest

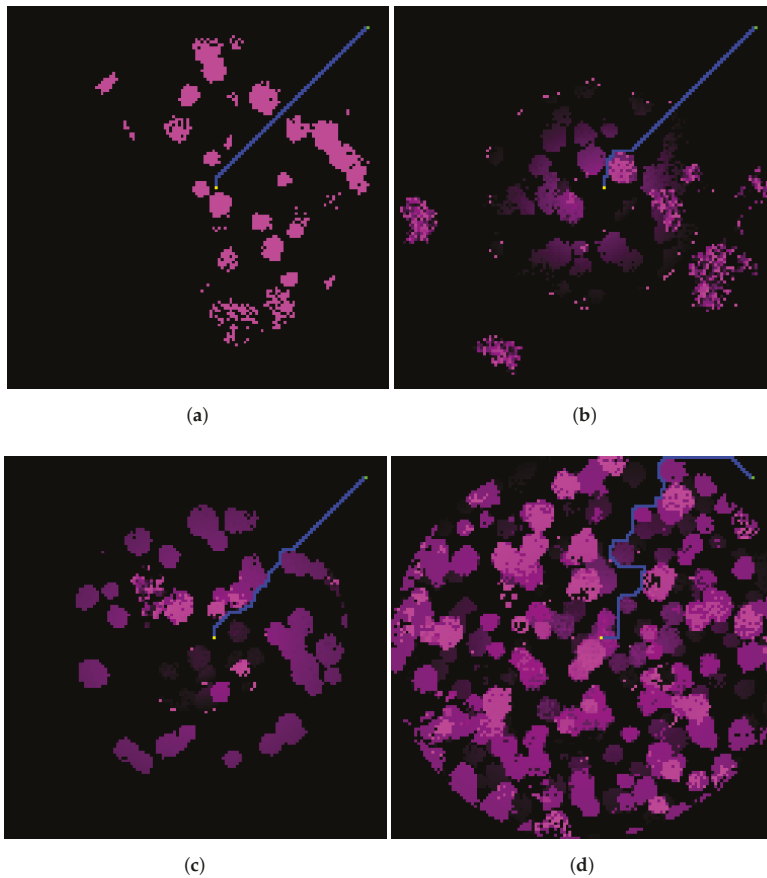
### 3.5.3. Simulation Results

In order to demonstrate how the LIDAR simulation could be used to evaluate autonomous performance, an example simulation featuring autonomous navigation with A\* [24] is presented. The A\* algorithm is a heuristic approach that has been extensively used in autonomous path planning for decades. An implementation based on that by [25] is used in this example. A cost map is created by using scans from the LIDAR sensor. The map had a grid resolution of 0.5 m, and the grid size was 400 × 400. Successive LIDAR scans were registered into world coordinates using a simulated GPS and compass and placed into the grid. Each reflected LIDAR point was assigned a cost,  $c$ , based on the point height,  $h$  using the formula

$$c = \max(100h/h_{max}, 100) \tag{8}$$

where  $h_{max}$  is the maximum negotiable vegetation height of the vehicle. For these simulations,  $h_{max} = 1.5$ . The vehicle was placed in the southwest corner of the scene and given a goal about 130 m northeast of the starting location. The simulated robot was a generic passenger-sized, skid-steered wheeled vehicle with a maximum speed of 2 m/s. The simulation was repeated for four different LIDAR models: the Sick LMS-291 S05, the Velodyne HDL-64E, HDL-32E, and VLP-16. The sensors were mounted on the vehicle at a height of 2 m above the ground. The resulting cost maps after 10 simulated seconds are shown in Figure 10a–d.





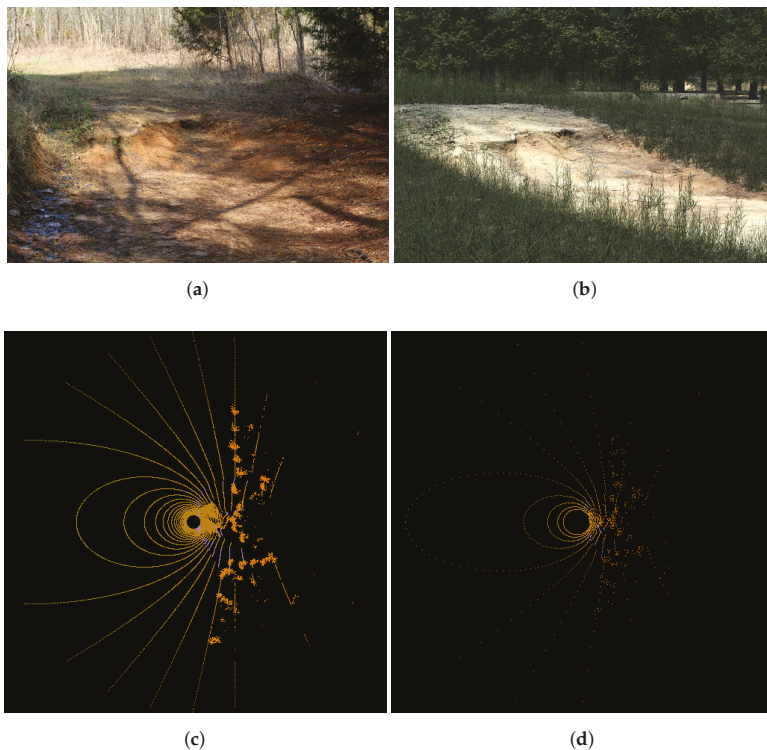
**Figure 10.** Example cost maps calculated using different sensors. Black corresponds to zero cost, while brighter regions are higher cost. (a) Sick LMS-291 (b) Velodyne VLP-16 (c) Velodyne HDL-32E (d) Velodyne HDL-64E.

Visual comparison of the cost maps and the optimal paths calculated by the A\* path planner reveal how the simulator can be used to discover the impact of sensing capability on autonomous performance. The LMS-291, which produces only a horizontal slice of the environment, does not detect most of the grass—it only detects the trees. Additionally, the vegetation that is detected has less range variability within objects—for example compare Figure 10a with Figure 10d. Lack of penetration into vegetation is a well known problem of high-divergence LIDAR like the LMS-291 [3].

There are also clear differences between the cost maps generated by the three Velodyne models. One interesting difference is observed between the VLP-16 and the HDL-64E. While the HDL-64E has a much denser point cloud, the VLP-16 has a much higher maximum opening angle than the HDL-64E ( $15^\circ$  above horizontal for the VLP-16 versus  $2^\circ$  above horizontal for the HDL-64E). This results in the VLP-16 detecting tall obstacles at intermediate ranges somewhat better than the HDL-64E. These show up as the ring-like structure in the VLP-16 cost map (Figure 10b). The HDL-64E, however, is better at detecting smaller nearby obstacles. Again, the relative desirability of these features depends on the application, but this demonstration illustrates how the simulation could be used to study autonomous navigation with realistic LIDAR simulations in densely vegetated environments.

### 3.6. Example of Highly Detailed Digital Environment

In order to demonstrate how the simulator could be used in real world scenarios, a highly detailed digital scene was developed based on a unique off-road terrain feature. The feature, shown in Figure 11a, is a large vertical step in an off-road trail with a tree root acting as a natural embankment. The geometry of the root feature was measured by developing a structure mesh from a sequence of 268 digital images. The resulting surface mesh contained over 22 million triangular faces and was approximately 2 m by 3 m by 0.4 m in height. The scene was augmented with randomly placed models of grass and trees. The resulting scene contained 24,564 grass and tree meshes for a total of 604,488,350 triangular faces. A digital rendering of the synthetic scene is shown in Figure 11b. Additionally, LIDAR simulations of the scene with two of the sensors discussed in this work are shown in Figure 11c,d as a demonstration of the capability of the LIDAR simulation to scan a complex scene in real-time. The points in Figure 11c,d are color-coded by intensity, and the root feature can be clearly distinguished near the center of each figure.



**Figure 11.** Example of a highly detailed environment reproduced from digital images (a) Real image of the root feature (b) Simulated camera image of the root feature. (c) A scan of the root feature by the HDL-32E scanner. (d) A scan of the root feature by the VLP-16 scanner.

Although this level of geometrical detail is probably unnecessary for LIDAR simulations, this exercise demonstrates the capability that the simulator has for highly detailed digital terrains for sensor simulations.

## 4. Discussion

While the design, development, and testing of autonomous ground vehicles is primarily done through physical experimentation, there are obvious benefits to using simulation. A recent paper on

simulation of autonomous urban driving systems stated several disadvantages of physical testing [26], including infrastructure cost, logistical difficulties, and inability to perform a statistically significant number of experiment for training and validation of autonomous algorithms.

Additionally, many important safety scenarios, such as pedestrian detection and avoidance, are dangerous or impractical to study in physical experiments. There is therefore a growing body of research on the use of simulation for the development, training, and testing of autonomous navigation algorithms for passenger-sized vehicles [5–7,26,27]. The simulator presented in this work adds to this growing field in the area of off-road autonomous navigation by providing a methodology for realistic, real-time simulation of LIDAR-vegetation interaction. This new capability can improve both the development and testing of off-road autonomous systems.

While simulation can never fully replace field testing, simulation offers several advantages over field testing. First, the simulation environment is known and controlled, meaning environmental factors can be controlled and eliminated as a source of variability in testing if desired. Second, simulation offers the ability to have perfect knowledge of “ground truth”, which can make training and testing detection and classification algorithms much simpler and faster.

Simulation tools for autonomous ground robotics can be typically divided into three broad categories, based on the physical-realism, environmental fidelity, and the purpose of the simulator. The first category of simulations are what one 2011 review called “Robotic Development Environments” (RDE) [28]. These include Gazebo [29], USARSim [30], Microsoft Robotics Developer Studio [31], and other simulators used to interactively design and debug autonomous systems in the early stage of development. Other more recent examples include customized simulations with simplified physics for closed-loop autonomy simulations in MATLAB [32,33].

While many of these tools are now defunct or unsupported due to the popularity of Gazebo, they share several traits in accordance with their intended use. First, RDE focus on ease of use and integration into the robotic development process—undoubtedly a reason behind Gazebo’s overwhelming popularity. Second, these simulators tend to avoid detailed simulation of the environment and the robot-environment interaction because this is outside the scope of their intended use. Last, RDE typically simulate in real-time or faster, allowing robotic developers to quickly design, develop, and debug autonomous systems.

The second category of robotic simulator could be called “Robotic Test Environments” (RTE), in keeping with the above nomenclature. Simulators such as the Virtual Autonomous Navigation Environment (VANE) [34], MODSIM [35], Robotic Interactive Visualization Experimentation Technology (RIVET) [36], the Autonomous Navigation Virtual Environment Laboratory (ANVEL) [37], and the CARLA [26] fall into this category. RTE are typically used to evaluate the performance of a robotic system in a realistic operational setting—necessitating a higher degree of realism in the robot-environment interaction physics. Even among RTE there is a wide range of realism in the sensor-environment interaction physics, depending on the application. For example, the RIVET, which is primarily used to study human-robot interaction, has lower-fidelity sensor simulations than then ANVEL, which has been used to evaluate mission effectiveness. The VANE has the most realistic LIDAR-environment interaction physics [38,39], but runs much slower than real-time.

The third class of simulators are empirical or semi-empirical simulators. These range from software that simply replays modified data collected in previous experiments to complex models developed from field data. For example, a realistic simulation of a Velodyne HDL-64E interacting with vegetation was developed by quantifying the statistics of LIDAR-vegetation interaction for a sensor in a particular environment and then digitizing the environment based on those statistics [23,40]. More recently, much attention has been given to Waymo’s Carcraft simulator, which uses a mixture of real and simulated data to virtualize previously measured events [27]. These empirical simulators can be quite realistic when predicting the performance of a specific autonomous system in a particular environment and are therefore useful in robotic development projects which already feature extensive

field testing. However, these empirical simulators cannot fill the need for predictive, physics-based modeling [41].

In this context, the LIDAR simulation presented in this paper is unique because it provides a more realistic model of the LIDAR-vegetation interaction than any of the other real-time simulators, but still maintains real-time or faster-than real-time computational performance. This work enables predictive, interactive simulations of robotic performance in realistic outdoor environments to be integrated into human-in-the-loop or hardware-in-the-loop testing of autonomous systems in complex outdoor environments.

## 5. Conclusions

In conclusion, this work has documented the development and validation of a high-fidelity, physics-based LIDAR simulator for autonomous ground vehicles that can be used to simulate LIDAR sensors interacting with complex outdoor environments in real-time.

The value of this capability in the development and testing of autonomous systems, as well as the improvements over past simulators, was presented in Section 4.

Future work will enhance the simulator in several ways. First, the interaction of the LIDAR with dust, snow, rain, and fog—all of which can adversely affect LIDAR performance—will be incorporated. Additionally, the environment representation will be enhanced to include retro-reflective surfaces like road signs.

**Author Contributions:** C.G. provided conceptualization, software, and writing; M.D., C.R.H. and D.W.C. provided conceptualization, reviewing, and editing.

**Funding:** Funding for this research was provided by the Center for Advanced Vehicular Systems, Mississippi State University. This research received no external funding.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Schwarz, B. LIDAR: Mapping the world in 3D. *Nat. Photonics* **2010**, *4*, 429. [[CrossRef](#)]
2. Kelly, A.; Stentz, A.; Amidi, O.; Bode, M.; Bradley, D.; Diaz-Calderon, A.; Happold, M.; Herman, H.; Mandelbaum, R.; Pilarski, T.; et al. Toward reliable off road autonomous vehicles operating in challenging environments. *Int. J. Robot. Res.* **2006**, *25*, 449–483. [[CrossRef](#)]
3. Macedo, J.; Manduchi, R.; Matthies, L. Ladar-based discrimination of grass from obstacles for autonomous navigation. In *Experimental Robotics VII*; Springer: Berlin, Germany, 2001; pp. 111–120.
4. Manduchi, R.; Castano, A.; Talukder, A.; Matthies, L. Obstacle detection and terrain classification for autonomous off-road navigation. *Auton. Robots* **2005**, *18*, 81–102. [[CrossRef](#)]
5. Chambers, D.R.; Gassaway, J.; Goodin, C.; Durst, P.J. Simulation of a multispectral, multicamera, off-road autonomous vehicle perception system with virtual autonomous navigation environment (vane). In *Electro-Optical and Infrared Systems: Technology and Applications XII; and Quantum Information Science and Technology*; International Society for Optics and Photonics: Washington, DC, USA, 2015; Volume 9648, p. 964802.
6. Carrillo, J.T.; Goodin, C.T.; Baylot, A.E. Nir sensitivity analysis with the vane. In *Infrared Imaging Systems: Design, Analysis, Modeling, and Testing XXVII*; International Society for Optics and Photonics: Washington, DC, USA, 2016; Volume 9820, p. 98200I.
7. Davis, J.E.; Bednar, A.E.; Goodin, C.T.; Durst, P.J.; Anderson, D.T.; Bethel, C.L. Computational intelligence-based optimization of maximally stable extremal region segmentation for object detection. In *Signal Processing, Sensor/Information Fusion, and Target Recognition XXVI*; International Society for Optics and Photonics: Washington, DC, USA, 2017; Volume 10200, p. 102000V.
8. Goodin, C.; Carrillo, J.T.; McInnis, D.P.; Cummins, C.L.; Durst, P.J.; Gates, B.Q.; Newell, B.S. Unmanned ground vehicle simulation with the Virtual Autonomous Navigation Environment. In *Proceedings of the 2017 International Conference on Military Technologies (ICMT)*, Brno, Czech Republic, 31 May–2 June 2017; pp. 160–165.

9. Wald, I.; Woop, S.; Benthin, C.; Johnson, G.S.; Ernst, M. Embree: A kernel framework for efficient CPU ray tracing. *ACM Trans. Gr.* **2014**, *33*, 143. [[CrossRef](#)]
10. SICK AG. *LMS200/211/221/291 Laser Measurement Systems*; Technical Description; SICK AG: Waldkirche, Germany, 2006.
11. Mississippi State University HPC2 Computing Overview, 2018. Available online: <https://www.hpc.msstate.edu/computing/hpc.php> (accessed on 30 May 2018).
12. Velodyne Acoustics, Inc. *HDL-32E User's Manual and Programming Guide*; Velodyne Acoustics, Inc.: Morgan Hill, CA, USA, 2012.
13. Woop, S.; Benthin, C.; Áfra, A.T. Embree Ray Tracing Kernels. 2017. Available online: <https://embree.github.io/data/embree-siggraph-2015-final.pdf> (accessed on 30 May 2018).
14. Matthies, L.; Bergh, C.; Castano, A.; Macedo, J.; Manduchi, R. Obstacle detection in foliage with ladar and radar. In *Robotics Research: The Eleventh International Symposium*; Springer: Berlin, Germany, 2005; pp. 291–300.
15. Gropp, W.; Thakur, R.; Lusk, E. *Using MPI-2: Advanced Features of the Message Passing Interface*; MIT press: Cambridge, MA, USA, 1999.
16. Durst, P.J.; Anderson, D.T.; Bethel, C.L. A historical review of the development of verification and validation theories for simulation models. *Int. J. Model. Simul. Sci. Comput.* **2017**, *8*, 1730001. [[CrossRef](#)]
17. Lalonde, J.F.; Vandapel, N.; Huber, D.F.; Hebert, M. Natural terrain classification using three-dimensional ladar data for ground robot mobility. *J. Field Robot.* **2006**, *23*, 839–861. [[CrossRef](#)]
18. Bradley, D.M.; Unnikrishnan, R.; Bagnell, J. Vegetation detection for driving in complex environments. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 503–508.
19. Urmson, C.; Anhalt, J.; Bagnell, D.; Baker, C.; Bittner, R.; Clark, M.; Dolan, J.; Duggins, D.; Galatali, T.; Geyer, C.; et al. Autonomous driving in urban environments: Boss and the urban challenge. *J. Field Robot.* **2008**, *25*, 425–466. [[CrossRef](#)]
20. Neuhaus, F.; Dillenberger, D.; Pellenz, J.; Paulus, D. Terrain drivability analysis in 3D laser range data for autonomous robot navigation in unstructured environments. In *Emerging Technologies & Factory Automation*; IEEE: Piscataway, NJ, USA, 2009; pp. 1–4.
21. Levinson, J.; Askeland, J.; Becker, J.; Dolson, J.; Held, D.; Kammel, S.; Kolter, J.Z.; Langer, D.; Pink, O.; Pratt, V.; et al. Towards fully autonomous driving: Systems and algorithms. In Proceedings of the 2011 IEEE Intelligent Vehicles Symposium (IV), Baden-Baden, Germany, 5–9 June 2011; pp. 163–168.
22. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? The kitti vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 16–21 June 2012; pp. 3354–3361.
23. Browning, B.; Deschaut, J.E.; Prasser, D.; Rander, P. 3d mapping for high-fidelity unmanned ground vehicle lidar simulation. *Int. J. Robot. Res.* **2012**, *31*, 1349–1376. [[CrossRef](#)]
24. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [[CrossRef](#)]
25. Weide, H. A-Star, 2018. Available online: <https://github.com/hjweide/a-star> (accessed on 30 May 2018).
26. Dosovitskiy, A.; Ros, G.; Codevilla, F.; López, A.; Koltun, V. CARLA: An open urban driving simulator. *arXiv* **2017**, arXiv:1711.03938.
27. Madrigal, A.C. *Inside Waymo's Secret World for Training Self-Driving Cars*; The Atlantic: Boston, MA, USA, 2017.
28. Michal, D.S.; Etkorn, L. A comparison of player/stage/gazebo and microsoft robotics developer studio. In Proceedings of the 49th Annual Southeast Regional Conference, Kennesaw, GA, USA, 24–26 March 2011; pp. 60–66.
29. Koenig, N.; Howard, A. Design and use paradigms for gazebo, an open-source multi-robot simulator. In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, 28 September–2 October 2004; Volume 3, pp. 2149–2154.
30. Carpin, S.; Lewis, M.; Wang, J.; Balakirsky, S.; Scrapper, C. USARSim: A robot simulator for research and education. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 1400–1405.
31. Jackson, J. Microsoft robotics studio: A technical introduction. *IEEE Robot. Autom. Mag.* **2007**, *14*. [[CrossRef](#)]
32. Castaño, F.; Beruvides, G.; Haber, R.E.; Artuñedo, A. Obstacle recognition based on machine learning for on-chip LiDAR sensors in a cyber-physical system. *Sensors* **2017**, *17*, 2109. [[CrossRef](#)] [[PubMed](#)]

33. Castaño, F.; Beruvides, G.; Villalonga, A.; Haber, R.E. Self-Tuning Method for Increased Obstacle Detection Reliability Based on Internet of Things LiDAR Sensor Models. *Sensors* **2018**, *18*, 1508. [[CrossRef](#)] [[PubMed](#)]
34. Goodin, C.; George, T.; Cummins, C.; Durst, P.; Gates, B.; McKinley, G. The virtual autonomous navigation environment: High fidelity simulations of sensor, environment, and terramechanics for robotics. In *Earth and Space 2012: Engineering, Science, Construction, and Operations in Challenging Environments*; ASCE: Reston, VA, USA, 2012; pp. 1441–1447.
35. Evans, A.W., III. *Investigating the Usefulness of Operator Aids for Autonomous Unmanned Ground Vehicles Performing Reconnaissance Tasks*; Technical Report; Army Research Lab Aberdeen Proving Ground Md Human Research and Engineering Directorate: Orlando, FL, USA, 2013.
36. Brewer, R.; Schaefer, K.E.; Avery, E. Robotic interactive visualization experimentation technology (RIVET): Game-based simulation for human-robot interaction research. In Proceedings of the 2015 Winter Simulation Conference, Huntington Beach, CA, USA, 6–9 December 2015; pp. 3224–3225.
37. Durst, P.J.; Goodin, C.; Cummins, C.; Gates, B.; Mckinley, B.; George, T.; Rohde, M.M.; Toschlog, M.A.; Crawford, J. A real-time, interactive simulation environment for unmanned ground vehicles: The autonomous navigation virtual environment laboratory (ANVEL). In Proceedings of the 2012 Fifth International Conference on Information and Computing Science (ICIC), Liverpool, UK, 24–25 July 2012; pp. 7–10.
38. Goodin, C.; Gates, B.Q.; Cummins, C.L.; George, T.R.; Durst, P.J.; Priddy, J.D. High-fidelity physics-based simulation of a UGV reconnaissance mission in a complex urban environment. In *Unmanned Systems Technology XIII*; International Society for Optics and Photonics: Washington, DC, USA, 2011; Volume 8045, p. 80450X.
39. Goodin, C.; Durst, P.J.; Prevost, Z.T.; Compton, P.J. A probabilistic model for simulating the effect of airborne dust on ground-based LIDAR. In *Active and Passive Signatures IV*; International Society for Optics and Photonics: Washington, DC, USA, 2013; Volume 8734, p. 87340D.
40. Deschaud, J.E.; Prasser, D.; Dias, M.F.; Browning, B.; Rander, P. Automatic data driven vegetation modeling for lidar simulation. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA), St. Paul, MN, USA, 14–18 May 2012; pp. 5030–5036.
41. Durst, P.J.; Goodin, C.; Gates, B.Q.; Cummins, C.L.; McKinley, B.; Priddy, J.D.; Rander, P.; Browning, B. The Need for High-Fidelity Robotics Sensor Models. *J. Robot.* **2011**, *2011*. [[CrossRef](#)]



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# LiDAR and Camera Detection Fusion in a Real-Time Industrial Multi-Sensor Collision Avoidance System

Pan Wei \*, Lucas Cagle, Tasmia Reza, John Ball and James Gafford

Center for Advanced Vehicular Systems (CAVS), Mississippi State University, Mississippi State, MS 39759, USA; ldc290@msstate.edu (L.C.); tr1044@msstate.edu (T.R.); jeball@ece.msstate.edu (J.B.); gafford@cavs.msstate.edu (J.G.)

\* Correspondence: pw541@msstate.edu; Tel.: +1-662-312-1916

Received: 14 May 2018; Accepted: 26 May 2018; Published: 30 May 2018

**Abstract:** Collision avoidance is a critical task in many applications, such as ADAS (advanced driver-assistance systems), industrial automation and robotics. In an industrial automation setting, certain areas should be off limits to an automated vehicle for protection of people and high-valued assets. These areas can be quarantined by mapping (e.g., GPS) or via beacons that delineate a no-entry area. We propose a delineation method where the industrial vehicle utilizes a LiDAR (Light Detection and Ranging) and a single color camera to detect passive beacons and model-predictive control to stop the vehicle from entering a restricted space. The beacons are standard orange traffic cones with a highly reflective vertical pole attached. The LiDAR can readily detect these beacons, but suffers from false positives due to other reflective surfaces such as worker safety vests. Herein, we put forth a method for reducing false positive detection from the LiDAR by projecting the beacons in the camera imagery via a deep learning method and validating the detection using a neural network-learned projection from the camera to the LiDAR space. Experimental data collected at Mississippi State University’s Center for Advanced Vehicular Systems (CAVS) shows the effectiveness of the proposed system in keeping the true detection while mitigating false positives.

**Keywords:** multi-sensor; fusion; deep learning; LiDAR; camera; ADAS

## 1. Introduction

Collision avoidance systems are important for protecting people’s lives and preventing property damage. Herein, we present a real-time industrial collision avoidance sensor system, which is designed to not run into obstacles or people and to protect high-valued equipment. The system utilizes a scanning LiDAR and a single RGB camera. A passive beacon is utilized to mark off quarantined areas where the industrial vehicle is not allowed to enter, thus preventing collisions with high-valued equipment. A front-guard processing mode prevents collisions with objects directly in front of the vehicle.

To provide a robust system, we utilize a Quanergy eight-beam LiDAR and a single RGB camera. The LiDAR is an active sensor, which can work regardless of the natural illumination. It can accurately localize objects via their 3D reflections. However, the LiDAR is monochromatic and cannot differentiate objects based on color. Furthermore, for objects that are far away, the LiDAR may only have one to two beams intersecting the object, making reliable detection problematic. Unlike LiDAR, RGB cameras can make detection decisions based on texture, shape and color. An RGB stereo camera can be used to detect objects and to estimate 3D positions. However, stereo cameras require extensive processing and often have difficulty estimating depth when objects lack textural cues. On the other hand, a single RGB camera can be used to accurately localize objects in the image itself (e.g., determine bounding boxes and classify objects). However, the resulting localization projected into 3D space is poor compared to the LiDAR. Furthermore, the camera will degrade in foggy or rainy environments, whereas the LiDAR

can still operate effectively. Herein, we utilize both the LiDAR and the RGB camera to accurately detect (e.g., identify) and localize objects. The focus of this paper is the fusion method, and we also highlight the LiDAR detection since it is not as straightforward as the camera-based detection. The contributions of this paper are:

- We propose a fast and efficient method that learns the projection from the camera space to the LiDAR space and provides camera outputs in the form of LiDAR detection (distance and angle).
- We propose a multi-sensor detection system that fuses both the camera and LiDAR detections to obtain more accurate and robust beacon detections.
- The proposed solution has been implemented using a single Jetson TX2 board (dual CPUs and a GPU) board to run the sensor processing and a second TX2 for the model predictive control (MPC) system. The sensor processing runs in real time (5 Hz).
- The proposed fusion system has been built, integrated and tested using static and dynamic scenarios in a relevant environment. Experimental results are presented to show the fusion efficacy.

This paper is organized as follows: Section 2 introduces LiDAR detection, camera detection and the fusion of LiDAR and camera. Section 3 discusses the proposed fusion system, and Section 4 gives an experimental example showing how fusion and the system work. In Section 5, we compare the results with and without fusion. Finally, Section 6 contains conclusions and future work.

## 2. Background

### 2.1. Camera Detection

Object detection from camera imagery involves both classification and localization of each object in which we have interest. We do not know ahead of time how many objects we expect to find in each image, which means that there is a varying number of outputs for every input image. We also do not know where these objects may appear in the image, or what their sizes might be. As a result, the object detection problem becomes quite challenging.

With the rise of deep learning (DL), object detection methods using DL [1–6] have surpassed many traditional methods [7,8] in both accuracy and speed. Based on the DL detection methods, there are also systems [9–11] improving the detection results in a computationally-intelligent way. Broadly speaking, there are two approaches for image object detection using DL. One approach is based on region proposals. Faster R-CNN [3] is one example. This method first runs the entire input image through some convolutional layers to obtain a feature map. Then, there is a separate region proposal network, which uses these convolutional features to propose possible regions for detection. Last, the rest of the network gives classification to these proposed regions. As there are two parts in the network, one for predicting the bounding box and the other for classification, this kind of architecture may significantly slow down the processing speed. Another type of approach uses one network for both predicting potential regions and for label classification. One example is you only look once (YOLO) [1,2]. Given an input image, YOLO first divides the image into coarse grids. For each grid, there is a set of base bounding boxes. For each base bounding box, YOLO predicts offsets off the true location, a confidence score and classification scores if it thinks that there is an object in that grid location. YOLO is fast, but sometimes can fail to detect small objects in the image.

### 2.2. LiDAR Detection

For LiDAR detection, the difficult part is classifying points based only on a sparse 3D point cloud. One approach [12] uses eigen-feature analysis of weighted covariance matrices with a support vector machine (SVM) classifier. However, this method is targeted at dense airborne LiDAR point clouds. In another method [13], the feature vectors are classified for each candidate object with respect to a training set of manually-labeled object locations. With its rising popularity, DL has also been used for 3D object classification. Most DL-based 3D object classification problems involve two steps: deciding



a data representation to be used for the 3D object and training a convolutional neural network (CNN) on that representation of the object. VoxNet [14] is a 3D CNN architecture for efficient and accurate object detection from LiDAR and RGBD point clouds. An example of DL for volumetric shapes is the Princeton ModelNet dataset [15], which has proposed a volumetric representation of the 3D model and a 3D volumetric CNN for classification. However, these solutions also depend on a high density (high beam count) LiDAR, so they would not be suitable for a system with an eight-beam Quanergy M8.

### 2.3. Camera and LiDAR Detection Fusion

Different sensors for object detection have their advantages and disadvantages. Sensor fusion integrates different sensors for more accurate and robust detection. For instance, in object detection, cameras can provide rich texture-based and color-based information, which LiDAR generally lacks. On the other hand, LiDAR can work in low visibility, such as at night or in moderate fog or rain. Furthermore, for the detection of the object position relative to the sensor, the LiDAR can provide a much more accurate spatial coordinate estimation compared to a camera. As both camera and LiDAR have their advantages and disadvantages, when fusing them together, the ideal algorithm should fully utilize their advantages and eliminate their disadvantages.

One approach for camera and LiDAR fusion uses extrinsic calibration. Some works on this approach [16–18] use various checkerboard patterns or look for corresponding points or edges in both the LiDAR and camera imagery. Other works on this approach [19–23] look for corresponding points or edges in both the LiDAR and camera imagery in order to perform extrinsic calibration. However, for the majority of works on this approach, the LiDARs they use have either 32 or 64 beams, which provide relatively high vertical spatial resolution, but are prohibitively expensive. Another example is [24], which estimates transformation matrices between the LiDAR and the camera. Other similar examples include [25,26]. However, these works are only suitable for modeling of indoor and short-range environments. Another approach uses a similarity measure. One example is [27], which automatically registers LiDAR and optical images. This approach also uses dense LiDAR measurements. A third kind of approach uses stereo cameras and LiDAR for fusion. One example is [28], which fuses sparse 3D LiDAR and dense stereo image point clouds. However, the matching of corresponding points in stereo images is computationally complex and may not work well if there is little texture in the images. Both of these approaches require dense point clouds and would not be effective with the smaller LiDARs such as the Quanergy M8. Comparing with previous approaches, the proposed approach in this paper is unique in two aspects: (1) a single camera and relatively inexpensive eight-beam LiDAR are used for an outdoor collision avoidance system; and (2) the proposed system is a real-time system.

### 2.4. Fuzzy Logic

When Zadeh invented the term “fuzzy” to describe the ambiguities in the world, he developed a language to describe his ideas. One of the important terms in the “fuzzy” approach is the fuzzy set, which is defined as a class of objects with a continuum of grades of membership [29]. The relation between a linguistic variable and the fuzzy set is described as “a linguistic variable can be interpreted using fuzzy sets” in [30]. This statement basically means a fuzzy set is the mathematical representation of linguistic variables. An example of a fuzzy set is “a class of tall students”. It is often denoted as  $\tilde{A}$  or sometimes  $A$ . The membership function assigns to each object a grade of membership ranging between zero and one [29].

Fuzzy logic is also proposed in [29] by Zadeh. Instead of giving either true or false conclusions, fuzzy logic uses degrees of truth. The process of fuzzy logic includes three steps. First, input is fuzzified into the fuzzy membership function. Then, IF-THEN rules are applied to produce the fuzzy output function. Last, the fuzzy output function is de-fuzzified to get specific output values. Thus, fuzzy logic can be used to reason with inputs that have uncertainty.

Zhao et al. [31] applied fuzzy logic to fuse Velodyne 64-beam LiDAR and camera data. They utilized LiDAR processing to classify objects and then fused the classifications based on the

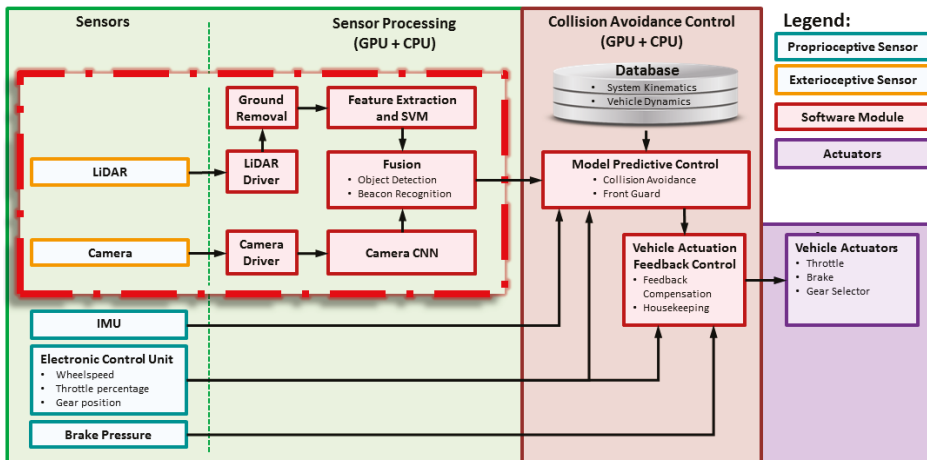
bounding box size (tiny, small, middle or large), the image classification result, the spatial and temporal context of the bounding box and the LiDAR height. Their system classified outputs as greenery, middle or obstacle. Again, the use of a Velodyne 64-beam LiDAR is prohibitively expensive for our application.

Fuzzy logic provides a mathematically well-founded mechanism for fusion with semantic meaning, such as “the LiDAR detects an object with high confidence”. This semantic statement can be quantified mathematically by the fuzzy function. The fuzzy logic system can then be used to fuse the fuzzified inputs, using the rules of fuzzy logic. Herein, we apply fuzzy logic to combine confidence scores from the camera and the LiDAR to obtain a detection score for the final fusion result.

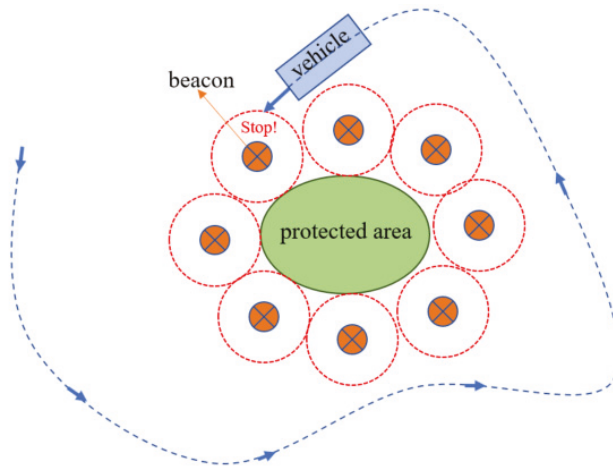
### 3. Proposed System

#### 3.1. Overview

The proposed system architecture block diagram is shown in Figure 1. In the figure, the camera and LiDAR are exteroceptive sensors, and their outputs go to the signal processing boxes. The LiDAR signal processing is discussed in Section 3.6, and the camera signal processing is discussed in Section 3.4. The LiDAR detects beacons and rejects other objects. The LiDAR signal processing output gives beacon locations as a distance in meters (in front of the industrial vehicle), the azimuth angle in degrees and a discriminant value, which is used to discriminate beacons from non-beacons. The camera reports detections as relative bounding box locations in the image, as well as the confidence of the detection and classification. The LiDAR and camera information is fused to create more robust detections. The fusion is discussed in Section 3.7.1. Figure 2 shows one application of using this system for collision avoidance. In this figure, the green area in the middle is quarantined and designated as a non-entry area for industrial vehicles.



**Figure 1.** Collision avoidance system block diagram. IMU = inertial measurement unit. GPU = graphics processing unit. CPU = central processing unit. SVM = support vector machine. CNN = convolutional neural network. Figure best viewed in color.



**Figure 2.** Using passive beacons to delineate a no-entry area.

As shown in Figure 1, the vehicle also has proprioceptive sensors, including wheel speed, steering angle, throttle position, engine control unit (ECU), two brake pressure sensors and an inertial measurement unit (IMU). These sensors allow the vehicle actuation and feedback control system to interpret the output of the system model MPC system and send commands to the ECU to control throttle and braking.

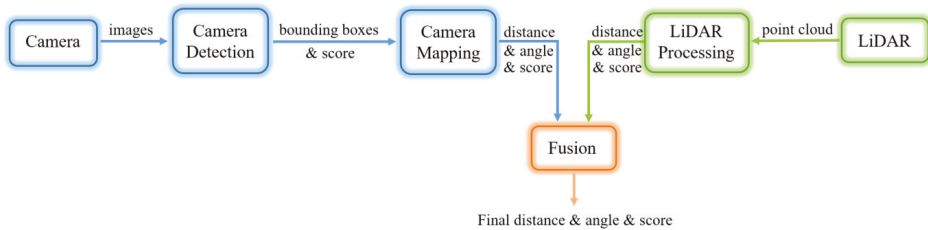
The MPC and vehicle actuation systems are critical to controlling the industrial vehicle, but are not the focus of this paper. These systems model the vehicle dynamics and system kinematics and internally maintain a virtual barrier around the detected objects. MPC by definition is a model-based approach, which incorporates a predictive control methodology to approximate optimal system control. Control solutions are determined through the evaluation of a utility function comparing computations of a branching network of future outcomes based on observations of the present environment [32–34]. The implemented MPC algorithm, to which this work is applied, predicts collision paths for the vehicle system with detected objects, such as passive beacons. Vehicle speed control solutions are computed for successive future time steps up to the prediction horizon. A utility function determines optimal control solutions from an array of possible future control solutions. Optimal control solutions are defined in this application as solutions that allow for the operation of the vehicle at a maximum possible speed defined as the speed limit, which ensures the vehicle is operating within the control limitations of the brake actuation subsystem such that a breach of a virtual boundary around detected objects may be prevented. In the implemented system, the prediction horizon may exceed 5 s. At the maximum possible vehicle speed, this horizon is consistent with the detection horizon of the sensor network. The development of the system model is partially described in [35,36].

The proposed fusion system is shown in Figure 3. In this system, we first obtain images from the camera, then through the camera object detection system, we estimate the bounding box coordinates of beacons together with the confidence scores for each bounding box. Next, the detection bounding box from the single camera is mapped through a neural network (NN) into a distance and angle estimate, in order to be able to fuse that information with the distance and angle estimates of the LiDAR processing.

On the other side, we have point cloud information from the LiDAR, then through LiDAR data processing, we estimate the distance and angle information of the detected beacons by the LiDAR. Through the LiDAR processing, we also obtain a pseudo-confidence score. With the LiDAR detection results in the form of distance and angle from both the camera and LiDAR, we fuse them together to

obtain a final distance, angle and confidence score for each detection. We give more details about this in Section 3.8.

In our application, the maximum speed of the vehicle is about 5 m per second. In order to have enough reaction time for collision avoidance, the detection frequency needs to be 5 Hz or above. We accomplish this requirement for sensor processing on one NVIDIA Jetson TX2 and use another TX2 for the MPC controller. More details are given in Section 3.2.



**Figure 3.** Proposed fusion system high-level block diagram.

### 3.2. Real-Time Implementation

To achieve the desired 5-Hz update rate for real-time performance, multiple compromises were made to improve software speed enough to meet the requirements. Notably, a naive approach for removal of ground points from the LiDAR point cloud was used instead of a more complex, and accurate, method. Furthermore, a linear SVM, with a limited feature set, was used as opposed to a kernel SVM or other non-linear methods. The specifics of these methods are covered in Section 3.6.

To manage intra-process communication between sensor drivers and signal processing functions, the Robot Operating System (ROS) was used [37]. ROS provides already-implemented sensor drivers for extracting point cloud and image data from the LiDAR and camera, respectively. Additionally, ROS abstracts and simplifies intra-process data transfer by using virtualized Transmission Control Protocol (TCP) connections between the individual ROS processes, called nodes. As a consequence of this method, multiple excess memory copies must be made, which begin to degrade performance with large objects like LiDAR point clouds. To ameliorate this effect, ROS Nodelets [38] were used for appropriate LiDAR processes. Nodelets builds on ROS Nodes to provide a seamless interface for pooling multiple ROS processes into a single, multi-threaded process. This enables efficient zero-copy data transfers between signal processing steps for the high bandwidth LiDAR computations. This proved essential due to the limited resources of the TX2, with central processing unit (CPU) utilization averaging at over 95% even after Nodelets optimization.

In Figure 4, the high-level flowchart of the relevant ROS nodes can be seen along with their execution location on the Jetson TX2's hardware. The NVIDIA Jetson TX2 has a 256 CUDA (Compute Unified Device Architecture) core GPU, a Dual CPU, 8 GB of memory and supports various interfaces including Ethernet [39]. As the most computationally-intensive operation, the camera CNN has the entire graphical processing unit (GPU) allocated to it. The sensor drivers, signal processing and fusion nodes are then relegated to the CPU. Due to the high core utilization from the LiDAR, care has to be taken that enough CPU resources are allocated to feed images to the GPU or significant throttling of the CNN can occur.

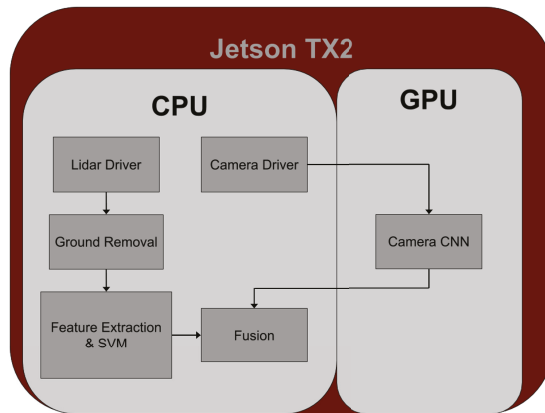


Figure 4. ROS dataflow on Jetson TX2.

### 3.3. Beacon

The beacon is constructed of a high-visibility 28" orange traffic cone with a 2" diameter highly-reflective pole extending two meters vertically. A beacon is shown in Figure 5. The beacon presents as a series of high intensity values in a LiDAR scan and thus provides a high signal level compared to most background objects. A beacon delineates an area in the industrial complex that is off limits to the vehicle, usually to protect high-valued assets.

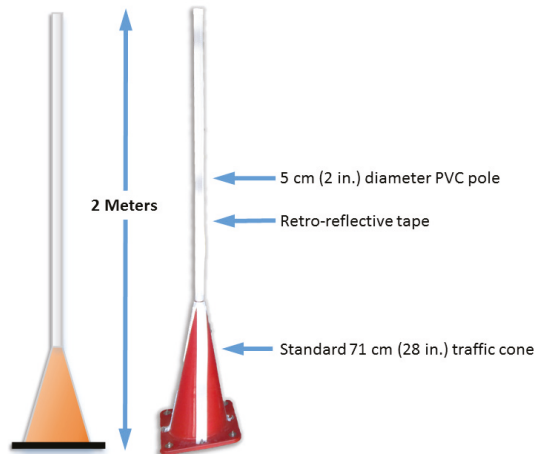


Figure 5. Beacon.

### 3.4. Camera Detection

Herein, we apply a deep-learning model called YOLO [1,2] to detect objects present in the single camera image. YOLO outperforms most other deep learning methods in speed, which is crucial for our application. YOLO achieves such speed by combining both region and class prediction into one network, unlike region-based methods [3,5,40], which have a separate network for region proposals.

A large training dataset of beacon images presented in Section 4 was collected on different days, different times of day and weather conditions (e.g., full sun, overcast, etc.) and was hand-labeled by

our team members. Figure 6 shows how we draw bounding boxes and assign labels on an image from the camera.



Figure 6. Labeling for camera detection.

The dataset we collected covers the camera’s full field of view (FOV) ( $-20^{\circ}$  to  $20^{\circ}$  azimuth) and detection range (5 to 40 m) (FLIR Chameleon3 USB camera with Fujinon 6-mm lens). It reliably represents the detection accuracy from different angles and distances. Figure 7 shows the markings on the ground that help us to locate the beacon at different angles and distances. These tests were performed early in the project to allow us to estimate the LiDAR detection accuracies.



Figure 7. (a) Markings on the ground. (b) View from the camera. Markings on the ground covering the full FOV and detection range of the camera.

All of the beacons have a highly consistent construction, with near-identical size, shape and coloring. These characteristics allow the detection algorithm to reliably recognize beacons at a range of up to 40 m under a variety of lighting conditions. Figure 8 shows one example of the camera detection of beacons.

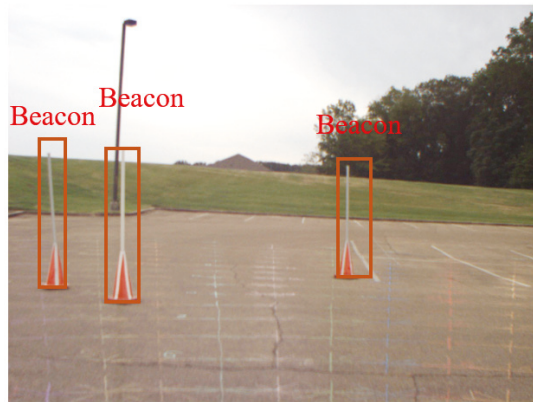


Figure 8. Camera detection example. Best viewed in color.

### 3.5. LiDAR Front Guard

The LiDAR detection mode is designed to detect beacons. However, other objects can be in the direct or near-direct path of the industrial vehicle. A front guard system is implemented with the LiDAR. Since many objects lack any bright points, thus being missed by the LiDAR beacon detection, we also employed a front-guard system, which clusters points in a rectangular area above the ground directly in front of the industrial vehicle. The front guard is then used to keep the vehicle from striking obstacles or people. After ground point removal, any remaining points within a rectangular solid region directly in front of the vehicle will be detected and reported to the control system.

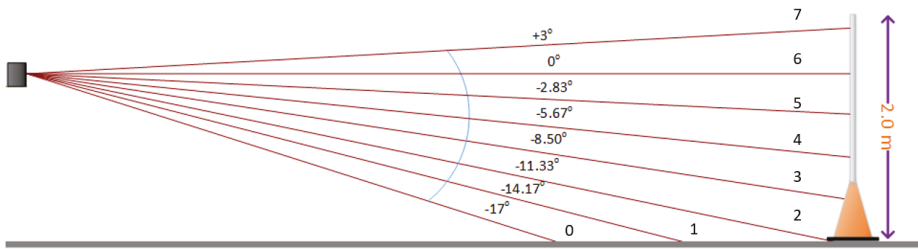
### 3.6. LiDAR Beacon Detection

This section discusses the LiDAR's beacon detection system.

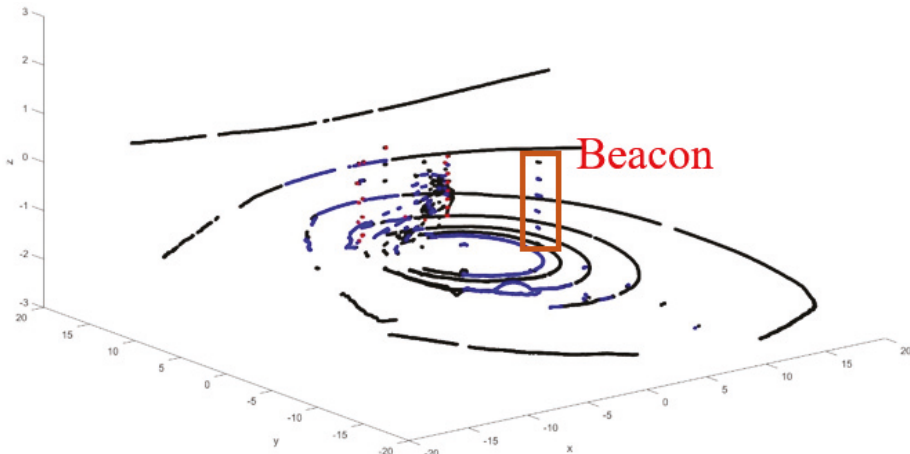
#### 3.6.1. Overview

The LiDAR data we collect are in the form of a three-dimensional (3D) and 360 degree field-of-view (FOV) point cloud, which consists of  $x$ ,  $y$ ,  $z$  coordinates, along with intensity and beam number (sometimes called ring number). The coordinates  $x$ ,  $y$  and  $z$  represent the position of each point relative to the origin centered within the LiDAR. Intensity represents the strength of returns and is an integer for this model of LiDAR. Highly reflective objects such as metal or retro-reflective tape will have higher intensity values. The beam number represents in which beam the returned point is located. The LiDAR we use transmits eight beams. The beam numbers are shown in Figure 9. Assuming the LiDAR is mounted level to the ground, then Beam 7 is the top beam and is aimed upward approximately  $3^\circ$ . Beam 6 points horizontally. The beams are spaced approximately  $3^\circ$  apart. Figure 10 shows one scan of the LiDAR point cloud. In this figure, the ground points are shown with black points. The beacon presents as a series of closely-spaced points in the horizontal direction, as shown in the square.

General object detection in LiDAR is a difficult and unsolved problem in the computer vision community. Objects appear at different scales based on their distance to the LiDAR, and they can be viewed from any angle. However, in our case, we are really only interested in detecting the beacons, which all have the same geometries. In our application, the industrial vehicle can only go so fast, so that we only need to detect beacons about 20 m away or closer. The beacons are designed to have very bright returns. However, other objects can also have bright returns in the scene, such as people wearing safety vests with retro-reflective tape, or other industrial vehicles. Herein, we propose a system that first identifies clusters of points in the scene that present with bright returns. Next, a linear SVM classifier is used to differentiate the beacon clusters from non-beacon clusters.



**Figure 9.** LiDAR beam spacing visualization and beam numbering. The LiDAR is the small black box on the left. Beam 6 is the horizontal beam (assuming the LiDAR is level).



**Figure 10.** Detection via LiDAR.

The LiDAR detection system works by utilizing intensity-based and density-based clustering (a modified DBSCAN algorithm) on the LiDAR point cloud. The system then examines all points near the cluster centroid by extracting features and using a linear SVM to discriminate beacons from non-beacons. Details of these algorithm are discussed in the subsections below.

### 3.6.2. LiDAR Clustering

A modified DBSCAN clustering algorithm [41], which clusters based on point cloud density, as well as intensity is used to cluster the bright points, as shown in Algorithm 1. The cluster parameter  $\epsilon$  was empirically determined to be 0.5 m based on the size of the beacon. Values much larger could group two nearby beacons (or a beacon and another nearby object) together into one cluster, and we wanted to keep them as separate clusters.

In Equation (1), the distances are estimated using Euclidean distances with only the  $x$  (front-to-back) and  $y$  (left-to-right) coordinates. Effectively, this algorithm clusters bright LiDAR points by projecting them down onto the  $x$ - $y$  plane. Alternately, all three coordinates,  $x, y, z$  could be used, but the features don not require this because they will be able to separate tall and short objects. This approach is also more computationally efficient than using all three coordinates in the clustering algorithm.



**Algorithm 1:** LiDAR bright pixel clustering.

**Input:** LiDAR point cloud  $P = \{x_j, y_j, z_j, i_j, r_j\}$  with  $NP$  points.

**Input:** High-intensity threshold:  $T_H$ .

**Input:** Cluster distance threshold:  $\epsilon$  (meters).

**Input:** Ground Z threshold:  $T_G$  (meters).

**Output:** Cluster number for each bright point in the point cloud.

**Remove non-return points:**

**for** each point in the point cloud **do**

  Remove all non-return points (NaN's) from the the point cloud.

**Remove ground points:**

**for** each point in the modified point cloud **do**

  Remove all points with Z-values below  $T_G$  from the point cloud.

**Remove non-bright points:**

**for** each point in the modified point cloud **do**

  Remove all points with intensity values below  $T_H$  from the point cloud.

**Cluster bright points:**

Assign all points to Cluster 0.

Set  $cl \leftarrow 0$ .

**for** each point  $P_j$  in the modified point cloud **do**

**if** The point does not belong to a cluster **then**

**Add point to cluster**

    Increment the number of clusters:  $cl \leftarrow cl + 1$ .

    Assign  $P_j$  to cluster  $cl$ .

    Set the centroid of cluster  $cl$  to  $P_j$ .

**Scan through all remaining points and re-cluster if necessary.**

**for** each point  $P_m$  with  $j < m \leq NP$  **do**

**if** Distance from point  $P_m$  to the centroid of cluster  $cl$  has distance  $< \epsilon$  **then**

        Add  $P_m$  to cluster  $cl$ .

        Recalculate the centroid of cluster  $cl$ .

### 3.6.3. LiDAR Feature Extraction

To extract the features, the ground points must be removed before feature processing occurs, or else false alarms can occur. Since this industrial application has a smooth, flat area for the ground, we employ a simple vertical threshold to remove ground points. If this system were to be generalized to areas with more varying ground conditions, ground estimation methods such as those in [42–47] could be utilized. However, for this particular application, this was not necessary.

Next, the point intensities are compared to an empirically-determined threshold. The beacon is designed so that it provides bright returns to the LiDAR via the retro-reflective vertical pole. This works well, but there are also other objects in the scene that can have high returns, such as other industrial vehicles with retro-reflective markings or workers wearing safety vests with retro-reflective stripes. In order to classify objects as beacons and non-beacons, hand-crafted features are utilized (these are discussed below). After ground-removal and thresholding the intensity points, we are left with a set of bright points. A second set of intensity points is also analyzed, which consists of all of the non-ground points (e.g., the point cloud after ground removal). The points in the bright point cloud are clustered. Beacons appear as tall, thin objects, whereas all other objects are either not as

tall, or wider. We extract features around the cluster center in a small rectangular volume centered at each object’s centroid. We also extract features using a larger rectangular volume also centered around the objects centroid. Features include counting the number of bright points in each region, determining the  $x$ ,  $y$  and  $z$  extents of the points in each region, etc. Beacons mainly have larger values in the smaller region, while other objects have values in the larger regions.

The two regions are shown in Figure 11. The idea of using an inner and an outer analysis region is that a beacon will mostly have bright points located in the inner analysis region, while other objects, such as humans, other industrial vehicles, etc., will extend into the outer regions. Equations (1) and (2) define whether a LiDAR point  $p_j$  with coordinates  $(x_j, y_j, z_j)$  is in the inner region or outer region, respectively, where the object’s centroid has coordinates  $(x_C, y_C, z_C)$ . Reference Figure 11 for a top-down illustration of the inner and outer regions.

Figure 11 shows an example beacon return with the analysis windows superimposed. Both the inner and outer analysis regions have  $x$  and  $y$  coordinates centered at the centroid location. The inner analysis region has a depth ( $x$  coordinate) of 0.5 m and a width ( $y$  coordinate) of 0.5 m, and the height includes all points with  $z$  coordinate values of  $-1.18$  m and above. The outer region extends 2.0 m in both  $x$  and  $y$  directions and has the same height restriction as the inner region. These values were determined based on the dimensions of the beacon and based on the LiDAR height. The parameters  $\Delta x_I, \Delta y_I$  and  $z_{MIN}$  define the inner region relative to the centroid coordinates. Similarly, the parameters  $\Delta x_O, \Delta y_O$  and  $z_{MIN}$  define the outer region relative to the centroid coordinates. A point is in the inner region if:

$$\begin{aligned} \left(x_C - \frac{\Delta x_I}{2}\right) \leq x_j \leq \left(x_C + \frac{\Delta x_I}{2}\right) \text{ and} \\ \left(y_C - \frac{\Delta y_I}{2}\right) \leq y_j \leq \left(y_C + \frac{\Delta y_I}{2}\right) \text{ and} \\ z_{MIN} \geq z_j, \end{aligned} \tag{1}$$

and a point is in the outer region if:

$$\begin{aligned} \left(x_C - \frac{\Delta x_O}{2}\right) \leq x_j \leq \left(x_C + \frac{\Delta x_O}{2}\right) \text{ and} \\ \left(y_C - \frac{\Delta y_O}{2}\right) \leq y_j \leq \left(y_C + \frac{\Delta y_O}{2}\right) \text{ and} \\ z_{MIN} \geq z_j. \end{aligned} \tag{2}$$

In order to be robust, we also extract features on the intensity returns. A linear SVM was trained on a large number of beacon and non-beacon objects, and each feature was sorted based on its ability to distinguish beacons from non-beacons. In general, there is a marked increase in performance, then the curve levels out. It was found that ten features were required to achieve very high detection rates and low false alarms. These ten features were utilized in real time, and the system operates in real time at the frame rate of the LiDAR, 5 Hz. The system was validated by first running on another large test set independent of the training set, with excellent performance as a result as presented in Section 3.6.4. Then, extensive field tests were used to further validate the results as presented in Section 4.

After detection, objects are then classified as beacons or non-beacons. These are appended to two separate lists and reported by their distance in meters from the front of the industrial vehicle, their azimuth angle in degrees and their discriminate value. Through extensive experimentation, we can reliably see beacons in the LiDAR’s FOV from 3 to 20 m.

LiDAR feature extraction follows the overall algorithm shown in Algorithm 2. The input is the LiDAR point cloud  $P = \{x_j, y_j, z_j, i_j, r_j\}$ , where  $j$  is the index variable for the  $j$ -th point, and  $x$ ,  $y$ ,  $z$ ,  $i$  and  $r$  refer to the  $x$  point in meters, the  $y$  point in meters, the  $z$  point in meters, the intensity and the beam number, respectively, for the  $j$ -th point. Note that all coordinates are relative to the center of the

LiDAR at point (0, 0, 0). The  $x$  coordinate is positive in front of the LiDAR, and negative behind. The  $y$  coordinate is positive to the left of the LiDAR and negative to the right. The  $z$  coordinate is positive above the LiDAR and negative below.

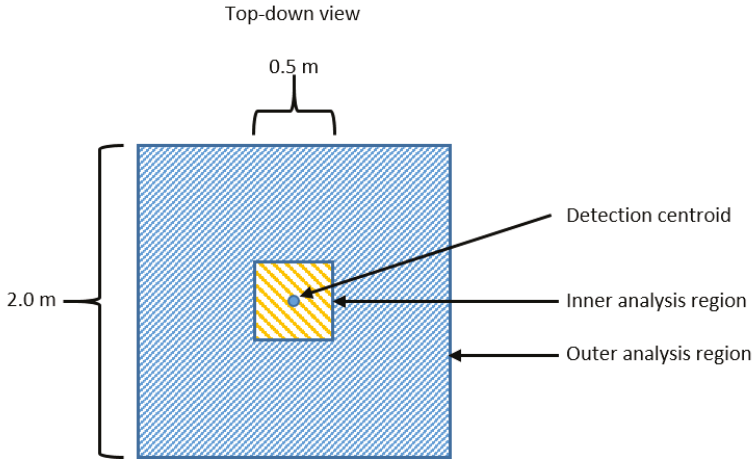


Figure 11. LiDAR detection regions (inner and outer) visualized from a top-down view.

---

**Algorithm 2:** LiDAR high-level feature extraction preprocessing.

---

**Input:** LiDAR point cloud  $P = \{x_j, y_j, z_j, i_j, r_j\}$ .

**Input:** Low-intensity threshold:  $T_L$ .

**Input:** High-intensity threshold:  $T_H$ .

**Input:** Ground  $Z$  threshold:  $T_G$  (meters).

**Output:** Feature vector  $f$ .

**Remove non-return points:**

**for each point in the point cloud do**  
 └ Remove all non-return points (NaN's) from the point cloud.

**Remove ground points:**

**for each point in the modified point cloud do**  
 └ Remove all points with  $Z$ -values below  $T_G$  from the point cloud.

**Create threshold point clouds:**

Set  $P_{HT} = \emptyset$ .

Set  $P_{LT} = \emptyset$ .

**for each point  $P_j$  in the modified point cloud do**

└ **if Point  $P_j$  has intensity  $\geq T_H$  then**  
 └ Add  $P_j$  to  $P_{HT}$ .

└ **if Point  $P_j$  has intensity  $\geq T_L$  then**  
 └ Add  $P_j$  to  $P_{LT}$ .

**Extract features:**

Extract features  $f$  using Algorithm 3.

---

In order to extract features for objects, any points that did not provide a return to the LiDAR are removed. These points will present as NaN's (not a number) in the LiDAR point cloud. Next, the estimated ground points are removed. The non-ground points are divided into two data subsets: The high-threshold (HT) data and the low-threshold (LT) data. The HT data points only contain high-intensity returns, while the LT data contain points greater than or equal to the low-intensity threshold. Both data subsets do not contain any ground points. The high-intensity threshold was set to 15 and the low-intensity threshold to zero. These threshold values were determined experimentally based on the examination of multiple beacon returns at various distances.

---

**Algorithm 3:** LiDAR feature extraction.

---

**Input:** LiDAR high-intensity point cloud  $P_{HT} = \{x_j, y_j, z_j, i_j, r_j\}$ .

**Input:** LiDAR low-intensity point cloud  $P_{LT} = \{x_j, y_j, z_j, i_j, r_j\}$ .

**Input:** Inner region  $x$ -extent:  $\Delta x_I$  (meters).

**Input:** Inner region  $y$ -extent:  $\Delta y_I$  (meters).

**Input:** Outer region  $x$ -extent:  $\Delta x_O$  (meters).

**Input:** Outer region  $y$ -extent:  $\Delta y_O$  (meters).

**Input:** LiDAR height above ground:  $Z_L = 1.4$  (meters).

**Output:** Feature vector  $\mathbf{f}$ .

**Cluster the high-intensity point cloud:**

**for** each point in  $p$  in the high-intensity point cloud **do**

    Cluster points and determine cluster centers.

**Calculate features:**

**for** each cluster center point  $c = (x_C, y_C, z_C)$  in the point cloud **do**

    Determine all points in  $P_{HT}$  in the inner region using Equation (1), and calculate Features 1, 13 and 17 from Table 1.

    Determine all points in  $P_{HT}$  in the outer region using Equation (2), and calculate Feature 4 from Table 1.

    Determine all points in  $P_{LT}$  in the inner region using Equation (1), and calculate Features 6, 7, 9 10, 11, 14, 16 and 18 from Table 1.

    Determine all points in  $P_{LT}$  in the outer region using Equation (2), and calculate Features 2, 3, 5 8, 12, 15, 19 and 20 from Table 1.

**Return**  $\mathbf{f} = [f_1, f_2, f_3, \dots, f_{20}]$ .

---

Table 1 describes the extracted features. For example, Feature 1 is generated from the high threshold data, which consists solely of high-intensity points. The data is analyzed only for high-intensity points in the inner analysis region. This feature simply computes the  $Z$  (height) extent, that is, the maximum  $Z$  value minus the minimum  $Z$  value. Note that some of the features are only processed on certain beams, such as Feature 2. The features are listed in order of their discriminate power in descending order, e.g., Feature 1 is the most discriminate, Feature 2 the next most discriminate, etc. Initially, it was unknown how effective the features would be. We started with 134 features, and these were ranked in order of their discriminating power using the score defined as:

$$SCORE = 500 \frac{TP}{TP + FN} + 500 \frac{TN}{TN + FP} \tag{3}$$

where  $TP$ ,  $TN$ ,  $FP$  and  $FN$  are the number of true positives, true negatives, false positive and false negatives, respectively [48]. A higher score is better, and scores range from zero to 1000. This score was used since training with overall accuracy tended to highly favor one class and provided poor

generalizability, which may be a peculiarity of the chosen features and the training set. Figure 12 shows the score values versus the number of features (where the features are sorted in descending score order). The operating point is shown for  $M = 20$  features (circles in the plot). The features generalize fairly well since the testing curve is similar to the training curve. In our system, using 20 features provided a good balance of performance versus computational complexity to compute the features. Each LiDAR scan (5-Hz scan rate) requires calculating features for each cluster. Experimentation showed that a total of 20 features was near the upper limit of what the processor could calculate and not degrade the LiDAR processing frame rate.

**Table 1.** Feature descriptions. Data subsets: HT = high threshold data. LT = low threshold data. The LiDAR rings are shown in Figure 9.

Feature Number	Data Subset	Analysis Region	Description
1	HT	Inner	Extent of Z in cluster. Extent $\{Z\} = \max\{Z\} - \min\{Z\}$ .
2	LT	Outer	Max of X and Y extents in cluster, Beam 7. This is $\max\{\text{extent}\{X\}, \text{extent}\{Y\}\}$ . Extent $\{X\} = \max\{X\} - \min\{X\}$ . Extent $\{Y\} = \max\{Y\} - \min\{Y\}$ .
3	LT	Outer	Max of X and Y extents in cluster, Beam 5. This is $\max\{\text{extent}\{X\}, \text{extent}\{Y\}\}$ . Extent $\{X\} = \max\{X\} - \min\{X\}$ . Extent $\{Y\} = \max\{Y\} - \min\{Y\}$ .
4	HT	Outer	Max $\{Z$ in cluster - LiDAR height}. Z is vertical (height) of LiDAR return.
5	LT	Outer	Extent of Z in cluster. Extent $\{Z\} = \max\{Z\} - \min\{Z\}$ .
6	LT	Inner	Number of valid points in cluster, Beam 7.
7	LT	Inner	Max of X and Y extents in cluster, Beam 6. This is $\max\{\text{extent}\{X\}, \text{extent}\{Y\}\}$ . Extent $\{X\} = \max\{X\} - \min\{X\}$ . Extent $\{Y\} = \max\{Y\} - \min\{Y\}$ .
8	LT	Outer	Number of points in cluster, Beam 5.
9	LT	Inner	Extent of X in cluster. Extent $\{X\} = \max\{X\} - \min\{X\}$ .
10	LT	Inner	Number of points in cluster, Beam 4.
11	LT	Inner	Number of points in cluster, Beam 5.
12	LT	Outer	Number of points in cluster, Beam 6.
13	HT	Inner	Number of points in cluster, Beam 6.
14	LT	Inner	Max of X and Y extents, Beam 5. This is $\max\{\text{extent}\{X\}, \text{extent}\{Y\}\}$ . Extent $\{X\} = \max\{X\} - \min\{X\}$ . Extent $\{Y\} = \max\{Y\} - \min\{Y\}$ .
15	LT	Outer	Number of points in cluster divided by the cluster radius in Beam 5.
16	LT	Inner	Extent of X in cluster. Extent $\{X\} = \max\{X\} - \min\{X\}$ .
17	HT	Inner	Number of points in cluster, Beam 7.
18	LT	Inner	Number of points in cluster.
19	LT	Outer	Number of points in cluster.
20	LT	Outer	Extent of Z in cluster. Extent $\{Y\} = \max\{Y\} - \min\{Y\}$ .

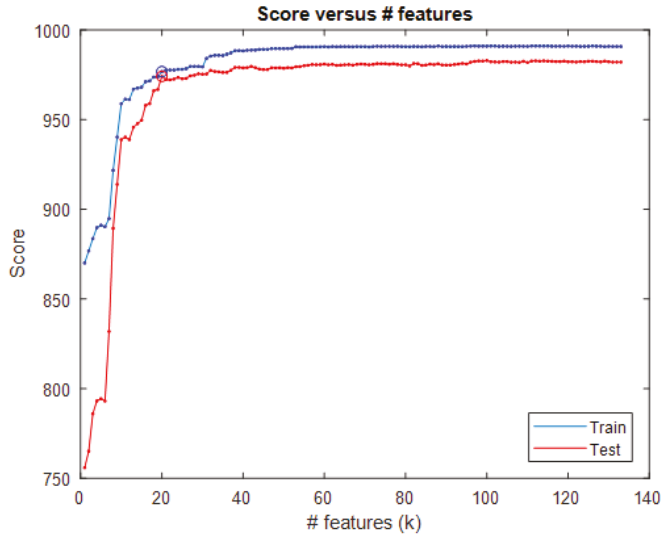


Figure 12. Score value versus number of concatenated features.

### 3.6.4. SVM LiDAR Beacon Detection

To detect the beacons in the 3D LiDAR point cloud, we use a linear SVM [49], which makes a decision based on a learned linear discriminant rule operating on the hand-crafted features we have developed for differentiating beacons. Herein, liblinear was used to train the SVM [50]. A linear SVM operates by finding the optimal linear combination of features that best separates the training data. If the  $M$  testing features for a testing instance are given by  $\mathbf{f} = [f_1, f_2, \dots, f_M]^T$ , where the superscript  $T$  is a vector transpose operator, then the SVM evaluates the discriminant:

$$D = \mathbf{f} \cdot \mathbf{w} + b, \tag{4}$$

where  $\mathbf{w}$  is the SVM weight vector and  $b$  is the SVM bias term. Herein,  $M$  was chosen to be 20. Equation (4) applies a linear weight to each feature and adds a bias term. The discriminant is optimized during SVM training. The SVM optimizes the individual weights in the weight vector to maximize the margin and provide the best overall discriminant capability of the SVM. The bias term lets the optimal separating hyperplane drift from the origin. Figure 13 shows an example case with two features. In this case, the  $D = 0$  hyperplane is the decision boundary. The  $D = +1$  and  $D = -1$  hyperplanes are determined by the support vectors. The SVM only uses the support vectors to define the optimal boundary.

The SVM was trained with beacon and non-beacon data extracted over multiple data collections. There were 13,190 beacon training instances and 15,209 non-beacon training instances. The test data had 12,084 beacons and 5666 non-beacon instances.

In this work,  $M = 20$  features were chosen. Each feature is first normalized by subtracting the mean of the feature values from the training data and dividing each feature by four times the standard deviation plus  $10^{-5}$  (in order to prevent division by very small numbers). Subtracting the mean value centers the feature probability distribution function (PDF) around zero. Dividing by four times the standard deviation maps almost all feature values into the range  $[-1, 1]$ , because most of the values lie within  $\pm 4\sigma_k$ . The normalized features are calculated using:

$$f'_k = \frac{f_k - \mu_k}{4\sigma_k + 10^{-5}} \tag{5}$$

where  $\mu_k$  is the mean value of feature  $k$  and  $\sigma_k$  is the standard deviation of feature  $k$  for  $k = 1, 2, \dots, M$ . The mean and standard deviation are estimated from the training data and then applied to the test data. The final discriminant is computed using Equation (4) where the feature vector is the normalized feature vector.

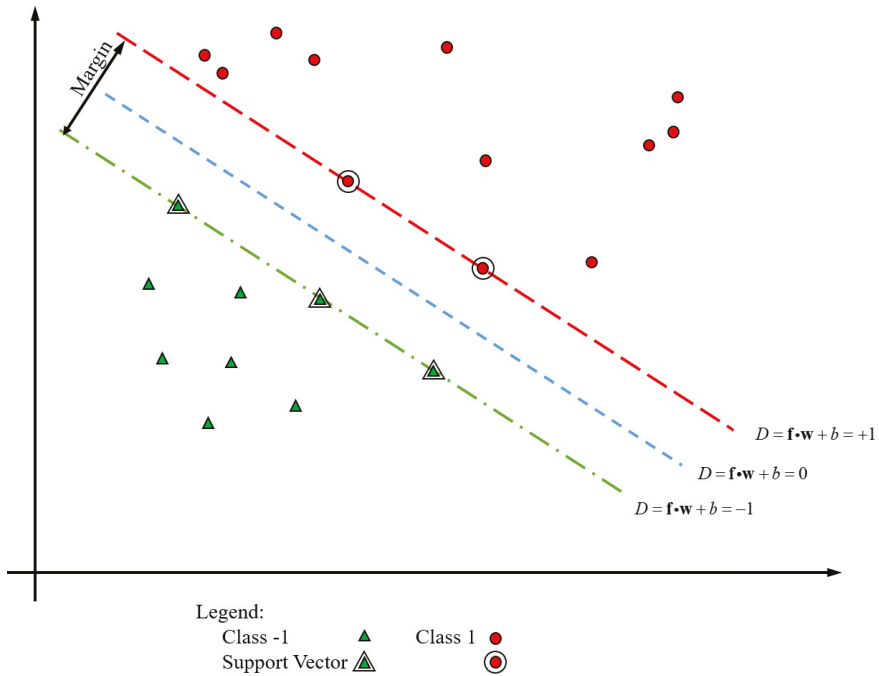


Figure 13. SVM 2D example. The margin is the distance from the  $D = -1$  to  $D = 1$  hyperplanes.

The optimal feature weights are shown in Figure 14. Since all the features are normalized, Features 1, 3 and 8 have the most influence on the final discriminant function.

If the discriminant  $D \leq 0$ , then the object is declared a beacon. Otherwise, it is declared a non-beacon (and ignored by the LiDAR beacon detection system). Once the SVM is trained, implementing the discriminant given in Equation (4) is trivial and uses minimal processing time.

Figure 15 shows the PDF of the LiDAR discriminant values for beacons and non-beacons, respectively. The PDFs are nearly linearly separable. The more negative the discriminant values, the more the object is beacon-like.

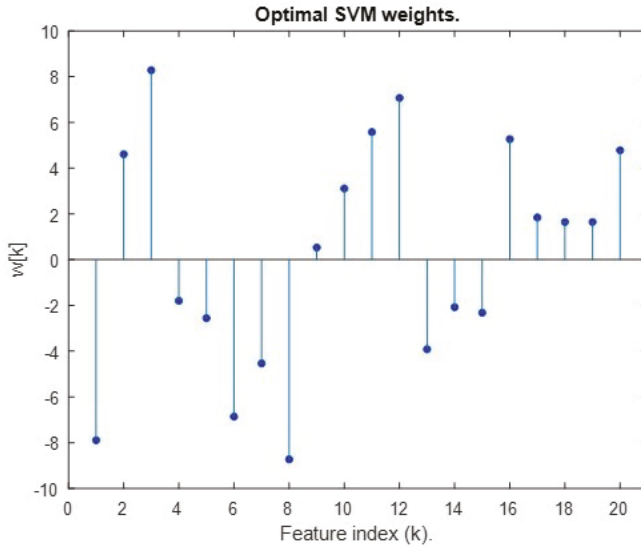


Figure 14. Optimal feature weights chosen by the SVM training optimizer.

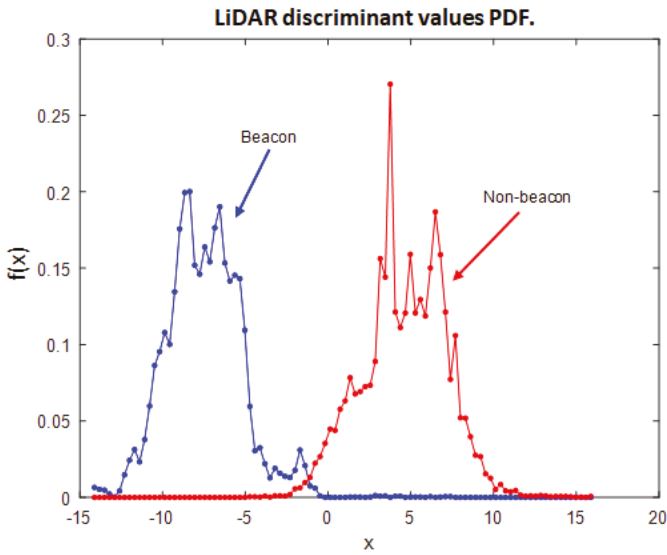


Figure 15. LiDAR discriminant values.

### 3.7. Mapping

This section discusses the camera detection mapping from bounding box coordinates to distance and angle. It also discusses the mapping of LiDAR discriminate value from the range  $[-\infty, \infty]$  to  $[0, 1]$ .

#### 3.7.1. Camera Detection Mapping

The LiDAR and the camera both provide information on object detection, although in entirely different coordinate spaces. The LiDAR provides accurate angle and range estimates with the



discriminant value offering a rough metric of confidence. The camera processing algorithms return bounding box coordinates, class labels and confidence values in the range  $[0, 1]$ . However, effective fusion requires a common coordinate space. Therefore, a mapping function is required to merge these sensor measurements into a shared coordinate space. To build this mapping function, the camera and LiDAR were mounted to the industrial vehicle, and the LiDAR’s accurate position measurements, alongside the camera’s bounding boxes, were used to collect training data. These data were then used to train a neural network. The end result was a neural network that could project the camera’s bounding boxes into a range and angle estimate in the LiDAR’s coordinate frame, as seen in Figure 16. In this figure, the middle portion simply represents the bounding boxes from the camera system, and the smaller boxes before the mapped results represent the NN mapper.

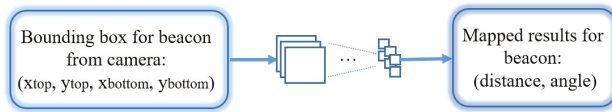


Figure 16. Detection mapping for the camera.

The detailed NN structure is shown in Figure 17. It consists of ten fully-connected layers: eight layers of ten neurons and two layers of twenty neurons. We choose this structure because of its simplicity to compute while outperforming other linear and non-linear regression methods.

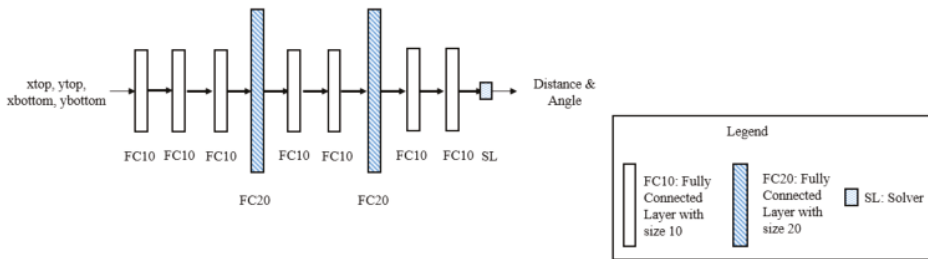


Figure 17. NN structure for detection mapping for the camera.

### 3.7.2. LiDAR Discriminate Value Mapping

With the camera bounding boxes mapped to LiDAR range angle, there still remains the mapping of the discriminate value from LiDAR and confidence scores from the camera into a shared space. For this, the  $[0, 1]$  range of the camera confidence offers a more intuitive measure than the LiDAR’s  $[-\infty, \infty]$  range for the discriminate value. Thus, the mapping function for this component will map the LiDAR discriminate to a  $[0, 1]$  space for fusion with the camera. To do this, a logistic sigmoid of the form:

$$f(x) = \frac{1}{1 + e^{-\alpha \cdot x}} \tag{6}$$

was used. The discriminate value is multiplied by a gain term,  $\alpha$ . The end result is a function that maps the discriminate value to a pseudo-confidence score with a range similar enough to the camera’s confidence score for effective fusion.

### 3.8. Detection Fusion

In this section, we discuss the fusion algorithm and hyperparameter optimization.

### 3.8.1. Fusion Algorithm

With these two mapping components, a full pipeline can now be established to combine the LiDAR and camera data into forms similar enough for effective fusion. Using distance and angle information from both the camera and LiDAR, we can correlate LiDAR and camera detections together. Their confidence scores can then be fused as shown in Algorithm 4.

---

**Algorithm 4:** Fusion of LiDAR and Camera detection.

---

**Input:** Detection from LiDAR in the form of [distance, angle, pseudo-confidence score].

**Input:** Detection from camera in the form of [distance, angle, confidence score].

**Input:** Angular threshold:  $A$  for angle difference between fused camera and LiDAR detection.

**Input:** Confidence threshold:  $C$  for final detection confidence score threshold.

**Output:** Fused detection in the form of [distance, angle, detection confidence].

**for each detection from camera do**

**if a corresponding LiDAR detection, with angle difference less than  $A$ , exists then**  
    | Fuse by using the distance and angle from LiDAR, and combine confidence scores  
    | using fuzzy logic to determine the fused detection confidence.

**else**  
    | Create a new detection from camera, using the distance, angle and confidence score  
    | from camera as the final detection result.

**for each LiDAR detection that does not have a corresponding camera detection do**

    Create a new detection from LiDAR, using the distance, angle and confidence score from  
    LiDAR as the final detection result.

**for each fused detection do**

    Remove detections with final confidences below the  $C$  confidence threshold.

Return fused detection results.

---

In this algorithm, the strategy for fusion of the distance and angle estimates is different from the fusion of confidence scores, which is shown in Figure 18. In the process of distance and angle fusion, for each camera detection, when there is a corresponding detection from LiDAR, we will use the distance and angle information from LiDAR, as LiDAR can provide much more accurate position estimation. When there is no corresponding LiDAR detection, we will use the distance, angle and confidence score from the camera to create a new detection. For each detection from LiDAR that does not have corresponding camera detection, we will use the distance, angle and confidence score from LiDAR as the final result. For confidence score fusion, we use fuzzy logic to produce a final confidence score when there is corresponding camera and LiDAR detection. The fuzzy rules used in the fuzzy logic system are as follows:

1. If the LiDAR confidence score is high, then the probability for detection is high.
2. If the camera confidence score is high, then the probability for detection is high.
3. If the LiDAR confidence score is medium, then the probability for detection is medium.
4. If the LiDAR confidence score is low and the camera confidence score is medium, then the probability for detection is medium.
5. If the LiDAR confidence score is low and the camera confidence score is low, then the probability for detection is low.

The fuzzy rules are chosen in this way because of the following reasons: First, if either the LiDAR or the camera shows a high confidence in their detection, then the detections are very likely to be true. This is the reason for Rules 1 and 2: the LiDAR is robust at detection up to about 20 m. Within that

range, the confidence score from the LiDAR is usually high or medium. Beyond that range, the value will drop to low. This is the reason for Rule 3. Third, beyond the 20-m detection range, the system will solely rely on the camera for detection. This is the reason for Rules 4 and 5.

Figure 19 shows the process of using fuzzy logic to combine the confidence scores from the camera and LiDAR. The camera input, LiDAR input and detection output are fuzzified into fuzzy membership functions as shown in Figure 19a–c. The output is shown in Figure 19d. In this example, when the confidence score from the LiDAR is 80 and the score from the camera is 20, by applying the five pre-defined fuzzy rules, the output detection score is around 56.

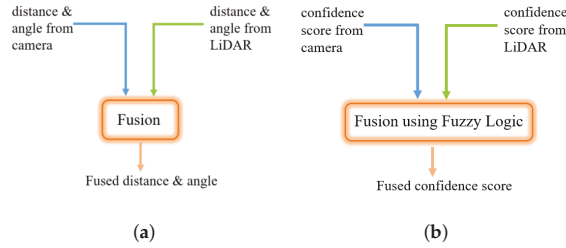


Figure 18. (a) Fusion of distances and angles. (b) Fusion of confidence scores. Fusion of distances, angles and confidence scores from the camera and LiDAR.

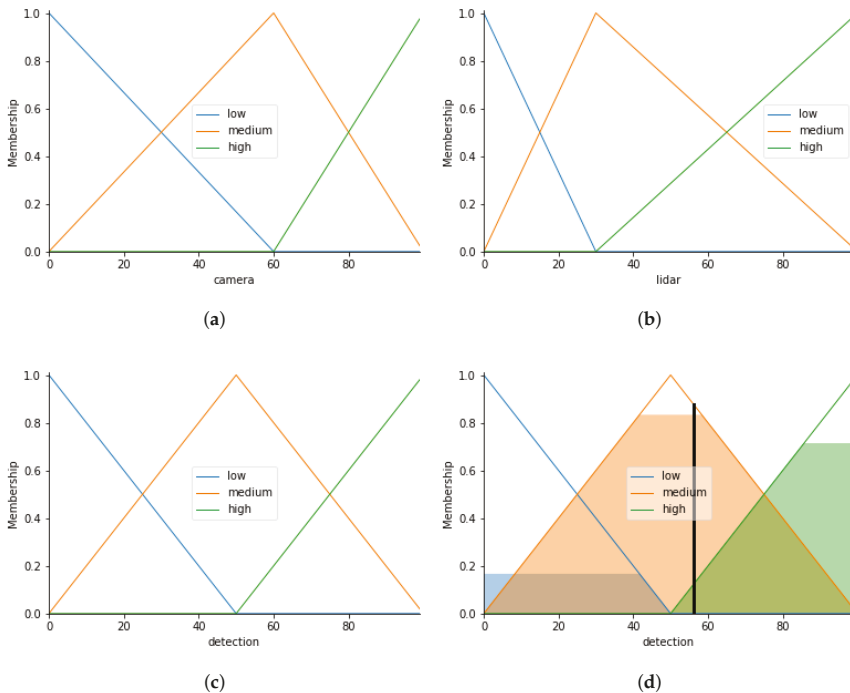


Figure 19. (a) Fuzzy membership function for camera. (b) Fuzzy membership function for LiDAR. (c) Fuzzy membership function for detection. (d) Fuzzy logic output. Combination of confidence scores from the camera and LiDAR using fuzzy logic. Figure best viewed in color.

### 3.8.2. Hyperparameter Optimization

In machine learning, the value of hyperparameter is set before the learning process. In Equation (6),  $\alpha$  controls the LiDAR sigmoid-like squashing function and is one of the hyperparameters to be decided. In Algorithm 4, there is another hyperparameter threshold  $C$  for the final detection. Figure 20 shows where these two hyperparameters affect the fusion process. For the optimization of hyperparameters, some commonly-used approaches include marginal likelihood [51], grid search [52,53], evolutionary optimization [54], and many others. Herein, we utilized grid search. This method is widely used in machine learning system as shown in [52,53]. In grid search, the two hyperparameters are investigated by selecting candidate values for each, then generating a 2D matrix of their cross product.

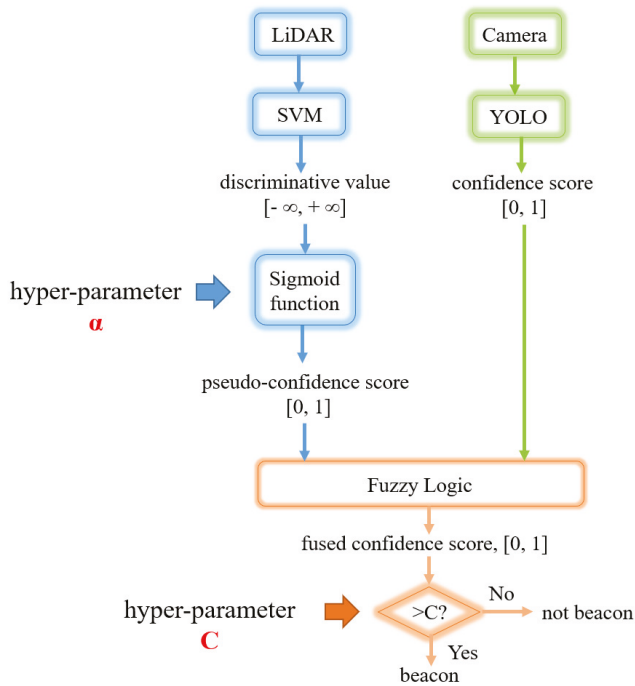


Figure 20. Fusion of confidence scores with two hyperparameters.

As we have two hyperparameters  $\alpha$  and  $C$  to be tuned, we use the Kolmogorov–Smirnov test [55], also known as the KS-test, to determine which hyperparameter values to choose. In our application, the KS-test is to find the  $\alpha$  and  $C$  that maximize the value of TPR minus FPR. As we want to maximize TPR and minimize FPR, the purpose of the KS-test is to find the balance between the two metrics.

The values chosen for  $\alpha$  are  $\frac{1}{100}$ ,  $\frac{1}{500}$ ,  $\frac{1}{1000}$ ,  $\frac{1}{5000}$ ,  $\frac{1}{10,000}$ ,  $\frac{1}{50,000}$ ,  $\frac{1}{100,000}$ ,  $\frac{1}{500,000}$  and  $\frac{1}{1,000,000}$ , while the values for  $C$  are 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9 and 0.95. We use 3641 testing LiDAR and camera pairs, and the resulting heat maps for TPR, FPR and the KS-test results are shown in Figure 21. As we want the TPR to be high, in Figure 21a, the green area shows where the highest TPR is, while the red area is where the lowest TPR is. Furthermore, we want the FPR to be low, in Figure 21b; the lower values are in the green area, while the higher values are in the red area. To balance between TPR and FPR, Figure 21c shows the KS-test results, in other words the difference between TPR and FPR. We want to choose the  $\alpha$  and  $C$  related to higher values (green area) in this heat map. Based on the KS-test, there are three pairs of  $\alpha$  and  $C$  values that can be chosen, and we choose  $\alpha$  to be  $\frac{1}{500,000}$  and  $C$  to be 0.65 in our experiment.

$\alpha$	C	0.5	0.55	0.6	0.65	0.7	0.75	0.8	0.85	0.9	0.95
1/100		1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1/500		1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.986
1/1,000		1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.999	0.966	0.880
1/5,000		1.000	1.000	1.000	1.000	0.999	0.989	0.977	0.944	0.798	0.711
1/10,000		1.000	1.000	1.000	1.000	0.991	0.988	0.966	0.934	0.744	0.711
1/50,000		1.000	1.000	1.000	0.999	0.991	0.985	0.957	0.916	0.710	0.085
1/100,000		1.000	1.000	1.000	0.999	0.991	0.985	0.954	0.883	0.268	0.000
1/500,000		1.000	1.000	1.000	0.999	0.991	0.967	0.848	0.502	0.029	0.000
1/1,000,000		1.000	1.000	1.000	0.999	0.991	0.947	0.833	0.476	0.029	0.000

(a)

$\alpha$	C	0.5	0.55	0.6	0.65	0.7	0.75	0.8	0.85	0.9	0.95
1/100		0.053	0.053	0.053	0.053	0.053	0.053	0.053	0.053	0.052	0.052
1/500		0.053	0.053	0.052	0.052	0.051	0.051	0.051	0.051	0.051	0.048
1/1,000		0.053	0.052	0.051	0.051	0.051	0.051	0.048	0.047	0.045	0.043
1/5,000		0.053	0.051	0.045	0.043	0.042	0.041	0.039	0.036	0.033	0.030
1/10,000		0.053	0.045	0.043	0.040	0.036	0.033	0.032	0.029	0.027	0.020
1/50,000		0.053	0.034	0.027	0.018	0.011	0.006	0.003	0.001	0.000	0.000
1/100,000		0.053	0.027	0.012	0.004	0.002	0.000	0.000	0.000	0.000	0.000
1/500,000		0.053	0.010	0.003	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1/1,000,000		0.053	0.004	0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.000

(b)

$\alpha$	C	0.5	0.55	0.6	0.65	0.7	0.75	0.8	0.85	0.9	0.95
1/100		0.947	0.947	0.947	0.947	0.947	0.947	0.947	0.947	0.948	0.948
1/500		0.947	0.947	0.948	0.948	0.949	0.949	0.949	0.949	0.949	0.938
1/1,000		0.947	0.948	0.949	0.949	0.949	0.949	0.952	0.952	0.921	0.837
1/5,000		0.947	0.949	0.955	0.957	0.957	0.948	0.938	0.908	0.765	0.681
1/10,000		0.947	0.955	0.957	0.960	0.955	0.955	0.934	0.905	0.717	0.691
1/50,000		0.947	0.966	0.973	0.981	0.980	0.979	0.954	0.915	0.710	0.085
1/100,000		0.947	0.973	0.988	0.995	0.989	0.985	0.954	0.883	0.268	0.000
1/500,000		0.947	0.990	0.997	0.999	0.991	0.967	0.848	0.502	0.029	0.000
1/1,000,000		0.947	0.996	0.999	0.999	0.991	0.947	0.833	0.476	0.029	0.000

(c)

Figure 21. (a) True positive rate (TPR) changing with  $\alpha$  and C. (b) False positive rate (FPR) changing with  $\alpha$  and C. (c) KS-test results changing with  $\alpha$  and C. Heat maps demonstrating the TPR, FPR and KS-test results changing with  $\alpha$  and C. Best viewed in color.

#### 4. Experiment

In our experiments, the LiDAR we use is a Quanergy M8-1 LiDAR, shown in Figure 22a. It has eight beams with a vertical spacing of approximately  $3^\circ$ . The camera images are collected by an FLIR Chameleon3 USB camera with a Fujinon 6-mm lens, as shown in Figure 22b.



Figure 22. (a) Quanergy M8-1 LiDAR. (b) FLIR Chameleon3 USB camera with a Fujinon 6-mm lens. The sensors used in our experiments.

#### 4.1. Data Collections Summary

To collect data and test our system in a relevant environment, we needed an open outdoor space without much clutter and interference from random objects. We chose an industrial setting that is very similar to the scenario where we will implement our system. The location has an office building and a large, relatively flat concrete area adjacent to the building. In the environment, there are a few parked cars, a trailer, a couple of gas storage tanks, etc.

Twenty extensive data collections were conducted from August 2017 to March 2018. These tests are designed to collect training data for our system and testing data to check the system's performance and to tune the system parameters. Many tests consisted of beacons placed in a random position in the sensors' FOV. Other tests had beacons in known locations in order to assess the LiDAR and camera accuracies for estimating the range and distance. Later tests used the LiDAR as the "ground truth", since it is highly accurate at estimating range and distance.

Most of the initial tests had the industrial vehicle stationary. Later, we performed tests with the industrial vehicle moving. For the initial tests where we needed to know beacon locations, we marked the different ground distances and angles. The ground point straight below the LiDAR is used as the 0-m distance, and the 0° angle is directly in front of the industrial vehicle. From there, we marked the distance from 3 m at each meter interval up to 40 m across the 0° angle line. The 0° angle is directly in front of the industrial vehicle. We also marked the -20°, -15°, -10°, -5°, 0°, 5°, 10°, 15° and 20° lines for beacon location. In this way, we constructed a dataset of beacons with labeled ground truths. For static beacon testing, we placed the beacon in both known and unknown locations and recorded the data.

For the stationary vehicle testing, we set up several tests to evaluate different parts of the system. For instance, a beacon was placed on a very short wooden dolly attached to a long cord. This allowed the beacon to be moved while the people moving it were not in the sensors' field of view. This allowed the beacon to be placed at a known distance along the 0° angle line and drag toward the vehicle to collect data at continuous locations. We also did similar tests along the -40°, -20°, -10°, 10°, 20° and 40° angle lines. Finally, we collected data where the beacon was drag from the -40° angle to all the way across the 40° angle at roughly fixed distances. These data collections gave us a large amount of data to use in training and testing our system.

We also performed data collections with humans. The retro-reflective stripes on the vests present bright returns to the LiDAR, which may cause false positives. To measure these effects, we had different people wear a highly reflective vest and stand in front of the LiDAR and cameras in a known location. Data were gathered with the person in multiple orientations facing toward and away from the camera. This test was also performed without the reflective safety vests. Other tests had people walking around in random directions inside the sensors' FOV. The data collected here were part of the non-beacon data used to train the SVM.

In order to examine interference when people and beacons were also in the scene, we had people wearing a vest standing close to the beacon in known locations. Then, we recorded the data, which helped the system to distinguish the beacon and people even if in close contact. Furthermore, we also wanted some dynamic cases of beacons and people in close proximity. To gather these data, the beacon was pulled around using the rope and dolly, while a second person moved around the beacon.

There was also a series of data collections with the vehicle driving straight at or just past a beacon or another object, to test that the sensor system was stable under operating conditions. The vehicle was driven at a constant speed, accelerating and braking (to make the vehicle rock) and in a serpentine manner. Finally, we performed many data collections with the vehicle driving around with different obstacles and beacons present. This was mainly to support system integration and tune the MPC controller and actuators. All of these collections provided a rich dataset for system analysis, tuning and performance evaluation.

#### 4.2. Camera Detection Training

Training images for YOLO also include images taken by a Nikon D7000 camera and images from PASCAL VOC dataset [56]. The total number of training images is around 25,000.

The network structure of YOLO is basically the same as the default setting. One difference is the number of filters in the last layer. This number is related to the number of object categories we are trying to detect and the number of base bounding boxes for each of the grid sections into which YOLO divides the image. For example, we can divide each image into 13 by 13 grids. For each grid, YOLO predicts five bounding boxes, and for each bounding box, there are  $5 + N$  parameters ( $N$  represents the number of categories to be predicted). As a result, the last layer filter size is  $13 \times 13 \times 5 \times (5 + N)$ . Another difference is that we change the learning rate to  $10^{-5}$  to avoid divergence.

#### 4.3. LiDAR Detection Training

The LiDAR data were processed through a series of steps. We implemented a ground clutter removal algorithm to keep ground points from giving us false alarms, to avoid high reflections from reflective paints on the ground and to reduce the size of our point cloud. Clusters of points are grouped into distinct objects and are classified using an SVM. The SVM is trained using a large set of data that have beacons and non-beacon objects. The LiDAR data were divided into two disjoint sets: training and testing. In the training dataset, there were 13,190 beacons and 15,209 non-beacons. The testing dataset contained 5666 beacons and 12,084 non-beacons. The confusion matrices for training and testing are shown in Tables 2 and 3. The results show good performance for the training and testing datasets. Both datasets show over 99.7% true positives (beacons) and around 93% true negatives (non-beacons).

**Table 2.** LiDAR training confusion matrix.

	Beacon	Non-Beacon
Beacon	13,158	32
Non-Beacon	610	14,599

**Table 3.** LiDAR testing confusion matrix.

	Beacon	Non-Beacon
Beacon	5653	13
Non-Beacon	302	11,782

## 5. Results and Discussion

### 5.1. Camera Detection Mapping Results

Camera and LiDAR detections generate both different types of data and data on different scales. The LiDAR generates range and angle estimates, while the camera generates bounding boxes. The LiDAR discriminant score can be any real number, while the camera confidence score is a real number in the range  $[0, 1]$ . The data from camera detection are first mapped into the form of distance and range, the same form as the data reported from LiDAR. We can accomplish this conversion fairly accurately because we know the true size of the beacon. The data collected for mapping cover the entire horizontal FOV of the camera, which is  $-20^\circ$  to  $20^\circ$  relative to the front of the industrial vehicle. The distance covered is from 3 to 40 m, which is beyond the maximum detection range of our LiDAR detection algorithm. We collected around 3000 pairs of camera and LiDAR detection for training of the mapping NN and 400 pairs for testing.

From Figure 23, we observe that the larger the  $x$ -coordinate, the larger the angle from the camera, and the larger the size of the bounding box, the nearer the object to the camera. From our training data, we see that the bounding box's  $x$ -coordinate has an approximately linear relationship with the LiDAR detection angle, as shown in Figure 24a. We also observe that the size of the bounding box from the camera detection has a nearly negative exponential relationship with the distance from LiDAR detection as shown in Figure 24b. To test these relationships, we ran least squares fitting algorithms to estimate coefficients for a linear fit between the  $[Xmin, Xmax]$  of the camera bounding box and the LiDAR angle and for an exponential fit between the  $[Width, Height]$  and the LiDAR estimated distance.

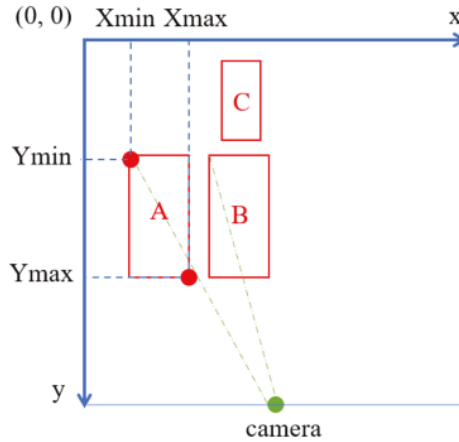


Figure 23. One sample frame with three different bounding boxes.

Herein, we compare the NN and linear regression result methods for angle estimation, and we also compare the results using NN and exponential curve fitting method for distance. The results are shown in Table 4. The table shows the mean squared error (MSE) using both the NN and the linear regression for the angle estimation, as well as the MSE for distance estimation. The table also reports the coefficient of determination, or  $r^2$  scores [57]. The  $r^2$  scores show a high measure of goodness of fit for the NN approximations. We find that the NN provides more accurate predictions on both distance and angle.

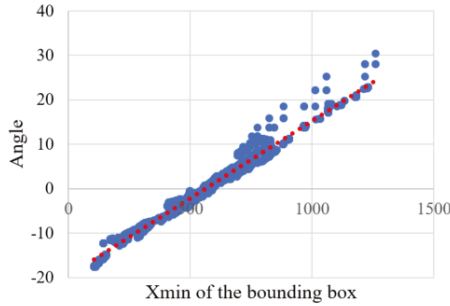
Table 4. Mapping analysis. The best results are shown in bold.

	Angle (Degrees)		Distance (Meters)	
	NN	Linear Regression	NN	Exponential Curve Fitting
Mean Squared Error (MSE)	<b>0.0467</b>	0.0468	<b>0.0251</b>	0.6278
$r^2$ score	<b>0.9989</b>	<b>0.9989</b>	<b>0.9875</b>	0.9687

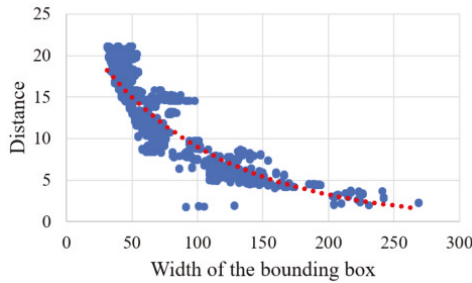
### 5.2. Camera Detection Results

In order to see how well the camera works on its own for reporting distance and angle for its detections, 400 testing images are processed. We compare the mapping results with the LiDAR detection, which is treated as the ground-truth. The results are shown in Figure 25. From Figure 25a, we can see that the camera can fairly accurately predict the angle with a maximum error around  $3^\circ$ , and this only happens near the image edges. On the other hand, the camera's distance prediction has relatively large errors, as shown in Figure 25b. Its maximum error is around 1 m, which happens when the distance is around 6 m, 9 m, 15 m and 18 m.





(a) Bounding box  $X_{min}$  from the camera vs. the angle from LiDAR.  $X_{min}$  units are pixels, and angle units are degrees.



(b) Bounding box  $X_{min}$  from the camera vs. the angle from LiDAR. The bounding box width is pixels, and the distance is meters.

**Figure 24.** Scatter plots of camera/LiDAR training data for fusion. (a) Camera bounding box  $X_{min}$  vs. LiDAR estimated angle. (b) Camera bounding box width vs. LiDAR estimated distance.

### 5.3. Fusion Results

We also compare the true positive detections, false positive detections and false negative detections using only LiDAR data with linear SVM method, only camera image results with YOLO and the proposed fusion method. The results are shown in Tables 5 and 6.

In Tables 5 and 6, TPR, FPR and FNR are calculated by Equations (7a) to (7c), respectively, in which TP is the number of true positives (the number of true beacons detected), TN is the number of true negatives (the number of non-beacons detected correctly), FP is the number of false positive (there is an object detected as a beacon that is not a beacon) and FN is the number of false negatives (there is a beacon, but we have no detection).

$$\text{True Positive Rate (TPR)} = \frac{TP}{TP + FN} \tag{7a}$$

$$\text{False Positive Rate (FPR)} = \frac{FP}{FP + TN} \tag{7b}$$

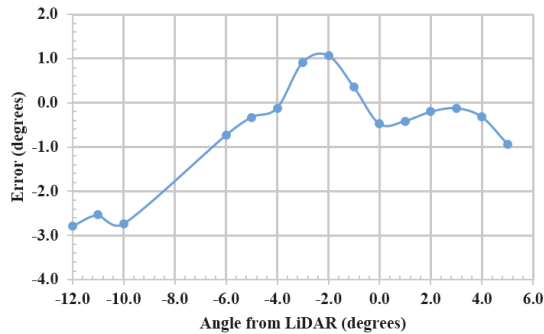
$$\text{False Negative Rate (FNR)} = \frac{FN}{TP + FN} \tag{7c}$$

**Table 5.** Fusion performance: 3 to 20-m range. The best results in **bold**.

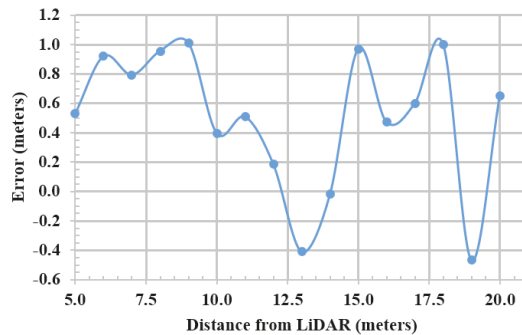
	LiDAR Only	Camera Only	Fusion
True Positive Rate (TPR)	93.90%	<b>97.60%</b>	<b>97.60%</b>
False Positive Rate (FPR)	27.00%	<b>0.00%</b>	6.69%
False Negative Rate (FNR)	6.10%	<b>2.40%</b>	<b>2.40%</b>

**Table 6.** Fusion performance: 20 to 40-m range. The best results in **bold**.

	Camera Only	Fusion
True Positive Rate (TPR)	<b>94.80%</b>	<b>94.80%</b>
False Positive Rate (FPR)	<b>0.00%</b>	<b>0.00%</b>
False Negative Rate (FNR)	<b>5.20%</b>	<b>5.20%</b>



(a)



(b)

**Figure 25.** (a) Angle error of the camera. (b) Distance error of the camera. Camera errors.

From these results, one can see that from the camera, there are more detections within its FOV up to 40 m, while its estimation of position, especially for distance, is not accurate. The camera, however, can fail more quickly in rain or bad weather, so we cannot always rely on the camera in bad weather [58,59]. On the other hand, the LiDAR has accurate position estimation for each point, but the current LiDAR processing cannot reliably detect beyond about 20 m; and its false detection rate

(including false positives and false negatives) is relatively high. After fusion, the results show that the system can detect beacons up to 40 m like the camera can, and the LiDAR can help more accurately estimate the position of these detections, which is important to the control system (which will need different control responses when there is a beacon 20 m away versus the case where the beacon is 5 m away). Figure 26 shows the benefit of using the camera and LiDAR fusion system.

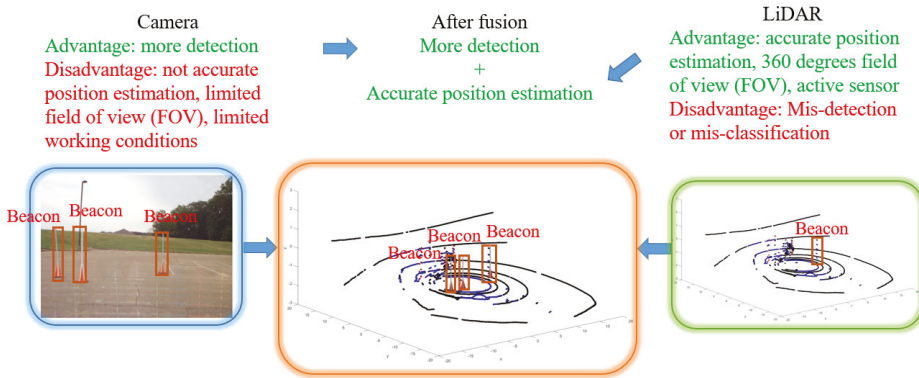


Figure 26. Benefit of using the camera and LiDAR fusion system.

## 6. Conclusions and Future Work

In this paper, we proposed a multi-sensor detection system that fuses camera and LiDAR detections for a more accurate and robust beacon detection system. The proposed system has been designed as a real-time industrial system for collision avoidance and tested with prototypes in various scenarios. The results show that fusion helps to obtain more accurate position and label information for each detection. It also helps to create detection beyond the range of LiDAR while within the range of the camera.

In the future, we will try to implement multiple cameras or use wider FOV cameras to cover a wider FOV. We will also adjust the NN for more accurate prediction. We want to investigate using DL to learn to recognize beacons based on training data, without having to implement hand-crafted features. This approach can provide better results, but will be much more computationally demanding on the Jetson. We experimented in low-light conditions and found that the camera detection system worked well until about 20 min after sunset, where the camera images were very dark and had little contrast. The system was able to detect beacons, but the localization (bounding boxes) were poor. In these cases, we would rely on the LiDAR for detection, which really was not affected by low light. Furthermore, we want to construct an environmentally-controlled testing chamber where we can conduct experiments on camera degradation under known rain rates, amounts of fog and dust and under low-light conditions.

**Author Contributions:** Data curation, Lucas Cagle; Methodology, Pan Wei; Resources, Tasmia Reza and James Gafford; Software, Lucas Cagle; Writing – original draft, Pan Wei and John Ball.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.

2. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
3. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [[CrossRef](#)] [[PubMed](#)]
4. Szegedy, C.; Reed, S.; Erhan, D.; Anguelov, D.; Ioffe, S. Scalable, high-quality object detection. *arXiv* **2014**, arXiv:1412.1441.
5. Dai, J.; Li, Y.; He, K.; Sun, J. R-FCN: Object detection via region-based fully convolutional networks. In Proceedings of the Neural Information Processing Systems 2016, Barcelona, Spain, 5–10 December 2016; pp. 379–387.
6. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 21–37.
7. Dalal, N.; Triggs, B. Histograms of Oriented Gradients for Human Detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
8. Felzenszwalb, P.F.; Girshick, R.B.; McAllester, D.; Ramanan, D. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 1627–1645. [[CrossRef](#)] [[PubMed](#)]
9. Wei, P.; Ball, J.E.; Anderson, D.T.; Harsh, A.; Archibald, C. Measuring Conflict in a Multi-Source Environment as a Normal Measure. In Proceedings of the IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), Cancun, Mexico, 13–16 December 2015; pp. 225–228.
10. Wei, P.; Ball, J.E.; Anderson, D.T. Multi-sensor conflict measurement and information fusion. In *Signal Processing, Sensor/Information Fusion, and Target Recognition XXV*; International Society for Optics and Photonics: Bellingham, WA, USA, 2016; Volume 9842, p. 98420F.
11. Wei, P.; Ball, J.E.; Anderson, D.T. Fusion of an Ensemble of Augmented Image Detectors for Robust Object Detection. *Sensors* **2018**, *18*, 894. [[CrossRef](#)] [[PubMed](#)]
12. Lin, C.H.; Chen, J.Y.; Su, P.L.; Chen, C.H. Eigen-feature analysis of weighted covariance matrices for LiDAR point cloud classification. *ISPRS J. Photogr. Remote Sens.* **2014**, *94*, 70–79. [[CrossRef](#)]
13. Golovinskiy, A.; Kim, V.G.; Funkhouser, T. Shape-Based Recognition of 3D Point Clouds in Urban Environments. In Proceedings of the IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 29 September–2 October 2009; pp. 2154–2161.
14. Maturana, D.; Scherer, S. Voxnet: A 3d Convolutional Neural Network for Real-Time Object Recognition. In Proceedings of the IEEE/RISJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015; pp. 922–928.
15. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3d Shapenets: A Deep Representation for Volumetric Shapes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1912–1920.
16. Gong, X.; Lin, Y.; Liu, J. Extrinsic calibration of a 3D LIDAR and a camera using a trihedron. *Opt. Laser Eng.* **2013**, *51*, 394–401. [[CrossRef](#)]
17. Park, Y.; Yun, S.; Won, C.S.; Cho, K.; Um, K.; Sim, S. Calibration between color camera and 3D LIDAR instruments with a polygonal planar board. *Sensors* **2014**, *14*, 5333–5353. [[CrossRef](#)] [[PubMed](#)]
18. García-Moreno, A.I.; Gonzalez-Barbosa, J.J.; Ornelas-Rodriguez, F.J.; Hurtado-Ramos, J.B.; Primo-Fuentes, M.N. LIDAR and panoramic camera extrinsic calibration approach using a pattern plane. In *Mexican Conference on Pattern Recognition*; Springer: Berlin, Germany, 2013; pp. 104–113.
19. Levinson, J.; Thrun, S. Automatic Online Calibration of Cameras and Lasers. In Proceedings of the Robotics: Science and Systems, Berlin, Germany, 24–28 June 2013.
20. Gong, X.; Lin, Y.; Liu, J. 3D LIDAR-camera extrinsic calibration using an arbitrary trihedron. *Sensors* **2013**, *13*, 1902–1918. [[CrossRef](#)] [[PubMed](#)]
21. Napier, A.; Corke, P.; Newman, P. Cross-Calibration of Push-Broom 2d Lidars and Cameras in Natural Scenes. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany, 6–10 May 2013; pp. 3679–3684.
22. Pandey, G.; McBride, J.R.; Savarese, S.; Eustice, R.M. Automatic extrinsic calibration of vision and lidar by maximizing mutual information. *J. Field Robot.* **2015**, *32*, 696–722. [[CrossRef](#)]

23. Castorena, J.; Kamilov, U.S.; Boufounos, P.T. Autocalibration of LIDAR and Optical Cameras via Edge Alignment. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China, 20–25 March 2016; pp. 2862–2866.
24. Li, J.; He, X.; Li, J. 2D LiDAR and Camera Fusion in 3D Modeling of Indoor Environment. In Proceedings of the IEEE National Aerospace and Electronics Conference (NAECON), Dayton, OH, USA, 15–19 June 2015; pp. 379–383.
25. Zhang, Q.; Pless, R. Extrinsic Calibration of a Camera and Laser Range Finder (Improves Camera Calibration). In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, 28 September–2 October 2004; pp. 2301–2306.
26. Vasconcelos, F.; Barreto, J.P.; Nunes, U. A minimal solution for the extrinsic calibration of a camera and a laser-rangefinder. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 2097–2107. [[CrossRef](#)] [[PubMed](#)]
27. Mastin, A.; Kepner, J.; Fisher, J. Automatic Registration of LIDAR and Optical Images of Urban Scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 2639–2646.
28. Maddern, W.; Newman, P. Real-Time Probabilistic Fusion of Sparse 3D LIDAR and Dense Stereo. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016; pp. 2181–2188.
29. Zadeh, L.A. Fuzzy sets. *Inf. Control* **1965**, *8*, 338–353. [[CrossRef](#)]
30. Ross, T.J. *Fuzzy Logic with Engineering Applications*; John Wiley & Sons: Hoboken, NJ, USA, 2010.
31. Zhao, G.; Xiao, X.; Yuan, J. Fusion of Velodyne and Camera Data for Scene Parsing. In Proceedings of the 15th International Conference on Information Fusion (FUSION), Singapore, 9–12 July 2012; pp. 1172–1179.
32. Liu, J.; Jayakumar, P.; Stein, J.; Ersal, T. A Multi-Stage Optimization Formulation for MPC-based Obstacle Avoidance in Autonomous Vehicles Using a LiDAR Sensor. In Proceedings of the ASME Dynamic Systems and Control Conference, Columbus, OH, USA, 28–30 October 2015.
33. Alrifaae, B.; Maczjowski, J.; Abel, D. Sequential Convex Programming MPC for Dynamic Vehicle Collision Avoidance. In Proceedings of the IEEE Conference on Control TEchnology and Applications, Mauna Lani, HI, USA, 27–30 August 2017.
34. Anderson, S.J.; Peters, S.C.; Pilutti, T.E.; Iagnemma, K. An Optimal-control-based Framework for Trajectory Planning, Thread Assessment, and Semi-Autonomous Control of Passenger Vehicles in Hazard Avoidance Scenarios. *Int. J. Veh. Auton. Syst.* **2010**, *8*, 190–216.
35. Liu, Y.; Davenport, C.; Gafford, J.; Mazzola, M.; Ball, J.; Abdelwahed, S.; Doude, M.; Burch, R. *Development of A Dynamic Modeling Framework to Predict Instantaneous Status of Towing Vehicle Systems*; SAE Technical Paper; SAE International: Warrendale, PA, USA; Troy, MI, USA, 2017.
36. Davenport, C.; Liu, Y.; Pan, H.; Gafford, J.; Abdelwahed, S.; Mazzola, M.; Ball, J.E.; Burch, R.F. A kinematic modeling framework for prediction of instantaneous status of towing vehicle systems. *SAE Int. J. Passeng. Cars Mech. Syst.* **2018**. [[CrossRef](#)]
37. Quigley, M.; Gerkey, B.; Conley, K.; Faust, J.; Foote, T.; Leibs, J.; Berger, E.; Wheeler, R.; Ng, A. ROS: An open-source Robot Operating System. In Proceedings of the IEEE Conference on Robotics and Automation (ICRA) Workshop on Open Source Robotics, Kobe, Japan, 12–17 May 2009.
38. ROS Nodelet. Available online: <http://wiki.ros.org/nodelet> (accessed on 22 April 2018).
39. JETSON TX2 Technical Specifications. Available online: <https://www.nvidia.com/en-us/autonomous-machines/embedded-systems-dev-kits-modules/> (accessed on 5 March 2018).
40. Girshick, R. Fast R-CNN. *arXiv* **2015**, arXiv:1504.08083.
41. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996; pp. 226–231.
42. Wang, L.; Zhang, Y. LiDAR Ground Filtering Algorithm for Urban Areas Using Scan Line Based Segmentation. *arXiv* **2016**, arXiv:1603.00912. [[CrossRef](#)]
43. Meng, X.; Currit, N.; Zhao, K. Ground filtering algorithms for airborne LiDAR data: A review of critical issues. *Remote Sens.* **2010**, *2*, 833–860. [[CrossRef](#)]
44. Rummelhard, L.; Paigwar, A.; Nègre, A.; Laugier, C. Ground estimation and point cloud segmentation using SpatioTemporal Conditional Random Field. In Proceedings of the Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017; pp. 1105–1110.

45. Rashidi, P.; Rastiveis, H. Ground Filtering LiDAR Data Based on Multi-Scale Analysis of Height Difference Threshold. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2017**, *XLII-4/W4*, 225–229. [[CrossRef](#)]
46. Chang, Y.; Habib, A.; Lee, D.; Yom, J. Automatic classification of lidar data into ground and non-ground points. *Int. Arch. Photogr. Remote Sens.* **2008**, *37*, 463–468.
47. Miadlicki, K.; Pajor, M.; Saków, M. Ground plane estimation from sparse LIDAR data for loader crane sensor fusion system. In Proceedings of the 2017 22nd International Conference on Methods and Models in Automation and Robotics (MMAR), Miedzyzdroje, Poland, 28–31 August 2017; pp. 717–722.
48. Lillywhite, K.; Lee, D.J.; Tippetts, B.; Archibald, J. A feature construction method for general object recognition. *Pattern Recognit.* **2013**, *46*, 3300–3314. [[CrossRef](#)]
49. Vapnik, V. *Statistical Learning Theory*; Wiley: New York, NY, USA, 1998.
50. Fan, R.; Chang, K.; Hsieh, C. LIBLINEAR: A library for large linear classification. *J. Mach. Learn. Res.* **2008**, *9*, 1871–1874.
51. Zorzi, M.; Chiuso, A. Sparse plus low rank network identification: A nonparametric approach. *Automatica* **2017**, *76*, 355–366. [[CrossRef](#)]
52. Chicco, D. Ten quick tips for machine learning in computational biology. *BioData Min.* **2017**, *10*, 35. [[CrossRef](#)] [[PubMed](#)]
53. Hsu, C.W.; Chang, C.C.; Lin, C.J. *A Practical Guide to Support Vector Classification*; Technical Report; National Taiwan University: Taiwan, China, 2010.
54. Olson, R.S.; Urbanowicz, R.J.; Andrews, P.C.; Lavender, N.A.; Moore, J.H. Automating biomedical data science through tree-based pipeline optimization. In Proceedings of the European Conference on the Applications of Evolutionary Computation, Porto, Portugal, 30 March–1 April 2016; pp. 123–137.
55. Chakravarty, I.M.; Roy, J.; Laha, R.G. *Handbook of Methods of Applied Statistics*; McGraw-Hill: New York, NY, USA, 1967.
56. Everingham, M.; Van Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [[CrossRef](#)]
57. Draper, N.R.; Smith, H. *Applied Regression Analysis*; John Wiley & Sons: Hoboken, NJ, USA, 2014; Volume 326.
58. Larochelle, V.; Bonnier, D.; Roy, G.; Simard, J.R.; Mathieu, P. Performance assessment of various imaging sensors in fog. In Proceedings of the International Society for Optical Engineering, Aspen, CO, USA, 10–13 May 1998; pp. 66–81.
59. Park, D.; Ko, H. Fog-degraded image restoration using characteristics of RGB channel in single monocular image. In Proceedings of the IEEE International Conference on Consumer Electronics, Las Vegas, NV, USA, 13–16 January 2012; pp. 139–140.



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# Study on Crash Injury Severity Prediction of Autonomous Vehicles for Different Emergency Decisions Based on Support Vector Machine Model

Yaping Liao, Junyou Zhang \*, Shufeng Wang, Sixian Li and Jian Han

College of Transportation, Shandong University of Science and Technology, Huangdao District, Qingdao 266590, China; liaoyapingsk@163.com (Y.L.); shufengwang@sdust.edu.cn (S.W.); gsyysbyw666@163.com (S.L.); hanjianzrx@163.com (J.H.)

\* Correspondence: junyouzhang@sdust.edu.cn; Tel.: +86-139-0532-3314

Received: 24 October 2018; Accepted: 29 November 2018; Published: 3 December 2018

**Abstract:** Motor vehicle crashes remain a leading cause of life and property loss to society. Autonomous vehicles can mitigate the losses by making appropriate emergency decision, and the crash injury severity prediction model is the basis for autonomous vehicles to make decisions in emergency situations. In this paper, based on the support vector machine (SVM) model and NASS/GES crash data, three SVM crash injury severity prediction models (B-SVM, T-SVM, and BT-SVM) corresponding to braking, turning, and braking + turning respectively are established. The vehicle relative speed (REL\_SPEED) and the gross vehicle weight rating (GVWR) are introduced into the impact indicators of the prediction models. Secondly, the ordered logit (OL) and back propagation neural network (BPNN) models are established to validate the accuracy of the SVM models. The results show that the SVM models have the best performance than the other two. Next, the impact of REL\_SPEED and GVWR on injury severity is analyzed quantitatively by the sensitivity analysis, the results demonstrate that the increase of REL\_SPEED and GVWR will make vehicle crash more serious. Finally, the same crash samples under normal road and environmental conditions are input into B-SVM, T-SVM, and BT-SVM respectively, the output results are compared and analyzed. The results show that with other conditions being the same, as the REL\_SPEED increased from the low (0–20 mph) to middle (20–45 mph) and then to the high range (45–75 mph), the best emergency decision with the minimum crash injury severity will gradually transition from braking to turning and then to braking + turning.

**Keywords:** autonomous vehicle; crash injury severity prediction; support vector machine model; emergency decisions; relative speed; total vehicle mass of the front vehicle

## 1. Introduction

Traffic crashes have caused significant loss to society such as life and property loss, traffic congestion, etc. Rear-end crashes are considered to be the most frequently occurring type of traffic crashes in many countries [1]. In the United States, the 2015 crash records showed that rear-end crashes caused by the emergency braking of the frontal vehicles accounted for 27.7% of the total crashes, causing about 13% of serious injuries and fatalities [2]. One potentially important factor in aggravating crash severity is the inappropriate decisions made by drivers due to the inadequate surveillance and risk perception, lack of responsiveness, and misjudgment [3].

### 1.1. Research Status of Autonomous Vehicles

In recent years, in order to reduce crash casualties and alleviate traffic congestion, autonomous vehicles have attracted worldwide attention. The autonomous vehicle is an advanced stage of intelligent vehicle

development. It can comprehensively utilize its ability of perception, decision-making, and manipulation to replace human drivers and independently execute vehicle driving tasks in specific environments. Because road environments and weather conditions are complicated and changeable, the crucial task of realizing autonomous driving is to empower vehicles with a high degree of artificial intelligence, then the autonomous vehicles can make real-time judgment on the driving status and environment changes in all regions and weather conditions so as to ensure vehicle safe driving [4].

At present, the research on autonomous vehicles is mostly focused on safety control [5], and the crash injury severity prediction for decision-making under emergency situation are seldom studied. In this paper, the emergency situation refers to the situation where the crash is unavoidable. Under this situation, the experienced drivers usually need to first observe and judge the current scene, then rely on their own driving experience to weigh the risk of the crash target, and make an emergency decision with the minimum crash injury severity, such as braking, turning, or braking + turning. Due to the fact that drivers' driving experience is characterized by the driving data of regular vehicles, in order to adapt to complex traffic environment and reduce crash casualties, autonomous vehicles must be trained by these data based on machine learning to establish the crash injury severity prediction model corresponding to each emergency decision. In this way, autonomous vehicles can weigh the hazards of various emergency decisions in advance, like drivers, in order to make emergency decisions with the minimum injury severity [6].

## 1.2. Research Status of Crash Injury Severity Prediction Model

### 1.2.1. Statistical Models

At present, most studies used driving characteristics (age, gender), accident characteristics (time, date, weather, light conditions), vehicle characteristics (vehicle type, limit speed), road characteristics (curvature, slope, road adhesion coefficient, number of lanes), traffic control equipment, etc. as the impact indicators of crash injury severity, and based on the statistical models to model crash injury severity prediction [7–12]. Traditionally, the most used basic statistical models are binary logit/probit models [9,10] and multinomial logit/probit model [11,12]. In order to improve the predictive performance of these models, some studies considered the ordinal nature of injury severity variables and used the ordered logit/probit models instead of traditional models [13–15]. There are also some studies that take the heterogeneity and intrinsic relevance of the crash data into account, and develop more advanced statistical models. For example, Lee et al. [16] considered the heterogeneity of explanatory variables, and developed the heteroscedastic ordered logit (HOL) model for severity analysis of single- and multi-vehicle crash. Shaheed et al. [17] considered the intrinsic relevance in injury severity, and developed fully Bayesian hierarchical multinomial logit (BHML) model to correctly analyze the factors affecting occupant injury severity in winter seasons. Most of the above models assumed that the crash data had a certain distribution, and used the linear function to fit the relationship between dependent and explanatory variables. However, a common shortcoming of the above statistical models is that once the assumption is overturned, it will make an error in the estimation and prediction.

### 1.2.2. Machine Learning Model and Data Mining Techniques

In order to overcome the shortcoming of statistical models, some studies applied machine learning algorithms and data mining techniques to predict crash injury severity. For example, Kashani et al. [18] used classification and regression trees (CART) to analyze traffic crash data of the main two-lane, two-way rural roads of Iran, the results showed that CART could easily find important variables and improve the prediction accuracy of the fatality. Mujalli et al. [19] proposed a new Bayesian networks (BN) model to model the crash injury severity prediction, the results showed that the proposed BN model could simplify the prediction structure of the model without reducing the prediction performance. Zeng et al. [20] used the neural network model to predict crash injury severity combining with a convex combination (CC) algorithm, the results showed that the fitting and forecasting effect



of the neural network model were better than that of OL model. Delen et al. [21] analyzed and compared the prediction performance of artificial neural network (ANN), support vector machine (SVM), decision trees (DT), and logistic regression models on crash injury severity using the crash data of the national automotive sampling system/general estimation system (NASS/GES), the results showed that SVM model had the best prediction performance with the highest accuracy.

### 1.3. Impact of Relative Speed and Vehicle Weight on the Crash Injury Severity

Although the above studies have analyzed the important impact indicators and their influencing mechanisms on the crash injury severity, the relative speed between two crash vehicles (REL\_SPEED) and the gross vehicle weight rating (GVWR) on crash injury severity have been seldom taken into account. Some empirical studies have shown that the REL\_SPEED and GVWR have a great impact on the crash injury severity [22–25]. Tolouei et al. [22] analyzed the relationship between the crash injury severity and the mass ratio of crash vehicles by using the relevant theory of crash mechanics, the results showed that under certain conditions, the increase of vehicle mass would make crash accidents more serious. Jurewicz et al. [23] analyzed the relationship between the relative speed of two crash vehicles and the crash injury severity using the theory of crash dynamics, the results showed that in rear end crashes, the increase of the relative speed would increase the serious injury or death rate. The National Transportation Safety Board developed the range of vehicle speed variation for each crash injury severity level using a logical parameter of assessing the probability of injury [24], and Xu [25] used the conservation law of kinetic energy to establish the relationship between the relative speed of two vehicles before the crash and the speed variation. Under certain emergency scenarios, combined with these two studies [24,25], the correlation between the relative speed and the crash injury severity would be obtained indirectly. It can be seen that the REL\_SPEED and GVWR play important roles in the crash injury severity prediction, so they need to be analyzed in depth in the process of establishing a prediction model.

### 1.4. Objectives of This Study

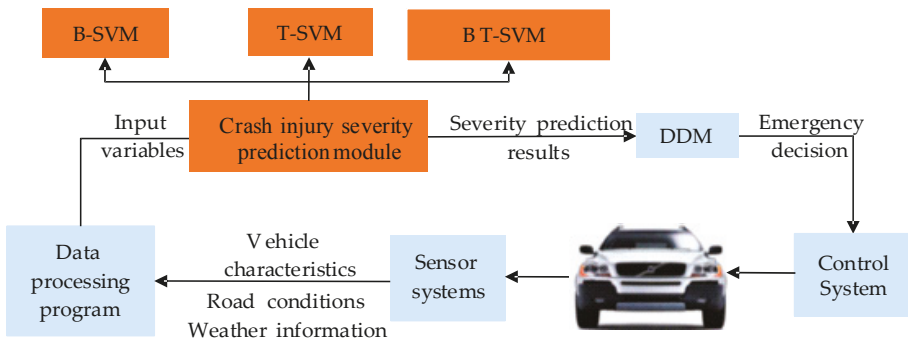
The present study intends to take the emergency situation of two-vehicle crash caused by emergency braking of the frontal vehicle as the research object, and studies the crash injury severity under three emergency decisions based on the NASS/GES data, including braking, turning, and braking + turning. By introducing the REL\_SPEED and the GVWR into the impact indicators of crash injury severity, the support vector machine (SVM) is used to establish crash injury severity prediction models for autonomous vehicles. Specifically, this study makes the following contributions:

- (1) A detailed description of the emergency decision-making process for autonomous vehicles under emergency situation is conducted;
- (2) Based on the NASS/GES crash sample data and SVM model, the braking-SVM (B-SVM), turning-SVM (T-SVM), and braking + turning-SVM (BT-SVM) injury severity prediction model corresponding to braking, turning, and braking + turning are established for autonomous vehicles through the parameter optimization and kernel function selection process of particle swarm optimization (PSO). Then the ordered logit (OL) and back propagation neural network (BPNN) models are established to verify the efficiency of SVM in prediction accuracy;
- (3) Based on the B-SVM, T-SVM, and BT-SVM model, a sensitivity analysis is conducted to quantify the impact of REL\_SPEED and GVWR on the crash injury severity;
- (4) Based on the same crash sample, statistically analyze and compare the ratios of crash injury severity output from B-SVM, T-SVM, and BT-SVM, and provide reference of emergency decision-making for autonomous vehicles in emergencies;
- (5) The research contents and conclusions are summarized, and the future research work is prospected.

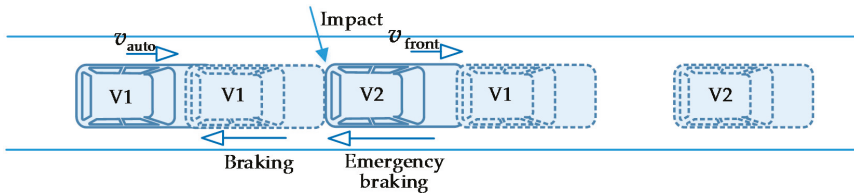
## 2. Emergency Decision-Making Process of Autonomous Vehicles under Emergency Situations

The emergency situations studied in this paper refer to the situations where the two-vehicle crash cannot be avoided when the frontal vehicle is in emergency braking, and we assumed that there are no conflicting vehicles in the adjacent lanes. The emergency decision-making process of autonomous vehicles under emergency situations is shown in Figure 1. Firstly, the autonomous vehicle collects the environment information using the sensor device, and inputs them into the data processing program for information selection, fusion and extraction, to obtain the input variables required by the B-SVM, T-SVM, and BT-SVM models. Then, the three models output their corresponding crash injury severity for decision-making mechanism (DDM), by comparing them, DDM outputs the emergency decision instruction with the minimum injury severity to the control system for operate the corresponding actions. The scenes of vehicle braking, turning, and braking + turning in emergency situation are showed in Figure 2.

During the whole process of information collection, transmission, processing, and execution, the crash injury severity prediction module plays an important role. It is a thinking and analysis device of the autonomous vehicles' central system and is related to the loss of life and property caused by traffic accidents. Therefore, the crash injury severity prediction model is an urgent need to ensure autonomous vehicles to run on the road safely. The following will use the historical accident data to model the crash injury severity prediction for different emergency decisions.

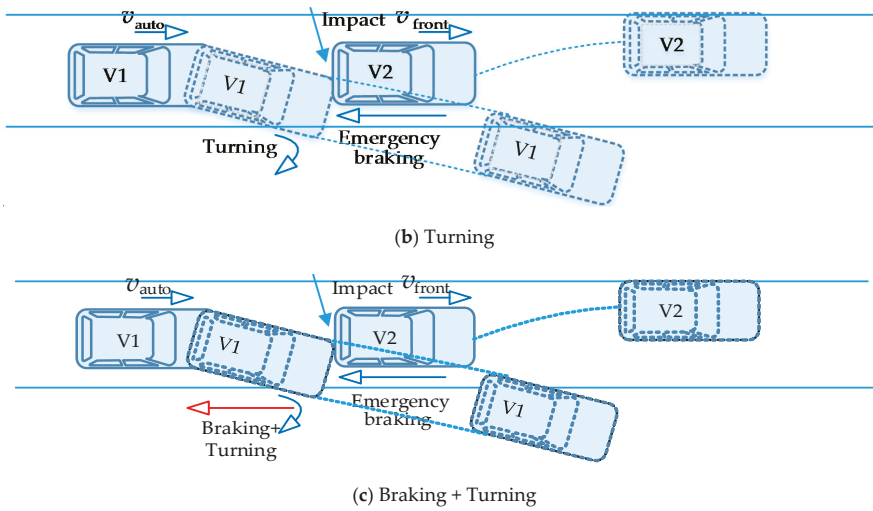


**Figure 1.** Emergency decision-making process of autonomous vehicles under emergency situation. (B-SVM, T-SVM, and BT-SVM refer to the SVM crash injury severity prediction model corresponding to braking, turning, and braking + turning, respectively, i.e., braking-support vector machine, turning-support vector machine, and braking + turning-support vector machine).



(a) Braking

Figure 2. Cont.



**Figure 2.** The crash scenarios of three emergency decisions under the emergency situation ( $v_{auto}$  refers to the speed of autonomous vehicle before making emergency decisions;  $v_{front}$  refers to the speed of the front vehicle; V1 is the autonomous vehicle, and V2 is the emergency braking vehicle).

### 3. Data Preparation

#### 3.1. Crash Data Description

In this study, the vehicle crash data come from the 2012–2015 GES data set in the NASS database system established by the Transportation Department of the United States [2], which has a certain authority, and is a nationally representative sample of police reported motor vehicle crashes of all types, from minor to fatal. The system was created to identify traffic safety problems area, estimate how many motor vehicle crashes of different kinds take place, and what happens when they occur. In addition, the quality of these accident data are further checked through computer processing and data supervisors, they are often used to answer motor vehicle safety questions from congress, lawyers, doctors, students, researchers, and the general public. For the sufficient reliability of GES data set, many experts in the field of vehicle active safety used them for vehicle crash analysis and prediction [21,26].

The data set is mainly composed of three sub-data sets, namely accident, vehicle, and crash participant data set [21]. The accident data set includes road conditions, environmental conditions and other accident-related features. The vehicle data set includes a large number of characteristic variables of the involved vehicles—such as the vehicle speed before the crash, the attempted avoidance decision, etc.—and the crash participant data set includes a large number of characteristic variables of the participation members (drivers, passengers, pedestrians, cyclists, etc.), such as crash injury severity, age, and gender. Each crash record contained in the three sub-data sets is matched by a uniform crash number (CASENUM). The maximum injury severity involved in a crash is the dependent variable of the prediction models in this study. In the GES data set, the maximum injury severity is divided into five types: no apparent injury, possible injury, suspected minor injury, suspected serious injury and fatal.

#### 3.2. Data Processing

##### 3.2.1. Data Screening

(1) This paper only studies the crash mechanism of two vehicles in the same lane in urban road environment, and the autonomous vehicle is a standard passenger vehicle. Therefore, from the GES

data set, we only extract the two-vehicle crash data of the urban road environment caused by the emergency braking made by the frontal vehicle, and the rear vehicle is a standard passenger vehicle.

(2) In order to ensure the accuracy of the later modeling, crash records with missing or wrong data will be eliminated to obtain complete and accurate samples.

(3) Removing some variables in the GES data set that are independent of the crash injury severity, such as driver’s zip code (DR\_ZIP), vehicle identification number (VIN).

(4) In this paper, the autonomous vehicle has no driver and obeys the traffic rules, so we remove the crash records caused by driver’s faults in the rear vehicle, such as drunken driving, drug driving, violating traffic rules, etc.

After the above data screening, a total of 15,164 crash records are obtained, of which 58.7% correspond to emergency braking decision, 18.4% to turning, and 22.9% to braking + turning. In addition, 14 variables are selected as the impact indicators of the crash injury severity prediction models, among them, REL\_SPEED refers to the relative speed of vehicles at the moment before the crash, their related descriptions are showed in Table 1.

### 3.2.2. Input and Output Variables of the Crash Injury Severity Prediction Models

#### (1) Input Variables

As seen in Table 1, there are 14 impact indicators for each sample in this paper. Excessive impact indicator characteristics will affect the search of impact rules, and sometimes several impact indicators can only reflect certain aspect characteristics of the data, which can easily cause a high degree of overlap between indicator characteristics and then cause data analysis obstacles. In order to reduce the computational complexity and improve forecasting efficiency of the crash injury severity prediction models, the Principal Component Analysis (PCA) method is used to reduce the dimensions of the impact indicators, so as to obtain a small amount of irrelevant new indicators on the basis of ensuring the integrity of the original information as much as possible [27,28]. The new indicators are used as the input variables to the prediction models.

**Table 1.** Description of the related variables of the crash injury severity.

Variable Coding	Symbol	Description	Data Type	Descriptive Statistics *
REL_SPEED	$l_1$	Relative speed of the two vehicles/mph	Numeric	36.72 (15.55)
GVWR	$l_2$	Gross vehicle weight rating	Nominal	10,000 lbs or Less (Low range): 0/64.0%; 10,001 lbs–26,000 lbs (Middle range): 1/22.3%; 26,001 lbs or More (High range): 2/13.7%
BODY_TYP	$l_3$	Vehicle type of the frontal vehicle	Nominal	Standard passenger car: 0/77.9%; Bus: 1/14.6%; Motorcycle: 2/5%; Medium/heavy truck: 3/2.5%
SPEEDREL	$l_4$	Speeding (The frontal vehicle)	Binary	No: 0/73.4%; Yes: 1/26.6%;
DAY_WEEK	$l_5$	Crash occurrence date	Nominal	Working day: 0/75.3%; Off day: 1/24.7%
HOUR_IM	$l_6$	Crash time	Nominal	6:00 a.m.–9:59 a.m.: 0/12.7%; 10:00 a.m.–2:59 p.m.: 1/22.1%; 3:00 p.m.–5:59 p.m.: 2/19.8%; 6:00 p.m.–8:59 p.m.: 3/20.9%; 9:00 p.m.–5:59 a.m.: 4/24.5%
LGTCN_IM	$l_7$	Light condition	Nominal	Daylight: 0/53.8%; Dawn or Dusk: 1/3.8%; Dark-lighted: 2/23.1%; Dark-unlighted: 3/19.3%

Table 1. Cont.

Variable Coding	Symbol	Description	Data Type	Descriptive Statistics *
WEATHR_IM	$l_8$	Weather condition	Nominal	Clear/Cloudy: 0/73.2%; Fog: 1/8.7%; Rain: 2/17.0%; Snow: 3/0.8%; Wind: 4/0.3%
VTRAFWAY	$l_9$	Lane type	Nominal	One-Way: 0/7.0%; Two-Way, Not Divided: 1/76.6%; Two-Way, Divided: 2/16.4%
VNUM_LAN	$l_{10}$	Number of lane(s)	Numeric	3.07 (1.32)
VALIGN	$l_{11}$	Curvature of lane(s)	Nominal	Straight: 0/81.7%; Curve: 1/18.3%
VPROFILE	$l_{12}$	Slope of lane(s)	Nominal	Level: 0/93.1%; Grade: 1/3.7%; Hillcrest: 2/2.2%
VSURCOND	$l_{13}$	Road surface conditions	Nominal	Dry: 0/81.4%; Wet: 1/15.4%; others: 2/3.2%
VTRAFCON	$l_{14}$	Traffic control equipment	Nominal	Regulatory Sign: 0/10%; Traffic Signals: 1/42.5%; No Controls: 2/46.4%; Warning signs: 3/1.1%

\* Numeric: mean (St. Dev.); Binary and Nominal: assignment/frequency percentage (%).

We randomly selected 1000 groups of samples for PCA, and the calculated results of PCA for each principal component are shown in Table 2. The cumulative variance contribution rate of the first six principal components has reached 89.7% (>85%), and their eigenvalues is greater than one. Therefore, we use the first six principal components instead of the original indicators for each sample.

Table 2. Eigenvalues and contribution rates.

Principal Component	Eigenvalue	Variance Contribution Rate	Cumulative
1	10.375	0.494	0.494
2	2.814	0.134	0.628
3	2.037	0.097	0.725
4	1.386	0.066	0.791
5	1.197	0.057	0.848
6	1.029	0.049	0.897
7	0.840	0.040	0.937
8	0.273	0.013	0.950
9	0.210	0.010	0.960
10	0.231	0.011	0.971
11	0.168	0.008	0.979
12	0.168	0.008	0.987
13	0.147	0.007	0.994
14	0.126	0.006	1.000

## (2) Output Variables

In the crash data samples obtained above, the samples corresponding to no apparent injury, possible injury, suspected minor injury, suspected serious injury and fatal, account for 54.2, 16.6, 16.8, 7.5, and 4.9% respectively. To balance the proportion of each injury severity, we classify them into three groups as the output variables of the prediction models, namely, no injury (no apparent injury), non-incapacitating injury (slight injury, suspected minor injury), incapacitating/fatal (suspended serious injury, fatal), as seen in Table 3.

Table 3. Output variables of the crash injury severity.

Crash Injury Severity	Description	Represented Value	Statistics
No injury	no apparent injury	−1	54.2%
Non-incapacitating	possible injury	0	33.4%
	suspected minor injury		
Incapacitating/fatal	suspected serious injury	1	12.4%
	fatal		

### 4. Methodology

#### 4.1. Support Vector Machine Model

Support vector machine (SVM) is a machine learning method based on statistical learning theory, which was put forward by C. Cortes and H. Drucker [29]. It has the characteristics of strong learning ability for small samples and good model generalization performance [30]. In recent years, SVM has achieved a breakthrough in theoretical research and algorithm implementation, and has been successfully applied to classification, function approximation, and time series prediction.

SVM was first used to solve the binary classification problem of linear discrete data. As shown in Figure 3, the basic principle is to find an optimal hyperplane that satisfies the data classification requirements, and obtain the maximum margin between two sample points while ensuring the classification accuracy.

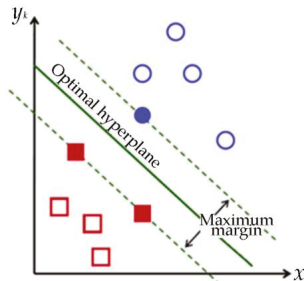


Figure 3. Concept of optimal hyperplane.

In the case of linear classification, suppose the training sample is  $SV = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$ ,  $x \in R^d$ ,  $y_k \in \{-1, 1\}$ ,  $k = 1, 2, \dots, m$ , among which  $x_k$  is the input variable,  $y_k$  represent the crash injury severity,  $m$  is the number of training samples, and  $R^d$  is a  $d$ -dimensional real number space.

SVM linear classification means that there exists a hyperplane  $\omega \cdot x + b = 0$  that can correctly classify all samples, where  $\omega$  is an adjustable weight vector and  $b$  is the bias. The hyperplane needs to meet

$$y_k(\omega \cdot x_k + b) \geq 1, k = 1, 2, \dots, m \tag{1}$$

Calculate the classification interval as

$$\min_{\{x_k|y_k=1\}} \frac{\omega \cdot x_k + b}{\|\omega\|} - \max_{\{x_k|y_k=-1\}} \frac{\omega \cdot x_k + b}{\|\omega\|} = \frac{2}{\|\omega\|} \tag{2}$$

The optimal hyperplane requires that the classification interval be maximized, that is, requires that the  $\|\omega\|$  is minimized, then the optimal hyperplane problem can be expressed as a minimum function that satisfies the constraint of (1)

$$\varphi(\omega) = \frac{1}{2} \|\omega\|^2 = \frac{1}{2} (\omega \cdot \omega) \tag{3}$$

Using the Lagrangian duality transformation and choosing the appropriate penalty function  $c$ , the optimal hyperplane problem can be transformed into solving the maximum of the quadratic programming problem, i.e.,

$$\omega(a) = \sum_{k=1}^m a_k - \frac{1}{2} \sum_{k,l=1}^m a_k a_l y_k y_l (x_k \cdot x_l)$$

$$s.t. \begin{cases} y_k [(\omega \cdot x_k) + b] - 1 \geq 0 \\ \sum_{k=1}^m y_k a_k = 0 \\ 0 \leq a_k \leq c, k = 1, \dots, m \end{cases} \tag{4}$$

where  $a_k$  is the lagrange coefficient.

Finally, the optimal classification function in the case of linear classification can be obtained

$$f(x) = \text{sgn}\{\omega^* \cdot x + b^*\} = \text{sgn}\left\{\sum_{k=1}^m a_k^* y_k (x_k \cdot x) + b^*\right\} \tag{5}$$

where  $a_k^*$ ,  $b^*$  are the parameters to determine the optimal hyperplane,  $(x_k \cdot x)$  is the dot product of two vectors.

When dealing with non-linear problems, it is necessary to map the input variables into the high-dimensional space and construct the optimal hyperplane in the high-dimensional space. At this time, we need to select appropriate kernel functions  $K(x, x_k)$  to achieve linear classification of nonlinear problems. Then, the objective function of the optimal hyperplane problem becomes

$$\omega(a) = \sum_{k=1}^m a_k - \frac{1}{2} \sum_{k,l=1}^m a_k a_l y_k y_l K(x_k, x_l) \tag{6}$$

Then get the optimal classification function in the case of nonlinear classification

$$f(x) = \text{sgn}\left\{\sum_{k=1}^m a_k^* y_k K(x, x_k) + b^*\right\} \tag{7}$$

There are three types of commonly used kernel functions:

- (1) Polynomial kernel function:  $K(x, x_k) = (\gamma(x \cdot x_k) + 1)^q$
- (2) Radial basis kernel (RBF) function:  $K(x, x_k) = \exp\left(-\frac{\|x - x_k\|^2}{\sigma^2}\right)$
- (3) Sigmoid kernel function:  $K(x, x_k) = \tan(v(x \cdot x_k) + c)$

The crash injury severity prediction model constructed in this paper is a three-classification problem (no injury, non-incapacitating, incapacitating/fatal), i.e.,  $y_k \in \{-1, 0, 1\}$ . Since the above traditional SVM model only considers the problem of binary classification, so the SVM model needs to be extended to build the multiple SVM classifiers. The multi-classifier construction of Libsvm (Libsvm is a fast and effective SVM software package) is implemented by combining multiple binary classifiers [31]. In the training, the three types of crash injury severity are classified into three binary combinations—including no injury and non-incapacitating, no injury and incapacitating/fatal, non-incapacitating and incapacitating/fatal—as shown in Figure 4. After training, three binary-class SVM training models are obtained. In the testing, these three SVM training models are used to classify each sample, and then the category with the largest number of classification results is selected as the sample category.

Since the purpose of predicting crash severity for autonomous vehicles is to make the emergency decision with the minimum injury severity, it is necessary to establish crash injury severity prediction models corresponding to different decisions (braking, turning, and braking + turning), so as to provide a comparative basis for autonomous vehicles in emergency situations. The three models are referred as B-SVM, T-SVM, and BT-SVM, respectively.

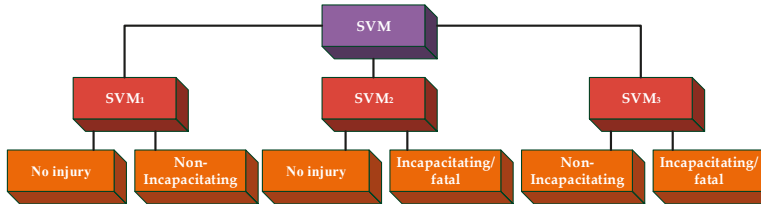


Figure 4. Multi-classifier construction of Libsvm.

In this paper, the sample set corresponding to each emergency decision is selected from the above crash records, and 70% samples are randomly selected from each sample set for training, and the remaining 30% for testing. In order to obtain the optimal B-SVM, T-SVM, and BT-SVM, the B-SVM, T-SVM, and BT-SVM with different kernel functions are trained and compared using the corresponding training sample set.

4.2. Process of Parameter Optimization and Kernel Function Selection

After input the training samples into the SVM models, we use the particle swarm optimization (PSO) algorithm to search for the optimal parameters. PSO is a parallel algorithm, which has been applied in many fields for its advantages of ease of execution, high accuracy, and fast convergence [32]. Its research method is to assume that the particles in the particle swarm are all individuals with no mass and volume. Relying on the independent particle’s adaptability and learning ability to the environment, the flight results of the individual particle and particle swarm are combined to adjust the particle’s own position and flight speed to achieve optimal search [33]. The optimal parameter selection process of SVMs based on the PSO algorithm is shown in the Figure 5 below. The classification accuracy is taken as the fitness function and the main parameters of PSO are set as follows: the population size is  $N = 50$ , the inertial weight is  $\mu = 0.9$ , the particle acceleration constant is  $C_1 = 1.4$ ,  $C_2 = 1.6$ , and the number of iterations is 500.

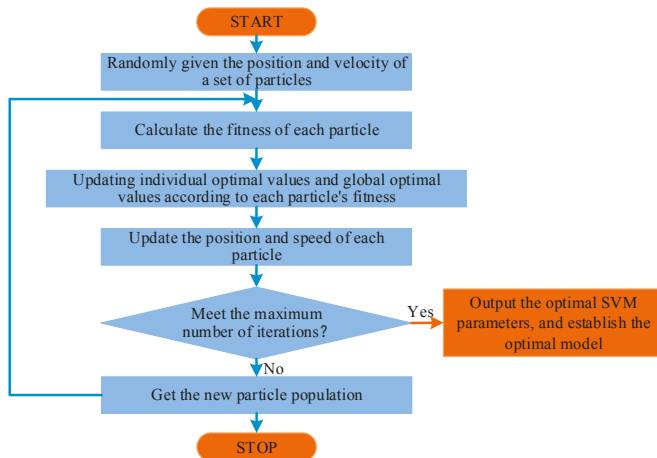


Figure 5. Optimal parameter selection of SVM based on the PSO algorithm.



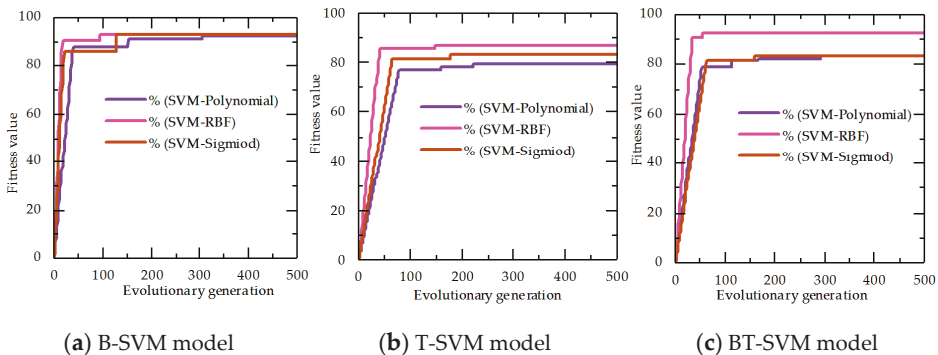
## 5. Results and Discussion

### 5.1. Estimation of SVM Crash Injury Severity Prediction Models

After several iterations, the parameter optimization results of B-SVM, T-SVM, and BT-SVM are obtained, as shown in the following Figure 6 and Table 4. It can be seen from the Figure 4 that the training accuracy of using RBF kernel function is the highest in the B-SVM, T-SVM, and BT-SVM, which are 93.176, 87.111, and 88.442% respectively, followed by the sigmoid kernel function with the training accuracy for 92.894, 83.338, and 83.225% respectively, and the polynomial kernel function with the training accuracy for 92.296, 79.275, and 83.113% respectively. The iterating numbers of the SVM models with RBF kernel function to achieve the highest training accuracy are less than that of the other kernel functions, which are 94, 148, and 132 for B-SVM, T-SVM, and BT-SVM respectively. The results show that the SVM models with RBF kernel function have the best performance on classification accuracy, which can be explained by the fact that the number of coefficients of RBF kernel function is less, and it is not limited by the spatial dimension and sample size.

**Table 4.** Training classification accuracy of SVMs/% with different kernel functions.

Kernel \ Model	B-SVM	T-SVM	BT-SVM
Polynomial	92.296	79.275	83.113
RBF	93.176	87.111	88.442
Sigmoid	92.894	83.338	83.225



**Figure 6.** Parameter optimization of three SVM models with different kernel functions.

Therefore, the RBF kernel function is selected as the kernel function of B-SVM, T-SVM, and BT-SVM model. The optimal parameters obtained by the training are shown in the Table 5.

**Table 5.** Optimal parameters of SVMs with RBF kernel functions.

Model \ Parameters	$c$	$\sigma$
B-SVM	3.7413	0.4857
T-SVM	21.0744	0.0277
BT-SVM	8.1121	0.1854

### 5.2. Performance of SVM Models

This paper trains the most widely used statistical algorithm, OL algorithm, and the machine learning algorithm, BPNN with the same training samples to establish OL and BPNN crash injury

severity prediction models, then compares their prediction accuracy with the above established SVM models to verify the superior performance of SVM in the crash injury severity prediction.

5.2.1. Establishment of OL Models and BPNN Models

(1) Establishment of OL Models

OL model is extended by the binary logit model, which provides a common and convenient framework for analyzing such data in which the dependent variable is both discrete and ordered. OL models only carry out regression analysis for significant variables. In this paper, we use the combined stepwise (CS) to research the significant variables affecting the injury severity for OL model. Based on all the candidate variables, CS removes the independent variables step by step which does not meet the required significant level, and then outputs the significant variables for the OL model. We make the significance level 0.05, when  $p < 0.05$ , the independent variable is retained. The OL prediction models corresponding to braking, turning, and braking + turning decision are recorded as B-OL, T-OL, and BT-OL, respectively. The parameter estimation results of each OL model are shown in the Table 6 below.

Table 6. Parameter estimation based on the OL model.

Variable	B-OL Model		T-OL Model		BT-OL Model	
	Estimate	p Value	Estimate	p Value	Estimate	p Value
REL_SPEED	0.032	0.016	0.029	0.02	0.017	0.019
GVWR	1.215	0.007	1.141	0.017	1.472	0.008
LGTCON_IM	0.175	0.000	–	–	0.277	0.000
WEATHR_IM	0.407	0.000	–	–	–	–
VNUM_LAN	–	–	−0.07	0.000	−0.251	0.001
VALIGN	–	–	0.286	0.000	0.332	0.003
VPROFILE	–	–	1.132	0.001	1.241	0.000
VSURCOND	0.139	0.007	0.802	0.003	0.998	0.015
Cutoff point 1	2.944	–	3.872	–	5.405	–
Cutoff point 2	4.503	–	6.743	–	9.277	–

(2) Establishment of BPNN Models

BPNN is one of the most widely used learning algorithm, which is a multilayer feed forward neural network trained in accordance with the error back propagation algorithm. In order to full verify the performance of SVM model, the BPNN is established to compare SVM model on the performance of predicting the crash injury severity in this paper. Three BPNN models corresponding to braking, turning, and braking + turning decision are recorded as B-BPNN, T-BPNN, and BT-BPNN, which consist of an input layer, five hidden layers, and an output layer, as seen in Figure 7. The six principal components obtained above are the parameters of input layer, and the corresponding crash injury severity is set as the parameter of output layer. The number of nodes in the hidden layers is determined through checking the prediction accuracy of BPNN with different number nodes.  $h_n^{l-1}$  refers to the output of the  $n$ th node in the  $(l - 1)$ th hidden layer. After setting up the relevant parameters, input the same 75% training samples into three BPNN models respectively for iterative training. When the iterative numbers of B-BPNN, T-BPNN, and BT-BPNN model arrived at 84, 152, 111, respectively, the error values of the three neural networks all converge to the target values, as seen in Figure 8. Then B-BPNN, T-BPNN, and BT-BPNN prediction model are obtained.

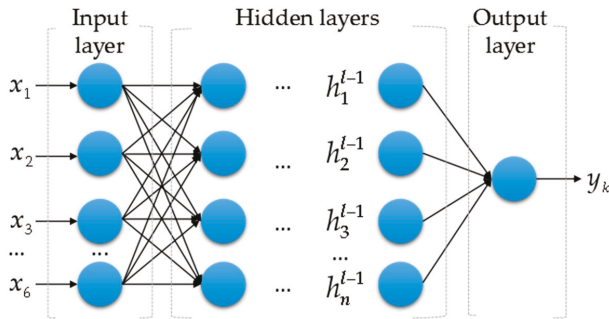


Figure 7. The structure of BPNN.

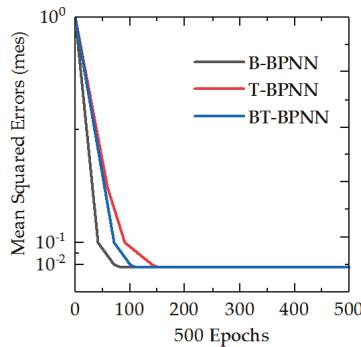


Figure 8. Iterative results of three BPNNs.

5.2.2. Performance Comparison of SVM, OL, and BPNN Models

We input the 25% test samples into the SVM models with RBF kernel function, OL and BPNN models to verify the performance of SVM model on the prediction accuracy. The prediction accuracy is the ratio of the accurate classification number in the samples. The prediction accuracy of the SVM, OL, and BPNN models is shown in the Tables 7–9 below.

Table 7. Prediction accuracy of SVM model in the training and testing.

Crash Injury Severity	Training (%)			Testing (%)		
	B-SVM	T-SVM	BT-SVM	B-SVM	T-SVM	BT-SVM
No injury	94.126	92.107	92.573	91.514	88.903	89.323
Non-incapacitating	91.355	86.642	86.339	89.039	82.871	84.680
Incapacitating/fatal	87.417	84.408	85.015	85.977	81.072	82.983
Overall	93.176	87.111	88.442	88.001	84.712	85.229

Table 8. Prediction accuracy of OL model in the training and testing.

Crash Injury Severity	Training (%)			Testing (%)		
	B-OL	T-OL	BT-OL	B-OL	T-OL	BT-OL
No injury	81.787	79.475	82.127	76.522	73.442	74.627
Non-incapacitating	77.086	71.618	73.503	71.553	62.785	69.320
Incapacitating/fatal	65.296	59.692	61.551	59.352	57.632	59.468
Overall	76.883	69.261	71.727	71.424	65.384	68.488

**Table 9.** Prediction accuracy of BPNN model in the training and testing.

Crash Injury Severity	Training (%)			Testing (%)		
	B-BPNN	T-BPNN	BT-BPNN	B-BPNN	T-BPNN	BT-BTNN
No injury	87.252	85.972	86.022	80.532	76.740	78.338
Non-incapacitating	84.421	78.883	80.554	73.101	67.263	71.679
Incapacitating/fatal	79.334	71.053	79.899	66.711	59.579	63.578
Overall	82.769	76.637	81.269	72.559	66.750	70.086

It can be seen from the above Tables 7–9 that the classification performance of SVM model is the best, with the 93.176, 87.111, and 88.442% accuracy for B-SVM, T-SVM, and BT-SVM respectively in the training, and with 88.001, 84.712, and 85.229% accuracy in the testing, followed by the BPNN and OL models. This would be attributed to the fact that SVM can map high-dimensional data using the RBF kernel function, while the traditional statistical model has poor prediction performance with the high-dimensional data.

Also, the results show that the classification accuracy for no injury is the highest in SVM, BPNN, and OL models in the training and testing, and then followed by non-incapacitating and incapacitating/fatal, which can be explained by the fact that the probability of no injury caused by crash accident is higher than that of non-incapacitating and incapacitating/fatal.

For the three decisions, the prediction effect of the crash injury severity prediction model corresponding to the turning decision is lower than that of the other decisions. It reveals that the crash characteristic significance that makes the vehicle take turning avoidance decision are lower than that of braking and the braking + turning.

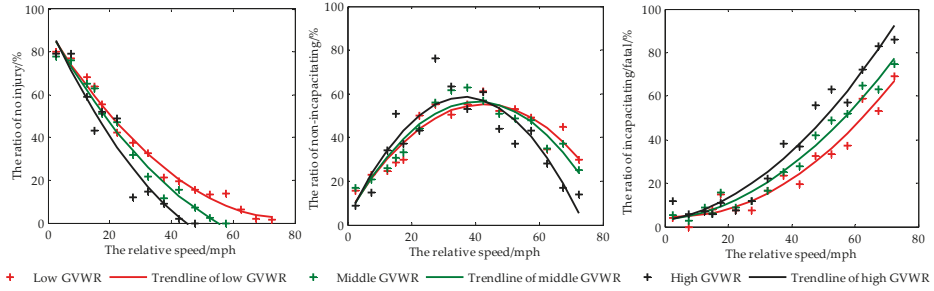
5.3. Sensitive Analysis of REL\_SPEED and GVWR on the Crash Injury Severity

The previous literature has shown that the REL\_SPEED and GVWR are important impact indicators of the crash injury severity, and the two variables have been selected as significant variables in the parameters estimation of OL models. So how does the REL\_SPEED and GVWR affect the crash injury severity? In this paper, we quantitatively evaluate the effects of REL\_SPEED and GVWR on crash injury severity by analyzing the sensitivity of B-SVM, T-SVM, and BT-SVM to the changes in REL\_SPEED and GVWR, as follows:

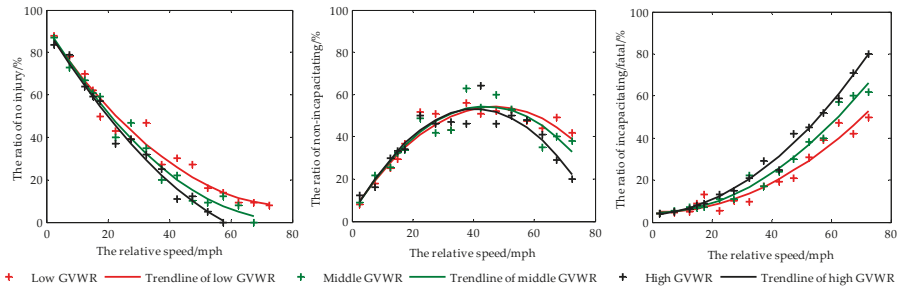
- Firstly, for all the crash samples with GVWR in low range (less than 10,000 lbs) and the REL\_SPEED of 0 to 20 mph, we reset other impact indicators as the standard values i.e.,  $(l_3, l_4, \dots, l_{14}) = (\underbrace{0, 0, \dots, 0}_7, \underbrace{1, 0, \dots, 0}_4)$  (considered as the normal road and environmental condition) and then input them into B-SVM, T-SVM, and BT-SVM model respectively, and calculate the ratio of each crash injury severity output from each SVM model at different REL\_SPEEDs.
- Then, based on the samples with the REL\_SPEED of 20 mph, control other impact indicators unchanged, we gradually increase the REL\_SPEED with an increase unit of 2.5 mph and the maximum limit of 75 mph. Every time the REL\_SPEED changes, a new set of crash samples is obtained and input into each SVM model. From the output of each SVM model, the ratio of each crash injury severity corresponding to the REL\_SPEED is calculated. In this way, we can get the trend that the ratio of each crash injury severity varies with the REL\_SPEED when GVWR is in the low range.
- Finally, based on the above obtained crash samples with GVWR in the low range and the REL\_SPEED of 0 to 75 mph, change the GVWR from low range to middle (10,001~26,000 lbs), high (more than 26,001 lbs) range, respectively. By calculating the ratio of each crash injury severity output from B-SVM, T-SVM, and BT-SVM model respectively, we can get the trend that

the ratio of each crash injury severity varies with the REL\_SPEED when GVWR is in the low, middle, and high range, respectively.

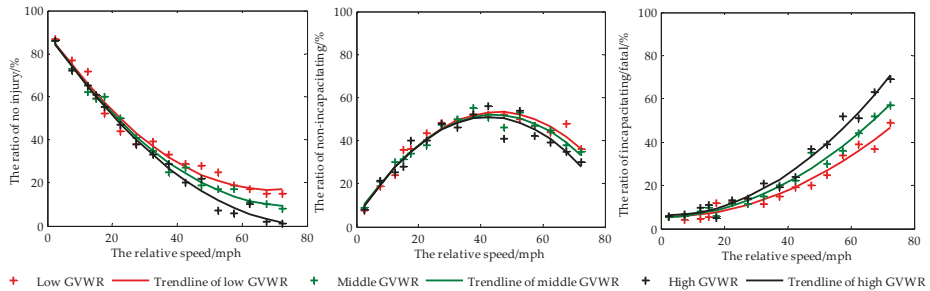
After the operations and data statistics, the quantitative effects of the REL\_SPEED and GVWR on the crash injury severity with the B-SVM, T-SVM, and BT-SVM model are shown respectively in the Figure 9 below.



(a) B-SVM



(b) T-SVM



(c) BT-SVM

**Figure 9.** The changing trend of crash injury severity ratio under different conditions. (a) With the B-SVM model, the changing trend of each injury severity ratio with the REL\_SPEED when the GVWR is in different ranges. (b) With the T-SVM model, the changing trend of each injury severity ratio with the REL\_SPEED when the GVWR is in different ranges. (c) With the BT-SVM model, the changing trend of each injury severity ratio with the REL\_SPEED when the GVWR is in different ranges. From left to right, each column represents the changing trend of no injury ratio, non-incapacitating and incapacitating/fatal ratio with the REL\_SPEED when the GVWR is in different ranges, respectively.

As can be seen from the above Figure 7, in the B-SVM, T-SVM, and BT-SVM model, each injury severity ratio has the similar changing trend when the GVWR is in different ranges. Take the B-SVM model at the low GVWR range as an example.

- (1) When the REL\_SPEED is in the low range (0–20 mph), the no injury ratio decreases rapidly, and the non-incapacitating ratio increases rapidly, while the ratio of incapacitating/fatal has no significance change. This phenomenon indicates that in the low REL\_SPEED range, with the increase of the REL\_SPEED, most of the decreased no injury accidents is converted to non-incapacitating accidents.
- (2) When the REL\_SPEED is in the middle range (20–45 mph), the increasing rate of the non-incapacitating ratio decreases gradually, while the increasing rate of the incapacitating/fatal ratio increases gradually, which indicates that in the middle REL\_SPEED range, the conversion from the decreased no injury accidents to incapacitating/fatal accidents is increasing gradually, and the conversion from the decreased no injury accidents to the non-incapacitating accidents is decreasing gradually.
- (3) When the REL\_SPEED is in the high range (45–75 mph), the no injury ratio tends to 0 gradually, and the non-incapacitating ratio decreases rapidly, while the ratio of incapacitating/fatal increases rapidly, which reveals that in the high REL\_SPEED range, most of the increased incapacitating/fatal accidents are converted from the decreased non-incapacitating accidents.
- (4) The results show that, with the same other conditions, the consequence of vehicle crash will become more serious as the REL\_SPEED increases.

In addition, the figures also show that with the three SVM models, when GVWR changes from the low range to the middle and high range, the no-injury ratio decreases to a greater extent, and the incapacitating/fatal ratio increases to a greater extent, which indicates that with the same other conditions, the increase of GVWR will increase the vehicle crash severity.

#### 5.4. Comparison of the Crash Injury Severity under Various Emergency Decisions

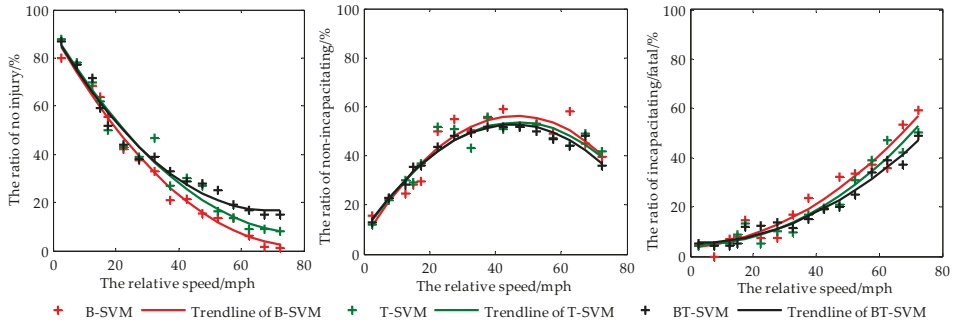
In emergency situation, the autonomous vehicle must first compare the crash injury severity by making each emergency decision, and then make the decision with the minimum crash injury severity. Therefore, it is necessary to make a comparative analysis of the crash injury severity corresponding to each emergency decision and mine the decision-making rules in different emergency conditions from the driving data of regular vehicle.

In this paper, we take the REL\_SPEED and GVWR as examples, maintain the other impact indicators to take standard values  $(l_3, l_4, \dots, l_{14}) = (\underbrace{0, 0, \dots, 0}_7, \underbrace{1, 0, \dots, 0}_4)$ , and respectively change

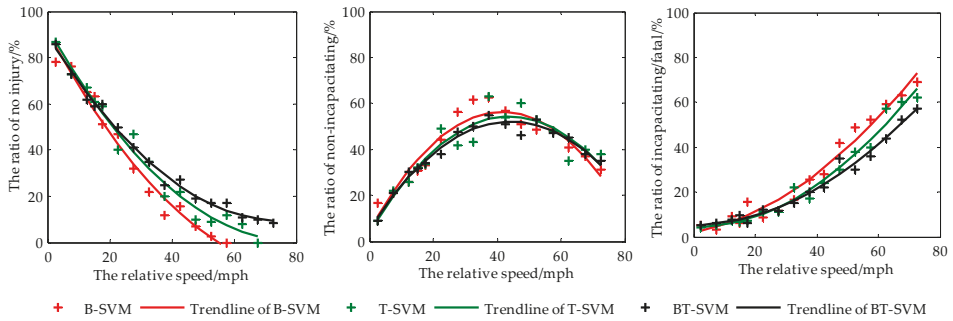
REL\_SPEED from 0 to 75 mph and GVWR from the low to the high range gradually, to compare and analyze the changing trend of the output of B-SVM, T-SVM, and BT-SVM with the change of REL\_SPEED and GVWR, respectively. The results are as shown in Figure 10. In fact, the results of these figures are the same as those in the previous section. However, the line combinations in each figure are different in the two section, which have different exploring purposes.

As can be seen from the Figure 10, when the GVWR is in the low range, in the low REL\_SPEED range (0–20 mph), the ratio of each crash injury severity is basically the same in the output of the three SVM models, which shows that the crash injury severity caused by making three emergency decisions is the same in the case of an emergency. In the middle REL\_SPEED range (20–45 mph), there is no significant difference in the ratio of each crash injury severity for the output of T-SVM and BT-SVM models, but they output a lower ratio of non-incapacitating and incapacitating/fatal than B-SVM, and a higher ratio of no injury than B-SVM, this means that compared with the braking decision, making turning and braking + turning can reduce the crash injury severity and have the same contribution effect. In the high REL\_SPEED range (45–75 mph), among the three SVM models, the BT-SVM model output the highest no injury ratio, and the lowest non-incapacitating and incapacitating/fatal ratio,

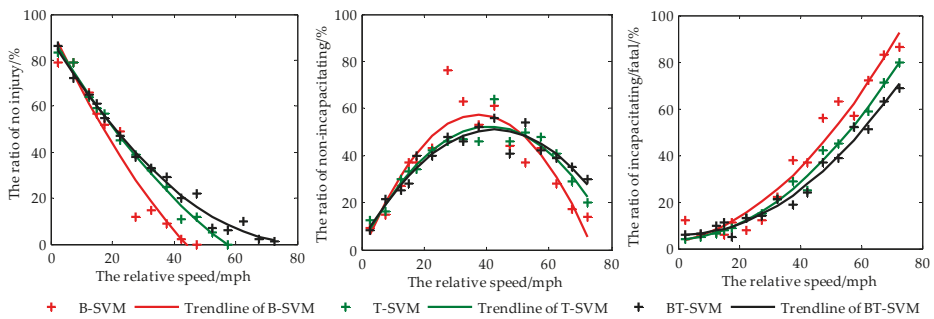
which indicates that under this situation, to reduce the crash injury severity, autonomous vehicles need to make braking and turning decisions simultaneously.



(a) GVWR in low range



(b) GVWR in the middle range



(c) GVWR in the high range

**Figure 10.** The changing trend of crash injury severity ratio with different SVMs. (a) When GVWR is in the low range, the changing trend of each crash injury severity ratio with the REL\_SPEED in different SVM models. (b) When GVWR is in the middle range, the changing trend of each crash injury severity ratio with the REL\_SPEED in different SVM models. (c) When GVWR is in the high range, the changing trend of each crash injury severity ratio with the REL\_SPEED in different SVM models.

When the GVWR changes from the low range to middle, high range in turn, the low and middle REL\_SPEED ranges that can achieve the above results are gradually moving forward, which makes

the low and middle REL\_SPEED range smaller and smaller, and the high REL\_SPEED range larger and larger.

For the same input, when the crash injury severity between emergency decisions is consistent, in order not to affect the traffic operation of other lanes and make the driving habits of autonomous vehicles consistent with regular vehicles, autonomous vehicles should try to take the simplest action.

## 6. Conclusions

In this paper, the two-vehicle crash caused by the emergency braking of the frontal vehicle was taken as the research object. Based on the SVM model and the NASS/GES crash samples, the SVM crash injury severity prediction models corresponding to three emergency decisions (namely B-SVM, T-SVM, and BT-SVM) were trained for autonomous vehicles through the process of the parameter optimization and the optimal kernel function selection. Compared with the other two kernel functions, the training accuracy of three SVM models with the RBF kernel function was the highest, which are 93.1758, 87.1112, and 88.4422% respectively. Then, with the test samples the SVM model were compared with the OL and BPNN models for classification accuracy, the comparison results showed that the SVM model had the best classification performance on the crash injury severity with the classification accuracy for more than 84%.

Secondly, based on the B-SVM, T-SVM, and BT-SVM model, the sensitivity analysis was conducted to explore the effects of REL\_SPEED and GVWR on the crash injury severity. The results showed that with the same conditions, increasing the REL\_SPEED and GVWR would make the crash more serious. Therefore, in the prediction models, these two variables could not be ignored.

Finally, in order to make the best emergency decision for autonomous vehicles in emergencies, this paper input the crash samples with normal road and environmental conditions into B-SVM, T-SVM, and BT-SVM model respectively, then calculated and compared the ratio of the crash injury severity output from each model. This study found that in the emergency situations, when the GVWR was in the low range, with the same other conditions, as the REL\_SPEED increased from the low to middle and then to the high range, the best emergency decision with the minimum crash injury severity would gradually transition from braking to turning and then to braking + turning. As the GVWR increased, the low and middle REL\_SPEED ranges that could achieve the above results would be narrowed, and the high REL\_SPEED range would be enlarged. To a certain extent, the results were basically consistent with the actual driving experience, which further demonstrated the rationality of the SVM prediction models in this paper.

Although the SVM crash injury severity prediction models established in this study can help autonomous vehicles to make the emergency decision with the minimum crash injury severity under emergency conditions, and the prediction performance is better than the other models, there are still some weak points and limits. For example, due to the limited data sources, we only start from the crash accident scenario on a simple single lane to explore the feasibility of the prediction methods proposed in this paper. In future research, we will factor adjacent lanes into our prediction methods, and we will use V2X equipment to deploy long-term micro-data collection strategy in the future to explore this research in depth and establish a better crash injury severity assessment model for autonomous vehicles.

**Author Contributions:** All authors contributed to the paper. Y.L. and J.Z. conceived and designed the experiments; S.W. and Y.L. performed the experiments and analyzed the data; J.H., S.L., and J.Z. contributed reagents/materials/analysis tools; J.Z. and Y.L. wrote the paper.

**Funding:** This research was funded by the National Natural Science Foundation of China (grant numbers 51678320 and 71801144); the Science & Technology Innovation Fund for Graduate Students of Shandong University of Science and Technology (grand number SDKDYC180373); the Education Innovation Project for Graduate Students of Shandong University of Science and Technology (grant number KDYC16026).

**Conflicts of Interest:** The authors declare no conflict of interest.



## References

1. Shawky, A.M.; Kishta, M.; Al-Harathi, H.A. Investigating Factors Affecting the Occurrence and Severity of Rear-End Crashes. *Transp. Res. Procedia* **2017**, *15*, 2098–2107.
2. United States Department of Transportation. The General Estimates System (GES) 2012–2015, National Automotive Sampling System (NASS). Available online: <https://www.nhtsa.gov/research-data/national-automotive-sampling-system-nass> (accessed on 5 April 2018).
3. Clinton, V.O.; John, S.S. Analyzing Road Safety in the United States. *Res. Transp. Econ.* **2013**, *43*, 98–111.
4. Lv, W.; Song, W.G.; Liu, X.D.; Ma, J. A Microscopic Lane Changing Process Model for Multilane Traffic. *Phys. A Stat. Mech. Appl.* **2013**, *392*, 1142–1152. [[CrossRef](#)]
5. Zhang, J.Y.; Liao, Y.P.; Wang, S.F. Study on Driving Decision-Making Mechanism of Autonomous Vehicle Based on an Optimized Support Vector Machine Regression. *Appl. Sci.* **2018**, *8*, 13. [[CrossRef](#)]
6. Zheng, J.; Suzuki, K.; Fujita, M. Car-following Behavior with Instantaneous Driver-vehicle Reaction Delay: A Neural-network-based Methodology. *Transp. Res. Part C Emerg. Technol.* **2013**, *36*, 339–351. [[CrossRef](#)]
7. Lord, D.; Mannering, F. The Statistical Analysis of Crash-frequency Data: A Review and Assessment of Methodological Alternatives. *Transp. Res. Part A Policy Pract.* **2010**, *44*, 291–305. [[CrossRef](#)]
8. Wu, Q.; Zhang, G.H.; Ci, Y.S.; Wu, L.N.; Tarefder, R.A. Exploratory Multinomial Logit Model-based Driver Injury Severity Analyses for Teenage and Adult Driver in Intersection-related Crashes. *Traffic Inj. Prev.* **2016**, *17*, 413–422. [[CrossRef](#)]
9. Young, R.K.; Liesman, J. Estimating the Relationship Between Measured Wind Speed and Overturning Truck Crashes Using a Binary Logit Model. *Accid. Anal. Prev.* **2007**, *39*, 574–580. [[CrossRef](#)]
10. Manan, M.M.A.; Várhelyi, A.; Çelik, A.K.; Hashim, H.H. Road Characteristics and Environment Factors Associated with Motorcycle Fatal Crashes in Malaysia. *IATSS Res.* **2017**. [[CrossRef](#)]
11. Shankar, V.; Mannering, F. An Exploratory Multinomial Logit Analysis of Single-vehicle Motorcycle Accident Severity. *J. Saf. Res.* **1996**, *27*, 183–194. [[CrossRef](#)]
12. Ye, F.; Lord, D. Comparing Three Commonly Used Crash Severity Models on Sample Size Requirements: Multinomial Logit, Ordered Probit and Mixed Logit Models. *Anal. Methods Accid. Res.* **2014**, *1*, 72–85. [[CrossRef](#)]
13. Abdel-Aty, M. Analysis of Driver Injury Severity Levels at Multiple Locations Using Ordered Probit Models. *J. Saf. Res.* **2003**, *34*, 597–603. [[CrossRef](#)]
14. Ayuso, M.; Santolino, M. Predicting Automobile Claims Bodily Injury Severity with Sequential Ordered Logit Models. *Insur. Math. Econ.* **2007**, *41*, 71–83. [[CrossRef](#)]
15. Garrido, R.; Bastos, A.; Almeida, A.D.; Elvas, J.P. Prediction of Road Accident Severity Using the Ordered Probit Model. *Transp. Res. Procedia* **2014**, *3*, 214–223. [[CrossRef](#)]
16. Lee, C.; Li, X. Analysis of Injury Severity of Drivers Involved in Single- and Two-vehicle Crashes on Highways in Ontario. *Accid. Anal. Prev.* **2014**, *71*, 286–295. [[CrossRef](#)] [[PubMed](#)]
17. Shaheed, M.S.; Gkritza, K.; Carriquiry, A.L.; Hallmark, S.L. Analysis of Occupant Injury Severity in Winter Weather Crashes: A Fully Bayesian Multivariate Approach. *Anal. Methods Accid. Res.* **2016**, *11*, 33–47. [[CrossRef](#)]
18. Kashani, A.T.; Mohaymany, A.S. Analysis of the Traffic Injury Severity on Two-Lane, Two-Way Rural Roads Based on Classification Tree Models. *Saf. Sci.* **2011**, *49*, 1314–1320. [[CrossRef](#)]
19. Mujalli, R.O.; Oña, J.D. A Method for Simplifying the Analysis of Traffic Accidents Injury Severity on Two-lane Highways Using Bayesian Networks. *J. Saf. Res.* **2011**, *42*, 317–326. [[CrossRef](#)]
20. Zeng, Q.; Huang, H. A Stable and Optimized Neural Network Model for Crash Injury Severity Prediction. *Accid. Anal. Prev.* **2014**, *73*, 351–358. [[CrossRef](#)]
21. Delen, D.; Tomak, L.; Topuz, K.; Eryarsoy, E. Investigating Injury Severity Risk Factors in Automobile Crashes with Predictive Analytics and Sensitivity Analysis Methods. *J. Transp. Health.* **2017**, *4*, 118–131. [[CrossRef](#)]
22. Tolouei, R.; Maher, M.; Titheridge, H. Vehicle Mass and Injury Risk in Two-Car Crashes: A Novel Methodology. *Accid. Anal. Prev.* **2013**, *50*, 155–166. [[CrossRef](#)] [[PubMed](#)]
23. Jurewicz, C.; Sobhani, A.; Woolley, J.; Dutschke, J.; Corben, B. Exploration of Vehicle Impact Speed—Injury Severity Relationships for Application in Safer Road Design. *Transp. Res. Procedia* **2016**, *14*, 4247–4256. [[CrossRef](#)]

24. Performance of Lap/Shoulder belts in 167 Motor Vehicle Crashes (Volume I); NTSB/SS-88/02; National Transportation Safety Board: Washington, DC, USA, 1926; pp. 14–17.
25. Hongguo, X. Vehicle accident mechanics. In *Automobile Accident Engineering*; China Communications Press: Beijing, China, 2004; ISBN 7-114-05044-5.
26. Shelley, B.; Rajesh, P.; Lacramioara, B. A Modified Rank Ordered Logit model to analyze injury severity of occupants in multivehicle crashes. *Anal. Methods Accid. Res.* **2017**, *14*, 22–40.
27. Saha, P.; Roy, N.; Mukherjee, D.; Sarkar, A.K. Application of Principal Component Analysis for Outlier Detection in Heterogeneous Traffic Data. *Procedia Comput. Sci.* **2016**, *83*, 107–114. [[CrossRef](#)]
28. Nagendra, S.M.S.; Khare, M. Principal component analysis of urban traffic characteristics and meteorological data. *Transp. Res. Part D Transp. Environ.* **2003**, *8*, 285–297. [[CrossRef](#)]
29. Shafizadeh-Moghadam, H.; Tayyebi, A.; Ahmadlou, M.; Delavar, M.R.; Hasanlou, M. Integration of genetic algorithm and multiple kernel support vector regression for modeling urban growth. *Comput. Environ. Urban Syst.* **2017**, *65*, 28–40. [[CrossRef](#)]
30. Hong, W.-C.; Dong, Y.C.; Zheng, F.F.; Lai, C.-Y. Forecasting urban traffic flow by SVR with continuous ACO. *Appl. Math. Model.* **2011**, *35*, 1282–1291. [[CrossRef](#)]
31. Zhu, L.L.; Liu, L.; Zhao, X.P.; Yang, D. Driver Behavior Recognition Based on Support Vector Machine. *J. Transp. Syst. Eng. Inf. Technol.* **2017**, *17*, 92–97.
32. Babazadeh, A.; Poorzahedy, H.; Nikoosokhan, S. Application of Particle Swarm Optimization to Transportation Network Design Problem. *J. King Saud Univ. Sci.* **2011**, *23*, 293–300. [[CrossRef](#)]
33. Deng, W.; Zhao, H.M.; Yang, X.H.; Xiong, J.X.; Meng, S.; Bo, L. Study on an Improved Adaptive PSO Algorithm for Solving Multi-Objective Gate Assignment. *Appl. Soft Comput.* **2017**, *59*, 288–302. [[CrossRef](#)]



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# Ethical and Legal Dilemma of Autonomous Vehicles: Study on Driving Decision-Making Model under the Emergency Situations of Red Light-Running Behaviors

Sixian Li, Junyou Zhang \*, Shufeng Wang, Pengcheng Li and Yaping Liao

Department of Transportation Engineering, College of Transportation, Shandong University of Science and Technology, Huangdao District, Qingdao 266590, China; gsypsbyw666@163.com (S.L.); shufengwang@sdust.edu.cn (S.W.); jiangli6211223q@126.com (P.L.); liaoyapingsk@163.com (Y.L.)

\* Correspondence: junyouzhang@sdust.edu.cn; Tel.: +86-139-0532-3314

Received: 7 September 2018; Accepted: 19 October 2018; Published: 22 October 2018

**Abstract:** Autonomous vehicles (AVs) are supposed to identify obstacles automatically and form appropriate emergency strategies constantly to ensure driving safety and improve traffic efficiency. However, not all collisions will be avoidable, and AVs are required to make difficult decisions involving ethical and legal factors under emergency situations. In this paper, the ethical and legal factors are introduced into the driving decision-making (DDM) model under emergency situations evoked by red light-running behaviors. In this specific situation, 16 factors related to vehicle-road-environment are considered as impact indicators of DDM, especially the duration of red light (RL), the type of abnormal target (AT-T), the number of abnormal target (AT-N) and the state of abnormal target (AT-S), which indicate legal and ethical components. Secondly, through principal component analysis, seven indicators are selected as input variables of the model. Furthermore, feasible DDM, including braking + going straight, braking + turning left, braking + turning right, is taken as the output variable of the model. Finally, the model chosen to establish DDM is the T-S fuzzy neural network (TSFNN), which has better performance, compared to back propagation neural network (BPNN) to verify the accuracy of TSFNN.

**Keywords:** autonomous vehicles; driving decision-making model; the emergency situations; red light-running behaviors; ethical and legal factors; T-S fuzzy neural network

## 1. Introduction

Research on autonomous vehicles (AVs) has examined social and technological trends in the development of future vehicles for their potential value. Strategic innovations in AVs could dramatically reduce congestion, emissions, traffic accidents, and the number of vehicles [1]. Furthermore, autonomous driving technology liberates humans from the driving task and significantly eliminates operation error caused by humans. Equipped with an intelligent system, AVs will complete attentive and precise tasks, including environment perception, decision-making, motion-planning, control, and execution [2]. Among them, decision-making systems have the capability to deal with complex decision environments and involve the layout of mathematical models [3]. Combined with comprehensive cognitive sequence activities, it is required for AVs to design a driving decision-making (DDM) model to form a prompt and accurate driving strategy. Taking the driver's behavior characteristic as the core, the micro-model, such as braking and lane-changing as the carrier, and the driver's behavior decision-making is modeled by machine learning [4]. Consequently, DDM is established in line with human driving habits. It is a key technology for the implementation of AVs and the advanced driver assistant systems (ADAS).

Nevertheless, at present, DDM of AVs are mainly focused on normal driving situations [5] to avoid collisions. DDM systems are usually affected by many elements, such as humans, vehicles, roads, and environments [6]. One particular challenge is the limitation of pattern recognition and obstacle detection, especially in most cases due to dead zones, object transparency, light reflection, weather conditions, and sensor failure [7]. Therefore, some collisions are bound to happen under emergent dangerous situations, which are defined as emergency situations. However, little research is focused on DDM under emergency situations.

The design of DDM under emergency situations is faced with ethical and legal dilemmas. AVs should reduce traffic accidents and avoid losses, but they will have to choose one between two evils, such as hitting a barrier and killing passengers to protect pedestrians or protecting passengers at the sacrifice of pedestrians [8]. The “trolley problem”, a typical case of decision-making under emergency situations, has rapidly becoming the most recognizable scientific example of ethical and legal situations, where individuals must decide whether to change the direction of a trolley that will sacrifice one person to spare five passengers [9–11]. It is difficult for human beings to reach consensus in decision-making when dealing with sacrificial dilemmas. In addition, AVs must obey the law. However, there are few legal documents for AVs, and even fewer judge the liability of AVs in traffic accidents. With more and more AVs on our roads, how to judge liability and who is scheduled to be held responsible in case of traffic accidents must be decided, which involves not only legal questions, but also moral ones [12]. Therefore, liability directly affects the design of DDM under emergency situations. On the other hand, the use of machine learning techniques in ethics and legal dilemmas concentrates on discriminating rules and principles that are often left implicit or obscured in controversy involving ethical philosophy, psychology, and legal rule [13]. How to integrate the ethical and legal factors into the DDM of AVs which accords with human-like decision-making mechanism is a difficult problem requiring an urgent solution.

Bandana et al. [14] took full account of the legal factors in the process of lane-changing for AVs and used a fuzzy controller to convert traffic rules into logical ones to control AVs, avoiding obstacles without violating traffic regulations. Sarah et al. [15] studied lane-change decision-making when AVs encountered obstacles. They took the comfort degree of the occupant as the ethical factor in the model predictive control (MPC) and verified the cost function decision model by the experiment. Furthermore, Goodall [16] proposed a three-stage strategy to standardize the ethical decision-making behavior of AVs. The determined criteria for collision evaluation (such as death is more serious than injury) were formed by ethics. Then, the results of the simulated collision test were combined with the neural network for machine learning. The internal knowledge of the neural network was transformed into an extractable rule to improve DDM. In addition, Iyad Rahwan, a cognitive scientist at the Massachusetts Institute of Technology, conducted an online test called “Moral Machines” [17] to study the effects of ethics on driving behavior, observing people’s choices for different emergency situations.

At present, much research uses neural network [18–20], decision tree model [21], support vector machine regression [22], fuzzy set theory [23], expert system [24] and Petri net [25] to establish DDM to study knowledge acquisition and presentation. Among them, neural networks are the most widely used and successful learning algorithms to establish DDM. Neural networks have many advantages, such as parallel computing, distributed information storage, fault tolerance, adaptive learning, and so on. However, it is not suitable for representing rule-based knowledge. Meanwhile, fuzzy logic is a process of uncertain and nonlinear reasoning. It is more suitable to express fuzzy and qualitative knowledge. Its reasoning mode is more similar to human thinking. However, generally speaking, it is not easy to achieve the function of adaptive learning. To sum up, the fuzzy neural network combines the knowledge configuration of fuzzy logic and the self-study ability of neural network, so it has the ability of uncertainty reasoning and self-study [26]. Combined with the advantages of artificial neural network and fuzzy system, Takagi-Sugeno fuzzy neural network (TSFNN) is chosen to establish DDM in this paper. On the one hand, the parameters in TSFNN have clear physical meaning, and can be assigned according to human experience, thus greatly improving the convergence rate.

On the other hand, it is simple to compute, which makes it handle large amounts of training samples well. Meanwhile, it has strong adaptive ability to constantly correct parameters through self-learning. In addition, in the research of DDM, TSFNN is widely used for obstacle avoidance control of intelligent vehicles [27,28].

Previous research has failed to study DDM by ethical or legal factors under emergency situations, which remains at the theoretical stage. In addition, the above study only considered either the ethical or legal factor, both of which have a significant impact on DDM and cannot be ignored.

Only when the ethical or legal factors are used in a specific situation can we judge whether DDM is consistent with the code of ethics and also legal. Therefore, in this paper, we take the emergency situation evoked by red light-running behaviors as an example. In addition, ethical or legal factors are described as the specific quantitative indicators in the situations by human drivers using a questionnaire.

Specifically, this study makes the following contributions:

- In this specific scene, DDM under the emergency situation is subtly proposed, combined with ethical and legal analysis.
- Quantitative ethical and legal factors are represented by specific indicators under emergency situations of red light-running behaviors. The duration of red light (RL), the type of abnormal target (AT-T), the number of abnormal target (AT-N) and the state of abnormal target (AT-S) are chosen to represent the ethical and legal factors.
- TSFNN model is developed to accomplish the inherent complex driving decisions, including braking + going straight, braking + turning left, braking + turning right. By analyzing the experimental data, TSFNN has better performance, compared to BPNN, for establishing DDM.

The remainder of the paper is organized as follows: Section 2 introduces the emergency situations of red light-running behaviors. In addition, the decision-making in this situation is analyzed by the dimensions of ethics and law. Section 3 presents the DDM involving ethical and legal factors. In Section 4, a virtual decision-making experiment is designed to provide samples for DDM. In addition, the validity of the model is verified by analyzing the experimental data. Finally, the conclusion is drawn in Section 5.

## 2. Analysis of the Emergency Situations

### 2.1. Definition of the Emergency Situations of the Red Light-Running Behaviors

In this paper, the emergency situations of red light-running behaviors particularly refer to where abnormal targets (pedestrians, non-motor vehicles, pets, etc., collectively regarded as “abnormal target”) continue to pass through a crosswalk, even though the RL is on in the crosswalk of the intersection. In this process, the avoidance space caused by the sudden entry of an abnormal target is insufficient. Therefore, a collision cannot be avoided under the emergency situations. A typical case is shown in Figure 1. We regard the blue vehicle as the AV. At this scene, the RLs are on for the crosswalk and left-turn lane. The left view of the AV is obstructed by the left turn-waiting green bus, thus the abnormal target fails to be detected in advance by the blue vehicle. It is about to encounter the emergency situation.

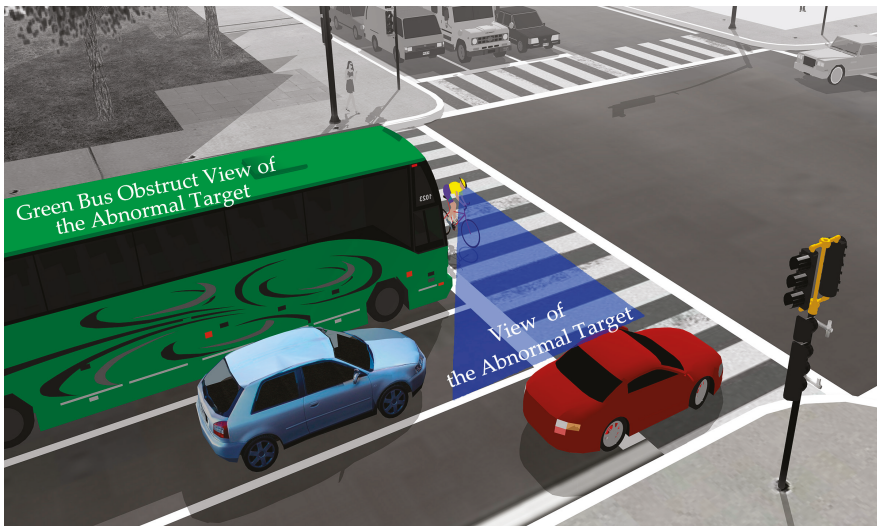


Figure 1. Typical emergency situation.

## 2.2. Analysis the Ethical Factor of DDM

Ethics refers to the norms of people’s internal values and external behavior [29]. To behave like a human being, it is necessary for a module to interpret ethics and morality into indicators by machine learning. Based on the statistical analysis of hundreds of questionnaires in “Moral Machines” and combined with the specific scene of the emergency situations, the ethical indicators of DDM are classified into the following four types.

- Type of Abnormal Target: refers to the types of targets in front of AVs under emergency situations, which are divided into human beings, animals, and transportation facilities. Meanwhile, human beings can be further subdivided into age, gender, and physical condition and so on.
- Number of Abnormal Target: refers to the number of targets in front of AVs under emergency situations. In addition, a survey [8] shows that saving more lives is a significant factor in DDM. Decision-making made by human drivers will change dramatically as the number of rescue targets changes.
- Special State of Abnormal Target: refers to prominent features that distinguish a target from an ordinary person, such as thieves, pregnant women, tramps, etc., as well as people violating the law (red light-running or reverse driving).
- Priority Protection: The principle of priority protection in driving decision can be divided into protecting the passengers and personnel inside the vehicle, protecting the pedestrians outside the vehicle, or reducing the total loss of the collision and so on. AVs may avoid harming several pedestrians by swerving to sacrifice one pedestrian or be faced with the dilemma of sacrificing its own passenger to save one or more pedestrians.

In fact, if all the above indicators are considered, the dimensions are sizable, and the complexity of DDM will be dramatically increased. At the same time, differences in individual cognition will cause considerable controversy in the selection of specific indicators. Therefore, we select the first two indicators for the study. Referring to German ethical principles of AVs [30], we define selection principles of ethical factors as follows: distinguish pedestrians, non-motorized vehicles, and pets; do not distinguish pedestrian’s age, gender, race, or body condition; do not distinguish brand, value of vehicles; do not distinguish breed of pets.

2.3. Analysis the Legal Factor of DDM

Under emergency situations, drivers will take full account of their own liability when making decisions. In China, due to a lack of legal documentation to deal with AV traffic accidents, we take existing legal documents as the basis for the legal analysis of the traffic accident liability judgment.

Traffic accident liability judgment of colliding with the abnormal target while going straight is as follows:

Referring to a real case of “red light-running behaviors” [31], we feature the detail of traffic accident liability judgment which is mainly related to the type (AT-T), state (AT-S), position of the abnormal target as well as the duration of RL through the crosswalk.

1. In case of the emergency situations, if the RL comes on and the abnormal target has passed the central line of the crosswalk, the driver has an obligation to avoid collision;
2. If the RL has lasted for a period of time and the abnormal target fails to pass the central line of the crosswalk, the liability should be determined according to the specific situations.

Traffic accident liability judgment of colliding with the abnormal target while changing lanes is as follows:

According to the relevant legal documents in China, the lane-change decision taken by the driver under dangerous conditions is defined as an emergency risk aversion [32]. If the driver can ensure the total loss caused by changing direction and collision is smaller than that of not changing lane, the driver’s lane-change decision will be allowed and the party causing the dangerous situations should bear full liability. In the actual situation, the damage caused by the collision is difficult to describe quantitatively, and traffic police departments rely more on experience to judge and make the responsibility determination. To reduce complexity, it is assumed that all lane-change decisions meet the definition of the emergency risk aversion. The results of the traffic accident liability analysis are shown in Table 1.

**Table 1.** The results of the traffic accident liability analysis.

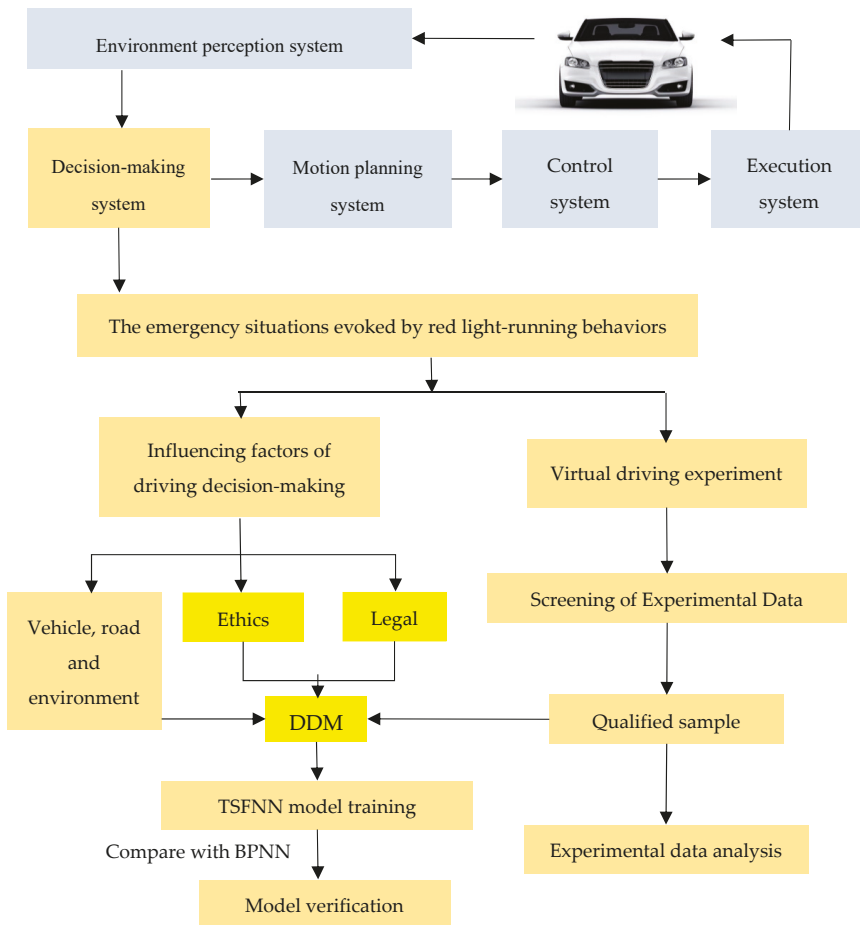
Serial Number	RL	AT-T	AT-S	Location of Abnormal Target	Liability of Drivers	
					Going Straight	Changing Lanes
1	Red light Comes on Just Now	Pedestrians	Run	Has Already Passed the Center Line Just Now	Above Primary Liability	
2		Non-motor Vehicles	Hold			
3			Ride			
4		Pet	Hold			
5			Not Pull			
6	Red light Has Already On	Pedestrians	Run	Has Not Passed the Center Line Yet	Equal Liability	
7		Non-motor Vehicles	Hold			
8			Ride			
9		Pet	Pull			
10			Not Pull			
					Secondary Liability	No Liability
					No Liability	

3. Design of DDM

Based on the environment perception, the decision-making system is central, which analyzes logic reasoning and provides decision-making for AVs. Among them, DDM is the mathematical model of the decision-making system, which plays an important role in vehicle avoidance and conflict avoidance. Using situation information, the environment perception system will detect the emergency situation mentioned in this paper. In addition, then, the decision-making system will classify and predict the behavior of other targets in the situations by a specific classifier.

3.1. Establish of DDM

We design a set of complete processes to establish the DDM, as shown in Figure 2.



**Figure 2.** Design of driving decision-making model (DDM) under the emergency situations of red light-running behaviors.

DDM is supposed to have the ability to learn, so that it can be perfected during learning. Therefore, the whole process is divided into two parts. The first part is the theoretical modeling. We need to clear the impact factors of decision-making under the emergency situations of red light-running behaviors [33]. According to the analysis of Section 2, covering the conventional vehicle, road, environment, as well as ethics and law, the selected factors are regarded as the input variables of DDM, and the output variable of DDM is the driving decision-making. The second part is acquisition of the data; we designed a virtual driving experiment that reproduces the emergency situations of red light-running behaviors. The driver controls the driving simulator to complete the experiment, which provides data for the learning of DDM.

Nonetheless, in the previous analysis of ethics and law, there are still some vague concepts in the indicators to characterize driving decisions, such as some legal factors, especially the duration of RL, and how long it can take to describe a signal light that has just changed from green to red? In this paper, we will present a clear answer. Furthermore, to enable DDM to deal with fuzzy information and have the ability to learn, we build a DDM based on TSFNN and compare it with BPNN to verify the effectiveness and superiority of TSFNN.



3.2. TSFNN

A TSFNN is introduced in this paper and its structure is shown in Figure 3.

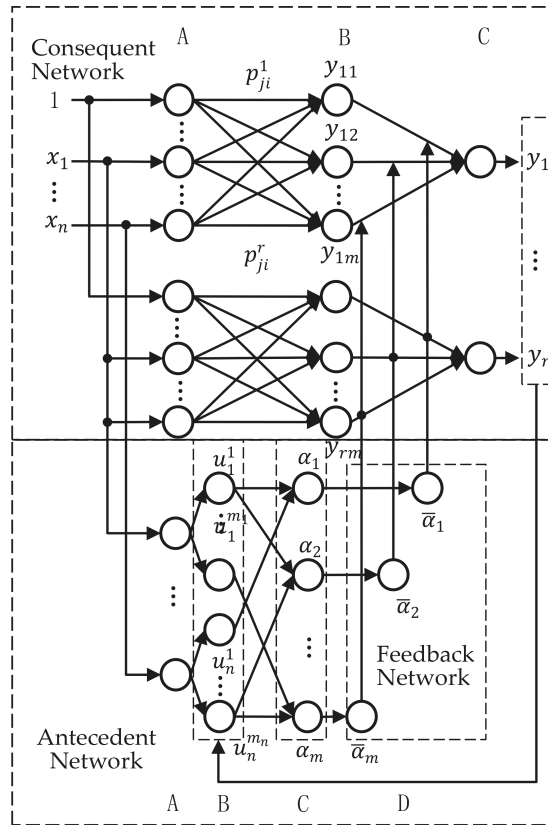


Figure 3. Structure of T-S fuzzy neural network (TSFNN).

Fuzzy neural network merges the reasoning ability of fuzzy logic system with the self-learning ability of artificial neural network. It is a powerful tool to address uncertainty and nonlinear problems. Since its fuzzy rules use a linear equation as the conclusion, in the process of dealing with multivariable systems, the number of fuzzy rules can be effectively reduced. In addition, it is easier to combine with self-adaptive method, which has been widely used in the field of intelligent control and decision-making.

The basic principle of the TSFNN is to characterize the membership degree  $u_f(u)$  of each element  $u$  to the fuzzy subset  $f$  by specific numerical values, so that many fuzzy concepts can be quantitatively described. The specific rules are as follows.

Suppose  $x = [x_1, x_2, \dots, x_n]^T$  denotes an input vector, each component  $x_i$  is a fuzzy linguistic variable, then:

$$R_f : \text{If } x_1 \text{ is } A_1^j, x_2 \text{ is } A_2^j, \dots, x_n \text{ is } A_n^j \tag{1}$$

$$\text{then } y_j = p_{j0} + p_{j1}x_1 + \dots + p_{jn}x_n \tag{2}$$

where  $R_j$  is the  $j^{\text{th}}$  fuzzy rule,  $A_i^j$  is the  $j^{\text{th}}$  linguistic value of the input variable  $x_i$ ,  $y_i$  is the output variable, according to the fuzzy rule,  $p_{ji}$  is the fuzzy system parameter.

3.2.1. The Antecedent Network of TSFNN

The antecedent network is divided into 4 layers [34,35].

Layer A (Input layer): The number of nodes is determined by the number of inputs, which is used to convey each input variable.

Layer B (Fuzzy layer): It is used to calculate the membership degree  $\mu_{A_j^i}$  of each input variable. In this study, the Gaussian membership function is adopted.

$$y = e^{-\frac{(x-c)^2}{\sigma^2}} \tag{3}$$

The parameter  $c$  and  $\sigma$  determine the center point of the function and the width of Gaussian membership function, respectively.

Layer C (Rule layer): It is used to calculate the firing strength  $\alpha_j$  of every fuzzy rule. In this paper, the continuous multiplication operator is adopted.

$$\alpha_j = u_{A_1^j}(x_1) * u_{A_2^j}(x_2) \dots * u_{A_n^j}(x_n) \tag{4}$$

where  $u_{A_i^j}(x_i)$  is the corresponding subordinate function.

Layer D (Normalized layer): It is used to calculate the normalized firing strength of corresponding rules [36]. The output variable  $\bar{\alpha}_j$  of the front part network is calculated by a weighted average method, which can be given by:

$$\bar{\alpha}_j = \alpha_j / \sum_{i=1}^m \alpha_i \tag{5}$$

3.2.2. The Consequent Network of TSFNN

The consequent network of TSFNN is divided into 3 layers, consisting of  $r$  parallel sub-networks with the same structure, each of which produces an output variable.

Layer E (Input layer): To compensate the constant in the fuzzy rule, the  $0^{th}$  node of the input layer is  $x_0 = 1$ .

Layer F (Function layer): It is used to calculate the consequent parameters of every rule. Let input weighted average to the unadjusted rules:

$$y_{ij} = p_{j0}^i + p_{j1}^i x_1 + \dots + p_{jn}^i x_n \tag{6}$$

Layer G (Combined layer): It is the output layer of the entire network.

$$y_i = \sum_{j=1}^m \bar{\alpha}_j y_{ij} \tag{7}$$

where  $y_i$  ( $i = 1, 2, \dots, r$ ) is the weighted sum of each rule. The output of the antecedent network is used as the connection weight of the layer G.

3.2.3. Network Learning Parameters

The learning parameters of TSFNN mainly include the connection weight  $p_{ji}^l$  of the consequent network and the central value  $c_{ij}$  and the width  $\sigma_{ij}$  of the membership functions of the nodes in layer B of the antecedent network. Suppose the error cost function as:

$$E = \frac{1}{2} \sum_{i=1}^r (t_i - y_i)^2 \tag{8}$$

In the formula,  $t_i$  and  $y_i$  represent the desired output and the actual output, respectively. The learning algorithm of the parameter  $p_{ji}^l$  is:

$$\frac{\partial E}{\partial p_{ji}^l} = -(t_1 - y_1)\bar{a}_j x_i \tag{9}$$

$$p_{ji}^l(k+1) = p_{ji}^l(k) + \beta(t_1 - y_1)\bar{a}_j x_i \tag{10}$$

By adjusting parameter  $p_{ji}^l$ , the structure of TSFNN can be simplified. The simplified structure is also a kind of multilayer feedforward network. Therefore, we refer error backpropagation algorithm of Back Propagation (BP) network into the learning algorithm to adjust parameters. The learning algorithm for parameter adjustment is as follows:

$$c_{ij}(k+1) = c_{ij}(k) - \beta \frac{\partial E}{\partial c_{ij}} \tag{11}$$

$$\sigma_{ij}(k+1) = \sigma_{ij}(k) - \beta \frac{\partial E}{\partial \sigma_{ij}} \tag{12}$$

#### 4. Parameters of DDM

The selected parameters of DDM, namely the input and output variables of model, will directly affect the validity of the model.

##### 4.1. Input Variables of DDM

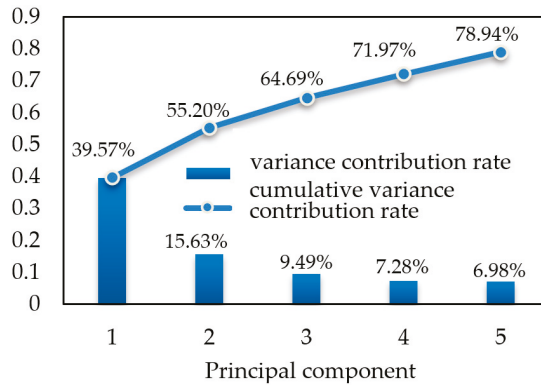
Based on the classification of influencing factors of decision-making in the existing research [37], the ethics and legal factors are taken into account as well, and a total of 16 influencing indicators are selected to design the questionnaires according to the emergency situations evoked by red light-running behaviors. Influencing indicators are as follows: velocity of host vehicle ( $V_1$ ), braking capacity of host car (B), the distance between the host vehicle and the obstacle (D), type of the right-lane vehicle (T), velocity of the right-lane vehicle ( $V_2$ ), the road line shape (LS), the lane width (W), the road mark (RM), duration of red light (RL), the weather condition and the visibility (WE), type of abnormal target (AT-T), number of abnormal target (AT-N), state of abnormal target (AT-S), velocity of abnormal target (AT-V), angle before collision ( $\theta$ ), damage area (DA).

2000 questionnaires were distributed, 1244 were recovered and the recovery rate reached 62.2%, among which 1148 effective questionnaires were obtained, and effective recovery rate reached 92.3%. Through analyzing and processing of the questionnaire data, the input variables of the model are determined.

As for input variables, on the one hand, if the input number is less, the model is too simple to reflect the driver’s decision rule precisely. On the other hand, if it is too much, the coupling relationship between the other influencing factors will increase the complexity of DDM and the training time. In this paper, principal component analysis (PCA) is adopted to convert multiple correlated indicators into fewer linear uncorrelated ones. PCA is an accepted tool in data analysis, and more generally [38]. The results of PCA for 16 influencing indicators are shown in Figure 4.

The cumulative variance contribution rate of the first five principal components reached 78.94%, indicating that components can represent the influencing factors of DDM. The further output of the rotational component matrix is shown in Table 2.

Among them, each load value is the correlation coefficient between each variable and PC. The PC with large correlation coefficients is screened; thus, the decision-making indicator set is determined, which is as shown in Table 3.



**Figure 4.** The results of principal component analysis (PCA) for 16 influencing indicators. The x-axis represents the first five principal components; y-axis represents the variance contribution rate of the principal component.

**Table 2.** Influencing factors of driving decision-making rotational component matrix.

Indicators	Components				
	1	2	3	4	5
V <sub>1</sub>	<b>0.938</b>	-0.045	-0.088	0.022	-0.052
RL	<b>0.927</b>	-0.009	-0.027	0.209	-0.118
AT-N	<b>0.889</b>	-0.073	-0.014	0.255	0.037
W	0.787	-0.106	-0.091	0.303	-0.109
B	-0.079	0.413	0.741	-0.131	0.053
LS	0.728	-0.012	0.134	0.061	-0.112
V <sub>2</sub>	0.699	-0.156	-0.122	0.318	0.097
AT-T	-0.129	<b>0.924</b>	0.064	-0.046	0.048
AT-S	-0.014	<b>0.907</b>	0.127	-0.054	-0.046
RM	-0.207	-0.047	<b>0.898</b>	0.250	-0.228
AT-T	-0.079	0.413	-0.131	<b>0.864</b>	0.053
DA	0.255	0.030	0.691	-0.424	0.242
T	0.511	0.001	-0.014	0.612	0.142
WE	0.491	0.096	-0.086	0.530	-0.148
D	-0.168	0.004	-0.011	0.049	<b>0.832</b>
θ	-0.191	0.032	0.105	-0.017	0.593

What bold indicates higher value in a column are selected as principal components; extraction method: Main component; rotation method: with the Kaiser standard orthogonal rotation method; a. the rotation converges after 8 iterations.

**Table 3.** Indicator set of DDM.

Number	Indicator of Decision-Making
1	Velocity of host vehicle (V <sub>1</sub> )
2	Duration of red light (RL)
3	Number of abnormal target (AT-N)
4	Type of abnormal target (AT-T)
5	State of abnormal target (AT-S)
6	The road marking RM
7	Velocity of the right-lane vehicle (V <sub>2</sub> )
8	The distance between the host vehicle and the obstacle (D)

The indicator set of DDM shows that the above factors with large correlation coefficients have an important impact on driving decision-making. In the analysis of Sections 2.2 and 2.3, the indicator

AT-T and AT-N imply the ethical factors; AT-T, AT-S, RL and RM imply the legal factors. As for the indicator road mark (RM), since the emergency situations occur before the intersection stop line, each lane marking line is a solid line. Drivers cannot drive over the solid line, otherwise they will be judged as violating traffic law in China. Furthermore, there is no discrimination between the samples, thus, RM is removed. The input variables of the remaining 7 indicators of DDM are marked as  $X_1$ – $X_7$ . Details of each input variables is shown in Table 4.

**Table 4.** Information of the input variable of the model.

Input Variables	Meanings of Input Variables	Principle of Process Data
$X_1$	Velocity of host vehicle ( $V_1$ ) refers to the instantaneous speed of the experimental vehicle when the driver is making decisions.	$V_1 \in [0, 40]$ , keep the data; $V_1 \in (40, \infty)$ , eliminate the data.
$X_2$	Duration of red light (RL). $0 \leq RT \leq 3$ , the right light is on just now; $RT > 3$ , the right light has already been on.	$0 \leq RT \leq 3$ , $X_2 = 1$ ; $RT > 3$ , $X_2 = 2$ .
$X_3$	Number of abnormal target (AT-N). Two groups of control experiments of AT-N = 1 and AT-N = 3 are set.	AT-N = 1, $X_3 = 1$ ; AT-N = 3, $X_3 = 2$ .
$X_4$	Type of abnormal target (AT-T) is divided into pedestrians, non-motorized vehicles, pets.	AT-T = pedestrians, $X_4 = 1$ ; AT-T = non-motorized vehicles, $X_4 = 2$ ; AT-T = pets, $X_4 = 3$ .
$X_5$	State of abnormal target (AT-S) is divided into legal and illegal behaviors. Pedestrian running, holding non-motorized vehicles, pulling pets are classified as the legal behaviors; cycling non-motorized vehicles, not pulling pets are classified as the illegal behaviors.	AT-S = legal behaviors, $X_5 = 1$ ; AT-S = illegal behaviors, $X_5 = 2$ .
$X_6$	Velocity of the right-lane vehicle ( $V_2$ ) refers to the instantaneous speed of the vehicle in the right lane when the driver is making a decision.	$V_1 \in [0, 40]$ , keep the data; $V_1 \in (40, \infty)$ , eliminate the data.
$X_7$	The distance between the host vehicle and the obstacle (D) refers to the distance between the centroid of the host vehicle and the abnormal target.	It is obtained joint calculation of the centroid coordinate of each object in the situations, regardless of the height difference in the centroid of two vehicles.

Note: Some provinces and cities in China stipulate that motor vehicles cannot exceed the speed of 40 km/h when passing through urban road intersections.

4.2. Output Variables of DDM

The driving decision-making (D) is taken as the output variable of DDM. In the emergency situations of Section 2.1, the blue vehicle is in the middle lane, and the abnormal target bursts in front of the vehicles. In this case, the driver’s decision-making is divided into the following three situations, as shown in Table 5.

**Table 5.** Output variables of decision-making models.

Decision-Making	Symbol	Threshold	Output Threshold Range
Braking + going straight	$D_1$	0.5	[0, 1]
Braking + turning left	$D_2$	1.5	(1, 2)
Braking + turning right	$D_3$	2.5	[2, 3]

$D_1$  refers to braking + going straight, which means that the driver drives along the current lane while braking. The abnormal target will be injured. Personnel of the host vehicle and the adjacent lane vehicles will be safe;  $D_2$  refers to braking + turning left, which means that the driver turn left

while braking. The cockpit of the host vehicle will hit the vehicle in the left lane. The personnel of the host vehicle and the left side vehicle will be injured. The vehicle in the right lane and the abnormal target will be safe;  $D_3$  refers to the braking + turning right, which means that the driver turns right while braking, hitting the vehicle in the right lane. Personnel of vehicle in the right lane will be injured, and the host vehicle and the abnormal target will be safe. The DDM is described as shown in Figure 5.

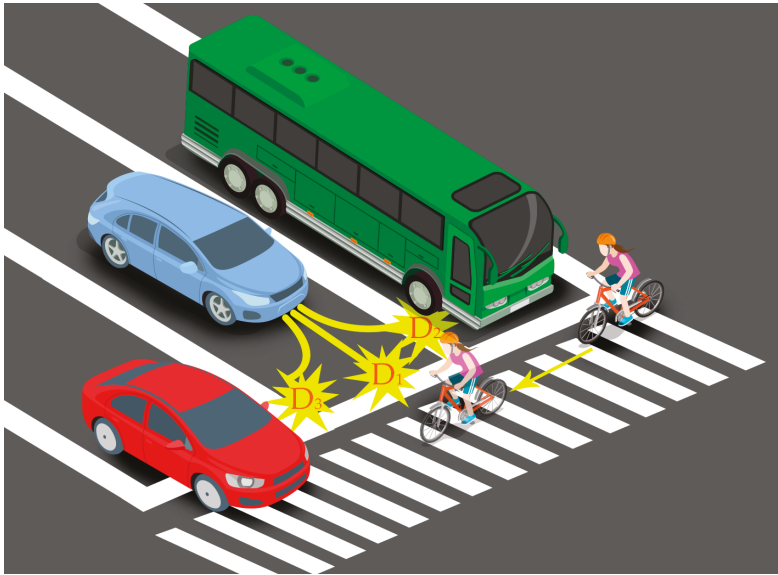


Figure 5. Three feasible driving decision-making.

## 5. Verification and Analysis of DDM

After establishing the model of seven input variables and one output variable, we need to use a driving simulation experiment to provide samples to train and verify DDM.

### 5.1. Virtual Driving Experiment

#### 5.1.1. Scenes and Equipment of Experiment

After establishing the model of seven input variables and one output variable, we need to use a driving simulation experiment to provide samples for the model, to train and verify the factors.

UC-win/Road software is used to establish the virtual emergency situations. Virtual scene can access FORUM8.0 driving simulator which is a real vehicle control unit and the sample collection frequency is set to 20 Hz. Driving simulator can be perfectly compatible with the virtual scene to present the driver real driving experience. At the same time, the experimental equipment can synchronously output more than 80 types of parameters such as target speed and steering wheel angle in the scene. On the one hand, using the UC-win/Road data recording module, the driver's operation process is recorded, which facilitates the experimental screening. On the other hand, it has ability to output more than 80 types of recorded items to record the physical status information of the vehicle and the surrounding objects when the emergency situations occur.

Based on the ten different scenarios listed in Table 1, the input variables  $X_2$ ,  $X_3$ ,  $X_4$  and  $X_5$  are used as the basis for the division of experimental scenes. Furthermore, the dimension of  $X_3$  is taken into account, thus, the  $2 \times 10 = 20$  scene is set up, which corresponds to ten kinds of situations analyzed by Section 2.3. After data processing, data of input variables  $X_1$ ,  $X_6$  and  $X_7$  can be obtained.

### 5.1.2. Staffs of Experiment

In the driving simulation experiment, 45 experienced drivers are recruited from those who participated in the questionnaire. Among them, there are 35 male drivers, 10 female drivers. Driving ages range from 2 to 10 years. In addition, the total driving mileage is more than 5000 km. The experimental staff are both physically and mentally healthy and energetic.

### 5.1.3. Process of Experiment

The volunteers firstly drive freely in the urban road network according to the traffic rules and will test after being familiar with the experimental equipment. After each scene begins, the driver drives normally. When the volunteers are about to arrive at the intersection, the control staff can control the abnormal targets and the adjacent lane vehicles, and artificially create the emergency situations. The driver makes decisions and then performs decisions until the collision with any target in face of the emergency situations, which is a set of experimental samples. In addition, each driver will participate in ten scenes. Therefore,  $10 \times 45 = 450$  groups of experimental data are collected. The driving simulation experiment process is shown in Figure 6.



Figure 6. The driving simulation experiment process.

## 5.2. Process of Experimental Data

### 5.2.1. Statistics of Experimental Data

Taking AT-T as the basis for data statistics, results of DDM are shown in Figure 7.

Among the three kinds of decision-making, the number of braking + going straight is up to 258; accounting for 57.3% of the total, and the number of braking + turning right decisions is 182; accounting for 40.4%; only ten decisions are braking + turning left, accounting for 4.5%. Combined with the experimental questionnaire, the following analysis can be made:

- Most drivers believe that braking + turning left decision will have more severe consequences. Drivers drive on the right side of the road in China. Therefore, the left-turning decision will cause the cockpit to hit the abnormal target directly, which is easy to cause damage to the driver.
- When the abnormal target is pedestrian, there is no significant difference in the number of decisions making between braking + going straight and braking + turning right; as for non-motorized vehicles, braking + turning right decision slightly increased.
- When the target is a pet, in the case of uncertainty about the collision consequences of turning right, a host of drivers will choose to go straight to sacrifice pets avoiding the right side of the vehicle caused by unknown consequences.

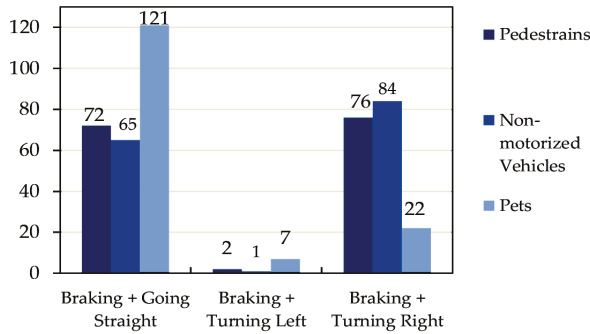


Figure 7. Statistics of experimental data.

5.2.2. Screening of Experimental Data

Human drivers may make mistakes in the execution of decision-making. Some drivers may be nervous, hesitant, and so on, leading to the results of the experiment are not consistent with the driver’s real decision-making in their mind. Therefore, to solve the above problems, some unqualified experimental data need to be corrected or eliminated. The playback function of UC-win/Road 13.0 is enabled to reproduce the previous experiment process, and the volunteer can revise the previous decision to ensure that the decision is consistent with the actual operation. In addition, the volunteers also randomly select the others’ experimental video to rate. A low score sample indicates that the decision is not acceptable to the public. We should eliminate such samples and improve the recognition of the total samples.

5.3. Results of Experiment

After the data processing, we eliminate 30 group data. 420 groups are used as samples for training and testing. 336 groups (80% of the total number of samples) are used as training samples (188 groups of straight samples and 148 groups of right-turning samples), and the remaining 84 groups are taken as test samples (49 groups of straight samples and 35 groups of right-turning samples) to train and verify the proposed DDM.

At the same time, back propagation neural network (BPNN) is used to establish the decision-making model for comparison. The experimental data is recorded by the data output module to train and verify DDM. BPNN is used for comparison to verify the validity of the model. With integrated system, explicit algorithmic process, data identification and simulation function, BPNN is one of most popular learning algorithms with the excellent ability to solve nonlinear problem [22,39]. To verify the performance of TSFNN, a typical feedforward BPNN is established to compare with TSFNN on the relationship between decision influencing factors and driver decision-making [22].

The training results of TSFNN and BPNN are shown in Figures 8 and 9 respectively, and the error comparison results of the two models are shown in Figure 10.

For further comparison, the mean absolute error (MAE) and the root-mean-square error (RMSE) are calculated, respectively. MAE can better reflect the actual situation of the forecast value error, and its calculation formula is as follows:

$$MAE = \frac{1}{m} \sum_{i=1}^m |f_i - y_i| \tag{13}$$

In this formula,  $f_i$  is the predicted value  $y_i$  is the actual value and  $m$  is the sample number.



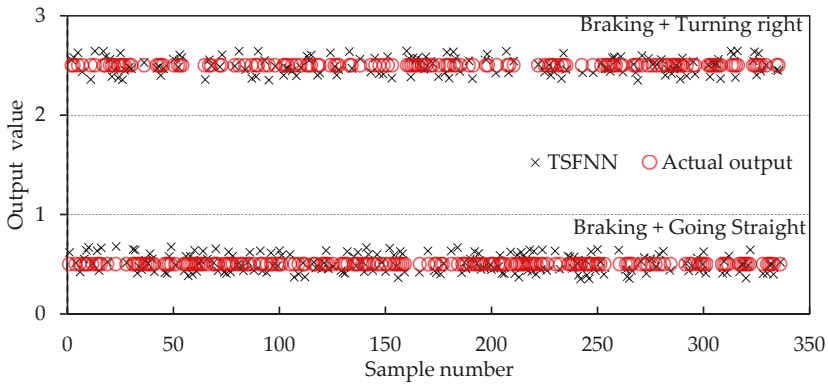


Figure 8. The training results of TSFNN.

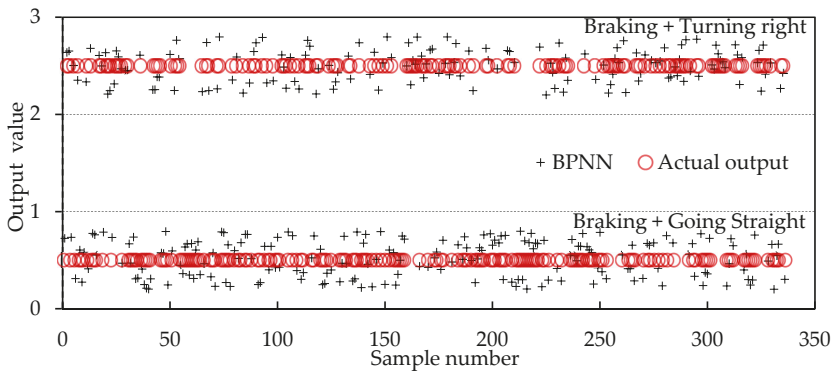


Figure 9. The training results of back propagation neural network (BPNN).

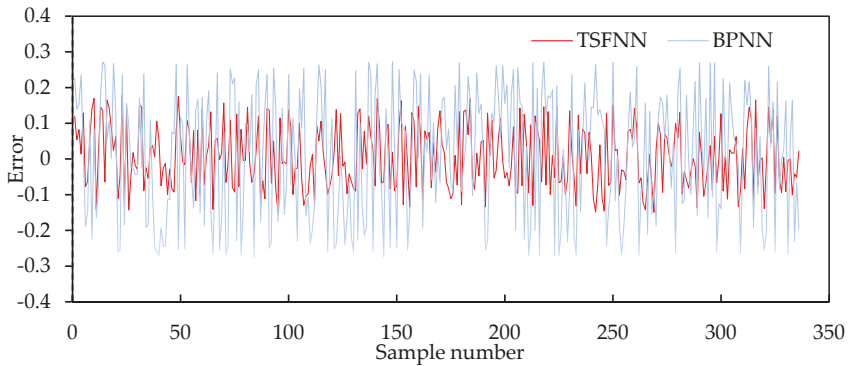


Figure 10. The error comparison results of TSFNN and BPNN.

RMSE is used to measure the deviation between the observed value and the actual value, and the formula is as follows:

$$RMSE = \sqrt{\frac{1}{m} \sum_{i=1}^m (f_i - y_i)^2} \tag{14}$$

The comparison of the two models is shown in Table 6.

The comparison results show that the decision model based on TSFNN has obvious advantages in accuracy and the overall error is smaller than BPNN. Therefore, TSFNN can accurately reflect the relationship between decision-making factors and DDM.

**Table 6.** Comparison of TSFNN and BPNN Prediction results.

	TSFNN				BPNN			
	Training Samples		Test Samples		Training Samples		Test Samples	
	Braking + Going Straight	Braking + Turning Right	Braking + Going Straight	Braking + Turning Right	Braking + Going Straight	Braking + Turning Right	Braking + Going Straight	Braking + Turning Right
<i>MAE</i>	8.08%	8.80%	7.24%	10.77%	12.11%	12.35%	11.43%	15.44%
<i>RMSE</i>	0.0044	0.0053	0.0037	0.0083	0.0099	0.0112	0.0092	0.0146
<b>Maximum absolute error</b>	17.96%	16.51%	19.60%	17.40%	27.18%	27.43%	31.18%	29.52%

## 6. Conclusions

In this paper, a DDM was developed to make accurate driving decision-making for AVs under the emergency situations evoked by red light-running behaviors. The ethical and legal factors which were difficult to describe in DDM were quantitatively characterized. RL, AT-T, AT-N and AT-S indicated ethical and legal factors. Seven main influencing factors of DDM were determined by PCA as the input variable of DDM. The driving decision-making (D) was developed to accomplish the inherent complex and precise tasks, including braking + going straight, braking + turning left, braking + turning right, which was taken into account as the output variable. Consequently, a DDM based on TSFNN was established. Training samples were collected by driving virtual experiments. At the same time, considering the interference of artificial factors, we eliminated the unqualified sample data. TSFNN was trained and compared with BPNN. Experimental results showed that the output results of TSFNN were more accurate, and the error was smaller than that of BPNN, and the quantitative ethical and legal factors could be accurately and successfully estimated by the proposed DDM.

Although this article integrates ethical and legal factors into DDM, there are still some limits. This paper considers the emergency condition evoked by red light-running behaviors at the intersections, but the traffic scene is infinite. We will consider more situations in the future. In addition, in the process of establishing a decision model, the correlative parameters of vehicle dynamics and the transient static decision are not considered. In addition, the influence of the dynamic space state of other objects in the situations on the driving decision is not considered either. The establishment of dynamic and real-time DDM is required in future study.

**Author Contributions:** S.L. and J.Z. conceived and designed the experiments; S.W. and P.L. performed the experiments and analyzed the data; P.L. and Y.L. contributed reagents/materials/analysis tools; S.L. and J.Z. wrote the paper.

**Funding:** This work was financially supported by the National Natural Science Foundation of China under Grant no. 71801144, and the Science & Technology Innovation Fund for Graduate Students of Shandong University of Science and Technology under Grant no. SDKDYC180373.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Pakusch, C.; Stevens, G.; Boden, A.; Bossauer, P.; Rosen, M.A. Unintended effects of autonomous driving: A study on mobility preferences in the future. *Sustainability* **2018**, *10*, 2404. [[CrossRef](#)]
2. Chen, J.; Zhao, P.; Liang, H.; Mei, T. Motion planning for autonomous vehicle based on radial basis function neural network in unstructured environment. *Sensors* **2014**, *14*, 17548–17566. [[CrossRef](#)] [[PubMed](#)]
3. Herrera-Viedma, E.; Chiclana, F.; Dong, Y.; Cabrerizo, F.J. Special issue on intelligent decision support systems based on soft computing and their applications in real-world problems. *Appl. Soft Comput.* **2018**, *67*, 610–612. [[CrossRef](#)]

4. Chen, X.; Tian, G.; Chan, C.Y.; Miao, Y.; Gong, J.; Jiang, Y. Bionic lane driving of autonomous vehicles in complex urban environments: Decision-making analysis. *Transp. Res. Rec. J. Transp. Res. Board* **2016**, *2559*, 120–130. [CrossRef]
5. Lv, W.; Song, W.; Liu, X.; Ma, J. A Microscopic lane changing process model for multilane traffic. *Phys. A Stat. Mech. Appl.* **2013**, *392*, 1142–1152. [CrossRef]
6. Wang, X.; Yang, X. Study on decision mechanism of driving behavior based on decision tree. *J. Syst. Simul.* **2008**, *20*, 414–415.
7. Castaño, F.; Beruvides, G.; Villalonga, A.; Haber, R. Self-tuning method for increased obstacle detection reliability based on internet of things LiDAR sensor models. *Sensors* **2018**, *18*, 1508. [CrossRef] [PubMed]
8. Bonnefon, J.F.; Shariff, A.; Rahwan, I. The social dilemma of autonomous vehicles. *Science* **2016**, *352*, 1573–1576. [CrossRef] [PubMed]
9. Blackburn, S. Trolley problem. *Yale Law J.* **2016**, *94*, 1395–1415.
10. Martin, R.; Kusev, I.; Cooke, A.J.; Baranova, V.; Schaik, P.V.; Kusev, P. Commentary: The social dilemma of autonomous vehicles. *Front. Psychol.* **2017**, *8*, 808. [CrossRef] [PubMed]
11. Greene, J. Our Driverless Dilemma. *Science* **2016**, *352*, 1514–1515. [CrossRef] [PubMed]
12. Hevelke, A.; Nida-Rümelin, J. Responsibility for crashes of autonomous vehicles: An ethical analysis. *Sci. Eng. Ethics* **2015**, *21*, 619–630. [CrossRef] [PubMed]
13. Hauser, M.; Cushman, F.; Young, L.; Jin, K.X.; Mikhail, J. A dissociation between moral judgments and justifications. *Mind Lang.* **2007**, *22*, 1–21. [CrossRef]
14. Barman, B.; Kanjilal, R.; Mukhopadhyay, A. Neuro-Fuzzy controller design to navigate unmanned vehicle with construction of traffic rules to avoid obstacles. *Int. J. Uncertain. Fuzziness Knowl. Based Syst.* **2016**, *24*, 433–449. [CrossRef]
15. Thornton, S.M.; Pan, S.; Erlien, S.M.; Gerdes, J.C. Incorporating ethical considerations into automated vehicle control. *IEEE Trans. Intell. Transp.* **2017**, *18*, 1429–1439. [CrossRef]
16. Goodall, N. Ethical decision making during automated vehicle crashes. *Transp. Res. Rec. J. Transp. Res. Board* **2014**, *2424*, 58–65. [CrossRef]
17. Rahwan, I. Moral Machine. Available online: <http://moralmachine.mit.edu/> (accessed on 28 May 2017).
18. Sadeghian, R.; Sadeghian, M.R. A decision support system based on artificial neural network and fuzzy analytic network process for selection of machine tools in a flexible manufacturing system. *Int. J. Adv. Manuf. Technol.* **2016**, *82*, 1795–1803. [CrossRef]
19. Azadeh, A.; Saberi, M.; Anvari, M.; Mohamadi, M.A. Integrated artificial neural network-genetic algorithm clustering ensemble for performance assessment of decision making units. *J. Intell. Manuf.* **2011**, *22*, 229–245. [CrossRef]
20. Efeendigil, T.; Önüt, S.; Kahraman, C. A decision support system for demand forecasting with artificial neural networks and neuro-fuzzy models: A comparative analysis. *Expert Syst. Appl.* **2009**, *36*, 6697–6707. [CrossRef]
21. Wang, X.; Wang, J.; Zhang, J.; Ban, X. Driver's behavior and decision-making optimization model in mixed traffic environment. *Adv. Mech. Eng.* **2015**, *7*, 759571. [CrossRef]
22. Zhang, J.; Liao, Y.; Wang, S.; Han, J. Study on driving decision-making mechanism of autonomous vehicle based on an optimized support vector machine regression. *Appl. Sci.* **2018**, *8*, 13. [CrossRef]
23. Campos, A.C.S.M.; Mareschal, B.; Almeida, A.T.D. Fuzzy Flowsort: An integration of the flowsort method and fuzzy set theory for decision making on the basis of inaccurate quantitative data. *Inform. Sci.* **2015**, *293*, 115–124. [CrossRef]
24. Jan-Jaap, V.D.B.; Sandell, M.; Borjesson, P.O. ML estimation of time and frequency offset in OFDM systems. *IEEE Trans. Signal Process.* **1997**, *45*, 1800–1805.
25. Chun, M.G.; Bien, Z. Fuzzy petri net representation and reasoning methods for rule-based decision making systems. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **1993**, *76*, 974–983.
26. He, K.; Sun, H.; Cheng, W. Application of fuzzy neural network based on t-s model for mobile robot to avoid obstacles intelligent control and automation. In Proceedings of the World Congress on Intelligent Control and Automation (WCICA), Chongqing, China, 25–27 June 2008; pp. 8282–8285.
27. Wang, P. Improved T-S Fuzzy Neural Network in Application of Speech Recognition System. *Comput. Eng. Appl.* **2009**, *45*, 246–248.

28. Zhang, Q. Intelligent vehicle fuzzy neural network—Based obstacle avoidance intelligent vehicle fuzzy neural network—Based obstacle avoidance. *Agric. Equip. Veh. Eng.* **2015**, *53*, 33–35.
29. Nyholm, S.; Smids, J. The ethics of accident-algorithms for self-driving cars: An applied trolley problem? *Eth. Theory Moral Pract.* **2016**, *19*, 1275–1289. [CrossRef]
30. Louis. German Autonomer: Morality Police. Available online: <http://louisproyect.wordpress.com/2011/12/06/german-autonomer-morality-police/> (accessed on 12 June 2011).
31. Wu, C.; Yao, L.; Zhang, K. The Red-Light Running Behavior of Electric Bike Riders and Cyclists at Urban Intersections in China: An Observational Study. *Accid. Anal. Prev.* **2012**, *49*, 186–192. [CrossRef] [PubMed]
32. Li, Y.; Shao, Z. The essence, legal basis and principle of Traffic accident responsibility determination of public security organs. *J. Shanghai Police Coll.* **2016**, *26*, 57–65.
33. Zhao, J.; Shao, X.; Zhao, L.; Zhao, S.; Niu, X. Driving behavior theory and computer simulation system of driver's risk perception based on 3D. *Procedia Soc. Behav. Sci.* **2013**, *96*, 1686–1695. [CrossRef]
34. Du, G.; Jiang, Z.; Diao, X.; Ye, Y.; Yao, Y. Variances handling method of clinical pathways based on T-S fuzzy neural networks with novel hybrid learning algorithm. *J. Med. Syst.* **2012**, *36*, 1283–1300. [CrossRef] [PubMed]
35. Hyun, C.H.; Park, C.W.; Kim, S. Takagi–Sugeno fuzzy model based indirect adaptive fuzzy observer and controller design. *Inform. Sci.* **2010**, *180*, 2314–2327. [CrossRef]
36. Wei, S. Driving Fatigue Fusion Detection Based on T-S Fuzzy Neural Network Evolved by Subtractive Clustering and Particle Swarm Optimization. *J. Southeast Univ.* **2009**, *25*, 356–361.
37. Urmson, C.; Anhalt, J.; Bagnell, D.; Baker, C.; Bittner, R.; Clark, M.N.; Dolan, J.; Duggins, D.; Galatali, T.; Geyer, C. Autonomous driving in urban environments: Boss and the urban challenge. *J. Field Robot.* **2008**, *25*, 425–466. [CrossRef]
38. Halko, N.; Martinsson, P.G.; Shkolnisky, Y.; Tygert, M. An Algorithm for the Principal Component Analysis of Large Data Sets. *SIAM J. Sci. Comput.* **2011**, *33*, 2580–2594. [CrossRef]
39. Chen, Y.; Yi, Z. The BP Artificial Neural Network Model on Expressway Construction Phase Risk. *Syst. Eng. Procedia* **2012**, *4*, 409–415. [CrossRef]



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# Communications and Driver Monitoring Aids for Fostering SAE Level-4 Road Vehicles Automation

Felipe Jiménez <sup>1,\*</sup>, José Eugenio Naranjo <sup>1</sup>, Sofía Sánchez <sup>1</sup>, Francisco Serradilla <sup>1</sup>, Elisa Pérez <sup>2</sup>,  
María José Hernández <sup>2</sup> and Trinidad Ruiz <sup>2</sup>

<sup>1</sup> University Institute for Automobile Research (INSIA), Universidad Politécnica de Madrid (UPM), 28031 Madrid, Spain; joseeugenio.naranjo@upm.es (J.E.N.); sofia.sanchez@upm.es (S.S.); fserra@eui.upm.es (F.S.)

<sup>2</sup> Psychology Faculty, Universidad Complutense de Madrid, Campus de Somosaguas, Ctra. de Húmera, s/n, 28223 Pozuelo de Alarcón, Madrid, Spain; elisaperez@psi.ucm.es (E.P.); mjhlloreda@gmail.com (M.J.H.); truíz@psi.ucm.es (T.R.)

\* Correspondence: felipe.jimenez@upm.es; Tel.: +34-91-336-53-17

Received: 3 August 2018; Accepted: 27 September 2018; Published: 2 October 2018

**Abstract:** Road vehicles include more and more assistance systems that perform tasks to facilitate driving and make it safer and more efficient. However, the automated vehicles currently on the market do not exceed SAE level 2 and only in some cases reach level 3. Nevertheless, the qualitative and technological leap needed to reach level 4 is significant and numerous uncertainties remain. In this sense, a greater knowledge of the environment is needed for better decision making and the role of the driver changes substantially. This paper proposes the combination of cooperative systems with automated driving to offer a wider range of information to the vehicle than on-board sensors currently provide. This includes the actual deployment of a cooperative corridor on a highway. It also takes into account that in some circumstances or scenarios, pre-set or detected by on-board sensors or previous communications, the vehicle must hand back control to the driver, who may have been performing other tasks completely unrelated to supervising the driving. It is thus necessary to assess the driver's condition as regards retaking control and to provide assistance for a safe transition.

**Keywords:** automated driving; cooperative systems; communications; interface; automated-manual transition; driver monitoring

## 1. Introduction

The solution to the major problems associated with road traffic seems to lie, at least to a significant extent, in automated and connected driving. In particular, the automated vehicle endeavours to have an impact on the high percentage of accidents, with the human factor being the main cause. Thus, according to statistical studies, two thirds of accidents are due exclusively to human failures while this factor is present in 90% of accidents [1]. The reasoning behind the advantages of the automated vehicle starts by eliminating this factor as much as possible. Additionally, congestion problems could be reduced if better information is available, which is the basis of the connected vehicle. Hence, breakthroughs in automated driving technologies herald a much safer and highly efficient future for transportation [2–4].

The idea of an automated vehicle is almost as old as the vehicle itself. Between the late 80s and the early 90s, a project was developed in Europe that was crucial in terms of establishing the bases of intelligent vehicles: the PROMETHEUS project (Programme for a European Traffic of Highest Efficiency and Unprecedented Safety) [5]. In that framework, several automated prototypes were shown in October 1994 on Highway 1 near Charles-de-Gaulle Airport in Paris. In 1995, a 1600 km trip was completed from Munich (Germany) to Copenhagen (Denmark), reaching a speed of 175 km/h, with 95% of the trip taking place in automated driving mode, under real conditions. A group from the

University of Parma travelled 2000 km in 1996 on Italian highways at an average speed of 90 km/h, with 94% of the trip taking place in automatic mode. The most noteworthy modern projects were CyberCars and CyberMove [6] in the 2000s.

Meanwhile, in 1991 the United States Congress urged the Ministry of Transportation to develop an automated vehicle and an infrastructure suitable for automated driving. The Carnegie Mellon University Laboratory developed 11 automated vehicles and in 1995 travelled 3000 miles with a vehicle that drove in automatic mode 98% of the time; in 1997, a demonstration was conducted with 20 vehicles on I-15 in San Diego. A major milestone was the organization by the Defense Advanced Research Projects Agency (DARPA) of 3 competitions for automated vehicles without a driver [7]. In the first race, in 2004 in the Mojave Desert, no car was able to finish the race. In 2005, the winning vehicle travelled 212 km and was not the only one to reach the finish line. In 2007 the competition moved to an urban environment, including street crossings.

However, the point at which these vehicles became widely known dates mainly to when Google announced their development. From that point, numerous initiatives and projects have emerged; manufacturers are incorporating more and more automation functions in their vehicles. However, this has highlighted the challenges that must be addressed, not only from a technological point of view.

The most widely used classification system for automated driving was developed by SAE [8] and includes 5 levels. Up to level 2, monitoring of the environment is left to the driver, although the task is already shared with the vehicle in level 2. In level 3, the vehicle takes on the task of monitoring the environment and in level 4 it also oversees the driving itself so that the driver can perform other tasks. The main difference between levels 4 and 5 is that, in the former, automation is only operative in some scenarios, so that, at some point, automatic-manual transitions take place with sufficient warning, whereas in level 5, this automation extends to all scenarios.

Despite the technological progress made in recent years, fully automated driving still faces several hurdles.

One of the main limitations is that many automated vehicles today only use on-board sensors to perceive the environment and these may not be robust enough for some driving conditions. As on-board sensors can only obtain current information about the immediate surroundings, these automated vehicles often have difficulty reliably anticipating the motion of vehicles close by [9]. Therefore, many disengagement incidents happen when an automated vehicle has to interact with nearby human-driven vehicles [10]. In order to facilitate the implementation of automated vehicles in real traffic, it is desirable to introduce beyond-line-of-sight information through cooperative systems [11–13]. They increase data redundancy for highly automated vehicles [14].

The Cooperative Systems (C-ITS) are based on the generation of safety and efficiency information in road transport and its diffusion through the use of Vehicle-to-X (V2X) communications networks. Several C-ITS experiments have been developed, such as [15] and [16], which were mainly focused on sending traffic information to vehicles using Vehicle-to-Infrastructure (V2I) communications, in such a way that the same information shown in the variable information panels is presented to the driver inside the vehicle through a Human Machine Interface (HMI). Also, there are experiments such as those of [17,18], where there is an exchange of information between some vehicles and others using vehicle-to-vehicle (V2V) communications services. In this way, the limit of the visual horizon of the vehicles is overcome and more comprehensive information is obtained regarding the driving environment with variables that may affect safety.

At present, the European ITS Platform has published the first two sets of cooperative services [19] that are currently available for deployment and have been named Day 1 and Day 1.5, in reference to their implementation terms. In this way, within the C-ITS Day 1 services we can find Emergency vehicle approaching (V2V), Hazardous location notification (V2I), Road works warning (V2I) and Weather conditions (V2I). Within C-ITS Day 1.5 we find Cooperative collision risk warning (V2V) and Zone access control for urban areas (V2I). These services have been extended and analysed in depth in Reference [20]. This set of services is currently being deployed in projects such as C-ROADS

(<https://www.c-roads.eu>), where C-ITS corridors are enabled, such as the highway corridor extending from Rotterdam (Netherlands) via Frankfurt (Germany) to Vienna (Austria).

However, it is clear that automated driving in the strict sense, at an SAE-3 or higher automation level [8], does not make sense if cooperative and connectivity components are not incorporated within the ego-vehicle, since the horizon limitations of visual sensors on the vehicles means they have the same limitations as human beings. Thus, a series of projects is currently being developed in the field of automated and connected driving, such as the MAVEN European projects (<http://www.maven-its.eu/>) and AUTOPILOT (<http://autopilot-project.eu/>).

Connected vehicle technology, such as vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) communication, may be able to resolve some of the existing problems resulting from heavy reliance on a priori information. V2I communication could provide a network for intersections, road signs and construction signs to transfer important infrastructure information, such as road layout changes, speed limits and traffic light information to autonomous vehicles [21,22]. Similarly, V2V communication allows vehicles to share data [23,24]. By integrating V2V and V2I communication with autonomous vehicle technology, an effective “cooperative driving” network can be established [21,25].

Some scenarios and applications in which cooperative systems can improve autonomous vehicle operation are cooperative adaptive cruise control (CACC) [26], intersections [27] and platooning [28,29]. Furthermore, they can be used for positioning and updating digital map information with dynamic data [30]. An example of these early works on their impact is a study on flow control and congestion management in Automated Highway Systems [31]. More recently, some real test analyses and simulations have shown that safety and energy efficiency can be significantly improved for the connected automated vehicle as well as for the neighbouring human-driven vehicles (e.g., [32–34]).

Another relevant issue is the fact that automated driving changes the role of the driver and it should be considered when a new SAE level is reached [35]. In some of the worst accidents involving autonomous vehicles the driver was not properly supervising the driving environment or the driving task. In addition, manufacturers and researchers have not sufficiently addressed how to handle scenarios in which automated systems fail or cannot manage after drivers have become dependent on automation. Some studies consider that the attentional state of a driver who has full confidence in a self-driving system is similar to that of a passenger riding in a vehicle driven by another person [36]. Driver monitoring is becoming crucial as it has been shown that automation aids could lead to distractions and high reaction times in case of emergency.

Previous research showed that drivers transitioned to dependence on automation after only 15 min of autonomous driving, as evidenced by eye movement, braking preparation behaviour and arousal level [37,38]. In recent years, some studies have analysed whether drivers can successfully control vehicles after a system failure based on a range of biometric data [39]. Drivers’ mental workload is relatively low during the autonomous-driving scenario because they do not experience any stress from driving. However, drivers’ systolic blood pressure increases because of the transition from autonomous driving to manual control and because of the mental workload involved in controlling the vehicle on their own just after using an autonomous-driving system. From the viewpoint of the brain activity in the left frontal lobe, the data indicates that drivers’ cognition level during autonomous driving is lower than during manual driving. The eye-gaze data indicate that “mind distraction” occurs in participants while resuming control after a system failure because their brain activity is relatively low at that moment. The final results indicate that drivers who depend on autonomous control systems experience stress upon switching to manual control after a system failure.

The study [40] provides a taxonomy of different forms of autonomous vehicle handover situations. Scheduled, emergency and non-emergency handovers are covered and system and driver-initiated handovers are differentiated. Attention is drawn to the fact that the driver is often expected to maintain situational awareness and is hence legally responsible, whereas the handover situation may make this difficult or impossible and the driver may not respond appropriately. In this field, academic

examinations of transitions are becoming more frequent (e.g., [41,42]) [43] focus on driver skills and have developed an approach for safe transitions based on characterizing a driver's skills).

As cited above, eye tracking is one of the commonly used variables to monitor driver behaviour, studying indicators/variables for mental load assessment and gaze direction [44,45], among other things. Indexes such as saccadic eye-movements and duration of eye-blinks are useful for evaluating the attention of drivers while riding in Level 3 autonomous vehicles [36].

One of the most advanced driver assistance systems (ADAS) that compiles data on the driver's condition analyses somnolence using the percentage of eye closure (PERCLOS) [46] based on studies that suggest around 20% of all traffic accidents are related to fatigue [47]. Authors such as [48,49] evaluate the driver's condition by recording the face and head in a simulator and in a real vehicle. Another method to study PERCLOS is to capture biomedical signals instead of images, such as brain, muscle and cardiovascular activity [50–52].

Other authors argue that a good method to estimate mental workload is brain activity. Some ocular indexes can show variations in mental activity related to the task being performed, the eye being an extension of the brain [45]. Cognitive load (also referred to as mental workload) is commonly defined as the relationship between the cognitive demands placed on a user by a task and the user's cognitive resources [53] and can be estimated using performance, physiological and subjective measures. Pupil dilation [54], heart-rate variability and galvanic skin response are examples of physiological measures whose changes are related to variations in cognitive load levels [55]. The most popular eye-movement indices for mapping mental workload are pupil diameter and blink rate, as shown in the literature [56–58].

Finally, other studies focus on the design of vehicle driver interfaces [59]. Both automated driving phases as well as transition phases where the driver has to reengage with the driving task should be considered [60]. Concerning interfaces designed to transmit the road situation to the driver, haptic signals and vibrotactile stimuli have been studied in a variety of driving applications such as lane-keeping assistance [61], blind spot warning [62] and rear collision warning systems [63].

In summary, although much progress has been made in introducing automated driving on real roads, several challenges are still in the research agendas in order to achieve the highest automation levels in the shortest period of time.

In this paper, two main aids are presented to foster higher levels of vehicle automation. On the one hand, the architecture of vehicular communications is considered essential for high levels of automation in order to increase the vehicle's knowledge of the environment, thereby improving the range and capabilities of on-board perception sensors. A cooperative corridor along a real road is presented in order to obtain synergies of both C-ITS and the automated and connected vehicles, thereby enabling Automated Cooperative Driving. In this way, all the architecture designed to allow Traffic Management Centers (TMC) to generate C-ITS messages and transmit them to the road via V2X communications is explained and tested. Likewise, automated vehicles that circulate along this route have been adapted so that they are able to receive and process the C-ITS information, acting accordingly in the most appropriate manner possible. In this way, automated vehicles are able to take advantage of the cooperative environment, deployed and standardized at present, by including in their guidance systems the information received by this channel, which is managed and generated from the TMCs. This cooperative architecture follows current standards but uses some specifically developed communication modules. The main difference with other deployments is the interaction with autonomous vehicles in order to remotely control driving lane or speed in each scenario and road stretch.

Additionally, a study is presented on how driver awareness could be assessed when the automated vehicle should hand back control to the driver and the design of proper interfaces for facilitating this driver task of taking back vehicle control. This transition is critical for safety because long periods of inattention to driving tasks could make it difficult to retake control in a short time, especially when complex manoeuvres are required immediately after this changeover. As previously stated,



some research has been done and difficulties in achieving correct driver responses have been discussed. Furthermore, many studies propose variables to assess driver awareness but little work has been done on proposing methods to foster adequate driver psychophysical conditions in control transitions. Although there are some simple solutions to performing this control changeover, their efficiency should be tested with real drivers in order to implement the most reliable ones and thus make sure that drivers are prepared to take control in a short period of time. Finally, driver behaviour is analysed in specific scenarios after regaining vehicle control. Specifically, the merging manoeuvre is analysed because of its complexity and it is a common scenario that could appear after a control changeover.

Both analyses are necessary for achieving SAE level 4 and perhaps they are some of the topics that are more disruptive in comparison with previous levels

## 2. C-ITS Development

### 2.1. C-ITS Architecture

C-ITS architecture has been designed to guarantee the flow of information from the TMCs to the vehicles and vice versa, using mainly V2I communications. In this way, the TMCs can send the information provided by the cooperative services to the vehicles, which will inform the driver or carry out automated actions, depending on whether the vehicles are connected or automated and connected.

The complete value chain of the C-ITS has been developed, with the implementation of all the elements necessary to connect the TMCs with the vehicles, including the interfaces for the collection of information from various sources, intelligent processing of the latter and the transmission of C-ITS messages to the road through different means of communication.

Figure 1 shows the complete architecture where the different functional modules are represented. In this way, the C-ITS module is integrated within the TMC of the Spanish Traffic Authority (DGT) and its actions can be manually configured as a further module. Internally, it has an Information Collection Module that obtains internal data from the TMC (through the DATEX II protocol), as well as from other public sources, such as municipalities and enforcement services. This collected information serves as input to the Information Analysis Module, which is able to process it in such a way that it serves as an input to the decision-making module, the latter being responsible for deciding whether it is appropriate to generate some type of C-ITS message for the services that are enabled at that moment and for a geographical area that has Road Side Units (RSU) deployment. This module generates standardized C-ITS Decentralized Environmental Notification Message (DENM) type messages that are sent via an ad-hoc protocol to the corresponding Sending type RSUs, which are able to connect with the TMC via Infrastructure-to-Infrastructure (I2I) communications. These RSUs decode the message from the control centre and transmit the corresponding DENM packet to the specific geographical area, either directly or through another RSU operating in Relay mode, using the ETSI ITS G5 protocol, standardized by the European Telecommunications Standards Institute (ETSI) for Intelligent Transport Systems (ITS). This message is received by the connected vehicles, either with a driver or driverless, through their On Board Unit (OBU), which is also connected to the ETSI ITS G5 network.

In this way, the architecture covers all the elements involved, both in the generation of C-ITS services and in V2X communications, all of which are necessary to enable this type of capability on the road.

### 2.2. Deployment at the Test Site

Once the architecture was designed, a deployment of the ETSI ITS G5 communications systems was carried out on one of the motorways entering the city of Madrid, the A6 motorway, which is part of the hub of the urban accesses of the TEN-T Atlantic Pan-European corridor. This test site also has a reversible lane in the centre of the road, which can be opened or closed on demand directly from the TMC. This lane allows testing without the need to share the road with other vehicles, in order to more safely validate the developments but it can also be used with open traffic.

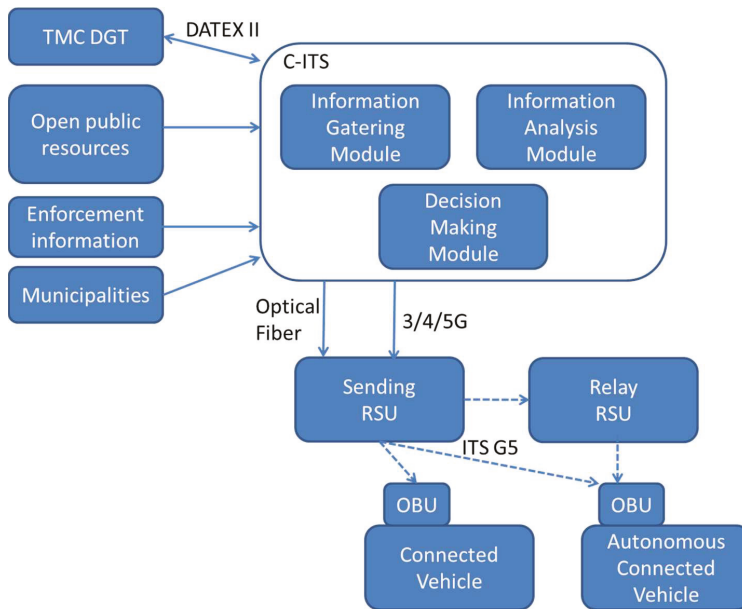


Figure 1. C-ITS functional architecture.

We proceeded with the installation of 18 RSU communications modules along a 16-km stretch of this road, with a separation of between 500 and 1000 m between them. These modules have been distributed according to their capabilities. 3 RSU Sending units with an I2I connection with the C-ITS control centre and 15 RSU Relay units that act as repeaters of the signal were deployed. This deployment allows ITS G5 connectivity throughout the section, guaranteeing the reception of the information by the vehicles that circulate at any point along the road. The RSU Sending communication modules are from the manufacturer Cohda and the RSU Relay modules are ITS-INSIA [64,65]. These last modules have been specifically developed for this purpose. They are based on an AR9220 chipset that uses the ath9k modified driver to allow 802.11p bands. This card is configured to work in the 5855–5925 GHz bands. The modules use the Debian “wheezy” operating system, kernel version 3.19.0. The kernel has been configured to allow OCB mode (Outside the Context of a BSS) mode and the ath9k driver. The module includes the Global Navigation Satellite System NV08C-CSM chipset (NVC). It is an integrated GLONASS + GPS + GALILEO + SBAS satellite navigation receiver for use in various applications demanding low cost, low power consumption and uncompromised performance. Figure 2 shows a Relay RSU and the deployment of a Sending RSU on a variable message panel on the road.



Figure 2. Relay RSU (left) and installation of Sending RSU on a variable message panel (right).

A complete C-ITS use case has been designed to demonstrate the successful performance of the architecture. It covers the integration of all subsystems that are part of the value chain of cooperative services. Thus, it is allowed to activate an incident in the TMC that generates a C-ITS message, which was received by the test site RSUs and geo-broadcast to all vehicles in the area with an OBU. Specifically, this message was received by an automated vehicle that performs the appropriate actions upon receipt of the new information or hands back control to the driver.

### 2.3. Automated and Connected Vehicles

To carry out the validation of the architecture, 2 connected vehicles were used, one manually driven and the other driverless. The connected vehicle is a Peugeot 307, equipped with a Cohda OBU connected to a tablet with a user interface that generates audible and visual warnings based on the information and alerts generated by the C-ITS service (Figure 3).



Figure 3. Human Machine Interface for the C-ITS driver warning messages.

Meanwhile, to validate the architecture in driverless vehicles, an automated vehicle provided by INSIA was used. All its actuators (accelerator, brake and steering) are automated and are controlled by a low-level layer that is capable of receiving orders from a high-level guidance system. The care is based on the Mitsubishi iMIEV platform (Figure 4) [66].

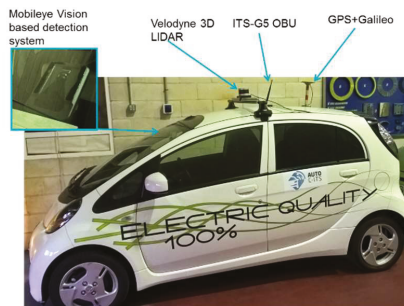


Figure 4. Detail of the INSIA automated vehicle used in the validation of the C-ITS Architecture.

This system receives the information provided by the on-board sensors and processes them, together with the information from an HD digital map, in order to generate driving behaviour similar to that of humans. This vehicle is equipped with a high-precision GPS/Galileo receiver, an inertial unit (INS) for positioning, a Mobileye perception system for obstacle detection, road lines and traffic signals and a Velodyne Lidar system for obstacle detection. Additionally, ITS-INSIA V2X OBU was added for the reception of C-ITS messages.

Likewise, the guidance system was expanded with a C-ITS information processing module, which is capable of both interpreting the information received through V2X and processing it to

adapt the behaviour of the vehicle while performing automated guidance. 2 main scenarios were tested: (1) Automated vehicle speed adaptation as it received a new target via C-ITS communications; (2) Automated lane change when a C-ITS message informs the vehicle that a lane is closed some distance ahead. In both cases, the automated vehicle supervised the environment using the on-board sensors and performed the manoeuvre safely.

### 3. Transition between Automated and Manual Driving Modes

Although up to automation level 2 the driver is responsible for the driving task and in level 3 he retains responsibility for supervision of this task or the environment, in level 4 the vehicle is responsible for these tasks so that the driver can perform other activities. However, unlike level 5, at level 4 the vehicle may return control to the driver. Therefore, it is necessary to establish methods to facilitate a safe transition as quickly as possible. To do this, the driver's activation level must be analysed and the driver must be able to take back the driving task. It is usually verified that a driver is awake attending to a motion measure, for example, by pressing a button on the vehicle. However, this action may not be reliable enough to indicate that the driver is ready to drive safely so other additional measures are proposed and tested.

#### 3.1. Study Approach

The aim is to evaluate drivers' alertness in a semi-automated driving simulation situation. The tests were carried out in a driving simulator, with one large screen where the driving scenario is projected and a driving position with steering wheel and pedals, as shown in Reference [67]. Different measures were used to check the efficiency and suitability of the handback of the driving task to the driver, that is, when the driving goes from being automated to manual. The experiment was carried out by 21 participants (8 were men and 13 women), with an average age of 23.95 years (SD = 5.72) and an average of five years of driving experience (SD = 5.11). All of them were drivers with normal vision or normal vision corrected. In order to evaluate participants' activation level, an ocular registration system was used. The Model 504 Ocular system log (ASL, Billerica, MA, USA) is a remote unobtrusive eye tracking system designed to measure the diameter of the pupil and coordinate where the user is looking. The eye movement camera launches an infrared beam. Through the reflection of the pupil and cornea, it determines where the subject is looking. The vertical and horizontal position of the eye, pupil diameter and 16-bit external data were recorded at a frequency of 50 Hz. Along with these eye movement data, the system returns an image of the subject's field of vision and their glance superimposed by a cursor in real time. The system has an accuracy of  $0.5^\circ$  of visual angle. In addition, a steering wheel and pedals connected to the system were used to study the driver's behaviour. A unidirectional microphone was used as a vocal key when recording the response time to the cognitive task. Figure 5 shows the laboratory with the simulator and the equipment used.



Figure 5. Driver simulator (left) and equipment for assessing driver activation level (right).

Three tasks performed by the participant to verify their activation level: (1) motor task: press a button; (2) fast and simple cognitive task, which could be arithmetical or verbal (perform an addition/subtraction or read a word); (3) execution task in driving (braking before a STOP signal that appears after the participant starts the simulated driving).

The different variables used to measure participants' activation levels were: (1) Time it takes to fix the eye on the road when listening to an audible warning (in ms); (2) Reaction time for the motor task (in ms); (3) Reaction time for the cognitive task (in ms); (4) Size of the pupil (in pixels) and (5) Time it takes to brake when the stop sign appears on the road (in ms).

The size of the pupil was recorded from the beginning of the cognitive task to ensure sufficient time for the pupil to accommodate to the light (until the sound signal ends, 2000 ms) and until the appearance of the stop sign, measured as the moment at which the screen illumination changes.

Two situations of driver activation (initial conditions) to which the participants were induced were established before each trial to achieve their distraction: situation of cognitive activation (reading or being distracted with their smartphone) or relaxation (with their eyes closed in a comfortable position). Each type of prior activation situation was combined with the two types of cognitive task, prior to taking control of driving: verbal (reading a projected word on the screen) and arithmetical (performing a simple addition or subtraction), leading to four experimental conditions in a repeated measures design. Each participant performed four trials, one per experimental condition. The order of appearance of the different conditions or types of trial was balanced across participants.

In the experimental process, after 5 min on average (randomized between 1 and 9) a warning is presented indicating that the participant must begin the driving task. After this stimulus, each participant performs two tasks to check their activation level (press the button and solve the cognitive task aloud). The word, addition or subtraction that appeared in each trial is randomly selected from a set of 72 different stimuli for each type of task. After a period of driving in the simulator, a stop sign appears before which the driver must react. Table 1 shows the sequence of the driver's tasks.

**Table 1.** Driver tasks.

Task	Time Interval (min)
Reading or relaxing (eyes closed)	1–9
Beep sound	
Press button	
Cognitive task	
Driving task	0.1–0.5
Stop signal	

### 3.2. Results

The test results were as follows:

- For the variable *Time that it takes to fix the gaze in the area of interest for driving*, a means difference contrast was made. There is an effect of the activation situation, in the sense that it takes longer to look when the participant is relaxed with his eyes closed (1071 ms) than when he is distracted by the mobile (881 ms),  $t_{16} = 2.602$ ,  $p = 0.019$ ,  $d = 0.48$ .
- For the variable *Size of the pupil*, an ANOVA of repeated measures  $2 \times 2 \times 2$  was performed (Activation situation  $\times$  Type of cognitive task  $\times$  Experimental moment). An effect of the experimental moment was found, with  $F(1, 8) = 27.09$ ,  $p = 0.001$ , partial  $\eta^2 = 0.77$ . When the participant was solving the cognitive task, his pupil diameter was greater than during driving (average values of 35.3 and 31.1 pixels, respectively). No differences were found either by activation situation or by type of task. These results seem to indicate that both cognitive tasks (verbal vs. arithmetical) entailed the same cognitive load for the participant. The results can be seen in Table 2.

- For the *Reaction Time to press the button*, a comparison of related means is also made and statistically significant differences are found for the Activation Situation factor, with  $t_{20} = -2066$ ,  $p = 0.05$ ,  $d = 0.4$ . When a participant is in a situation of activation, reading or distracted with his smartphone, his reaction time to this task is greater than if he is in a relaxed situation, (1492 ms vs. 1306 ms), that is, responding on average almost 200 ms later. One possible explanation for this result is that the reaction time is longer because you take your mobile phone in your hand before answering.
- For the variables Reaction time for the cognitive task and Reaction time to the stop signal, an ANOVA of repeated measures  $2 \times 2$  is performed. Regarding the *Reaction time for the cognitive task*, no effects or Situation are found of activation or the type of cognitive task on the reaction time. However, when the situation is reading, there seem to be indications that the answer to the arithmetical versus the verbal task is slower (2197 vs. 1599 ms, respectively,  $d = 0.7$ ).
- There are also no statistically significant differences for the two effects studied on the *reaction time to the stop sign*; however, and in the same way, if participants had been in an activation situation and had performed an arithmetical task they took on average 198 ms more to brake before the stop sign than those who had also been in an activation situation but had read a word ( $d = 0.38$ ).

**Table 2.** Results of pupil size (pixels).

		Cognitive Task	
		Verbal	Arithmetic
Initial driver situation	Reading	34.8	37.1
	Relaxing	34.1	35.3

Driving baseline: 31.1 pixels.

### 3.3. Discussion

According to the results obtained, the time it takes for a participant to press a button after an audible warning that he must take control of the vehicle is greater when he is distracted reading his smartphone than when he is relaxed, with his eyes closed. However, the time it takes to attend to the driving screen is greater when he is in a relaxed situation compared to the situation of prior activation or cognitive distraction. Both situations are equally plausible in a semi-automated real driving scenario. Probably, it is interfering with the fact that the driver must first release his cell phone to perform the motor task. This result could be easily extrapolated to reading a book, talking on the mobile phone and so forth. However, he seems to respond faster to the road when he is previously in an activation situation. This result highlights the need to complement a motor measurement (e.g.: pressing a button) with an eye-type measurement.

In addition, pupil dilation has been sensitive to changes in cognitive load, as reflected in the results when the participant performs a cognitive task before a driving simulation. A system of ocular recording must detect that the driver looks at the road in time and the change in his pupil dilation when he also performs a cognitive task.

Two types of cognitive tasks have been tested, with the intention of keeping their load equal and low. The results in the diameter of the pupil do not allow us to draw the conclusion that one task reflects more workload than the other. However, being distracted or in a situation of prior activation can interfere with the arithmetical task in two ways: (1) it takes longer to respond to the task itself of evaluating the activation if it is arithmetical (longer reaction time to the cognitive arithmetical task versus a verbal one) and (2) it seems that in these circumstances driving performance is also affected (greater average reaction time to the stop signal when the driver was in a situation of prior activation and performed an arithmetical task). For these reasons, the verbal task of reading a word is selected against the task of performing an arithmetical calculation to assess drivers' level of alertness.

#### 4. Driver Behaviour during a Safe-Critical Manoeuvre When Regaining Vehicle Control

Finally, after regaining control of the vehicle, the driver may have to manage a situation that requires a higher than usual attentional load, such as merging into another lane or changing speed. In this regard, whether in fact, such situations cause greater cognitive load is evaluated, so some assistance system might be advisable.

More specifically, the manoeuvre of merging to a different road or lane becomes a conflict situation, even more so when it takes place in high-density traffic, where the driver must recalculate and adapt their position and speed in a constant way, taking into account not only the features of the road but also considering the position and speed of the surrounding vehicles.

A detailed study of ocular behaviour in several subjects in driving situations has been conducted with the aim of proposing an interface for a driver assistance system, thus facilitating merging into traffic using intervehicular communications. The driver's ocular responses to stimuli have been recorded and subsequently analysed through an eye-tracking system to design an adaptable interface to assist them with the merge.

The final purpose of the research is to analyse human factor variables that would correlate to driver workload.

##### 4.1. Study Approach

The test group comprised 8 subjects, of similar age and driving experience, extracted from the previous sample. Subjects drove on routes located in the south-eastern district of Madrid, where they performed merging on both sides of the vehicle. Data on 10–20 merging manoeuvres were collected on both sides of the road for each driver in the tests.

For the tests carried out, a vehicle was instrumented in order to capture the data from the subjects. Participants used their own cars for the tests performed, to make driving manoeuvres more natural. The on-board computer used for logging data in these tests was a i7-6700U with 8 GB RAM and using GNU/Linux.

The subjects were fitted with a Tobii Pro Glasses 2 (Figure 6), a state-of-the-art eye tracking device consisting of a pair of glasses and a software controller installed in a CPU unit. It is designed to capture natural viewing behaviour in any real-world environment while ensuring outstanding eye tracking robustness and accuracy. The glasses were provided with 2 IR sensors and 2 cameras in each eye, which enabled gaze and pupil data to be obtained; in addition, a camera placed in the centre and facing to the front recorded a video of the scene. The gaze sampling frequency is between 50 and 100 Hz and it has automatic parallax compensation. The system involves 4 eye cameras, a gyroscope and an accelerometer. The main advantages are the fact that the system is unrestrained and unobtrusive, it is capable of robust tracking with a wide cross section of the population and it is easy to use without needing extensive training.



Figure 6. Eye tracking software Tobii Pro Glasses 2.

The dark pupil detection method is used to detect the pupil, extracting the diameter from the shaded area. At the beginning of the recording, a prior calibration of the subject’s gaze is needed through a target provided by the software manufacturers.

A ROS-based package connected to the Tobii glasses is running during the whole experiment, gathering, transforming and publishing data. A logging process then stores it for further analysis. The data stored along with their description and units are summarized in Table 3.

**Table 3.** Captured variables with the Eye Tracking glasses.

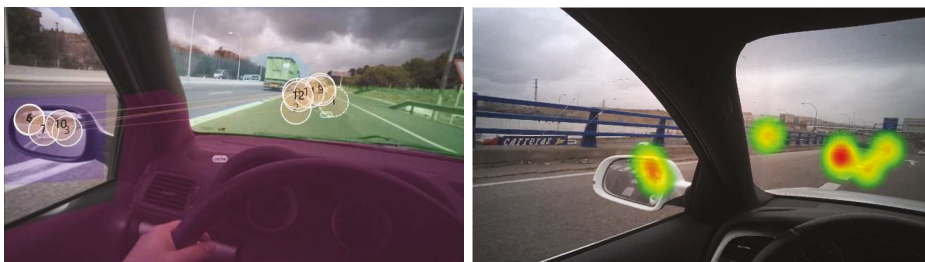
Variables	Description	Units
Gaze direction	Gazing direction for both eyes.	millimetre
Gaze position	Gaze position within the boundaries of the recording frame.	Up-left [0, 0] to bottom-right [1, 1]
Gaze position 3D	Where the pupil is located in 3D.	millimetre
Pupil diameter (Left and right)	Two variables with the diameter of each left and right pupil.	millimetre
Pupil centre (Left and right)	Two variables with the centre position for each left and right pupil.	millimetre
Gyroscope	The angular speed on each of the three axes.	millimetre
Video	The video recorded.	fps
Sampling rate		50 Hz

The glasses are connected to the embedded computer via an Ethernet cable. The transmission protocol is the one provided by Tobii [68], which works over UDP.

The image is segmented in Areas of interest (AOI) to analyse three variables: the total number of fixings per area, the duration of the first fixation and the area where it was made and the total duration of the fixation in the different areas. The areas are centre lane, right lane, left lane, right rear-view mirror, left rear-view mirror and centre mirror.

#### 4.2. Results

Pupil diameter and fixation duration were the variables chosen to study the mental load in merging situations, defined as merging time from the first second when the driver fixes his gaze on the rear-view mirror until he has fully entered the main road. Figure 7 shows an example of the evolution of eye fixation during a merging manoeuvre and a heat map of the gaze direction. The baseline of the subject is extracted, taking into account a resting situation where the activity of the eye does not perceive any external stimulus.



**Figure 7.** Eye fixations in the areas of interest. **Left figure:** Fixations numbered in order. **Right figure:** Heat map.



The following variables are statistically compared between areas of interest in relation to the baseline during normal driving conditions: (1) total duration of fixations; (2) total number of fixations; (3) duration of the first fixation. Table 4 shows the results obtained.

**Table 4.** Gaze direction comparison during merging manoeuvre with baseline.

Area of Interest	Are Differences Significant?		
	Total Duration	Fixations Number	Duration of First Fixation
Lane	Yes (baseline higher)	No	Yes (baseline lower)
Left lane	No	No	Yes (baseline lower)
Left rear screen	Yes (baseline lower)	Yes (baseline lower)	Yes (baseline lower)
Vehicle	No	No	No

In general, the value of the total duration and the number of fixations in the lane is greater in the base rate, since the driver focuses his attention on the lane and does not have to change attention focus in order to change. On the other hand, the value in the total duration and the number of fixations in the left mirror is greater during the merging manoeuvre because the driver has to pay attention to it in order to carry it out successfully.

Similarly, during the merging manoeuvre, the total number of fixations and the total duration between different areas of interest are calculated. Table 5 shows the results. The only significant coincidence between areas of interest is that quantitatively the driver looks more (both in length and number) in the left mirror than at the car when merging, because it is necessary to attend to the information provided by the mirror to perform the manoeuvre successfully.

**Table 5.** Gaze direction comparison between areas of interest during merging manoeuvre.

Area of interest	Are Differences Significant?	
	Total duration	Fixations number
Lane- Left lane	Yes (Lane > Left lane)	No
Lane- Left rear screen	No	No
Lane-Vehicle	Yes (Lane > Vehicle)	Yes (Lane > Vehicle)
Left lane- Left rear screen	No	Yes (Left lane < Left rear screen)
Left lane-Vehicle	No	Yes (Left lane > Vehicle)
Left rear screen-Vehicle	Yes (Left rear screen > Vehicle)	Yes (Left rear screen > Vehicle)

In addition, each eye pupil diameter is presented in Table 6.

**Table 6.** Pupil average diameters in merge situations and baseline.

Drivers	Average Pupil Diameter (mm)			
	Merging Scenario		Baseline Scenario	
	Left Eye	Right Eye	Left Eye	Right Eye
A	2.7116	2.7000	2.6261	2.5709
B	1.8747	1.8254	1.8866	1.8759
C	2.4157	2.3367	2.3192	2.2646
D	2.4898	2.5596	2.4603	2.4523
E	3.4948	3.4692	2.4089	2.4337
F	2.4801	2.4552	2.2878	2.2364
G	2.0994	2.0931	1.9586	1.9615
H	1.8746	1.8499	1.8318	1.7699

Drivers’ pupil diameters showed sensitivity in merging situations, the value increasing in relation to baseline. Drivers had to examine a changing situation where a decision had to be made quickly.

The results also indicate that the cognitive load in this situation rises due to the risk of the manoeuvre and the large volume of information, as is shown in other studies [69].

A Wilcoxon signed-rank test to paired-samples data was carried out to verify that statistically there are significant differences between both pupil dilations in both eyes. This test has been considered taking into account that the small size of the sample does not make it possible to determine whether the sample comes from a population with normal distribution. This test could be considered the equivalent of the t-test for paired-samples but operates with ranges instead of means.

A significant difference is found for the dilation of the right pupil between the baseline situation and merging situation, according to the variables obtained from the Wilcoxon test  $Z$  and  $p$ , which gave values of  $Z = -2.38$  and  $p = 0.016$  respectively. These significant differences are also found for the dilation of the left pupil ( $Z = -2.1$ ,  $p = 0.036$ ). These results show that there are significant differences between the two conditions, in the sense that the pupil is more dilated during merging than during baseline conditions.

#### 4.3. Discussion

Pupil diameter is considered sensitive to highway merging situations. It has been observed that pupils experience more dilation in a merging environment than in a resting situation. The diameter is inferior than for the baseline in both eyes, with this being related to driver cognitive load.

Regarding fixation duration, although in a merging situation it would seem that brain speed would be reflected in a decrease in reaction times, this condition is not fulfilled. Looking in the mirrors during the merging manoeuvre, the eye absorbs all the stimuli that can provide the necessary information for decision making in the shortest time possible; however, in the results obtained no evidence has been found for this hypothesis.

Finally, after analysing the heat maps of each of the merging situations of the different tests, a remarkably hot zone in both the left and right sides of the rear-view mirror was observed. When the subjects performed the merging manoeuvre, a large percentage of their fixations were located on the upper-inner part of the mirror in more than half of the tests conducted, as shown in Figure 7.

## 5. Conclusions

In this paper, two major issues regarding increasing road vehicle automation levels are tackled. First of all, the use of cooperative systems to provide the vehicle with a wider information horizon so that it can take decisions in advance. The definition, implementation and deployment of a C-ITS architecture for automated and/or connected vehicles on a real road have been presented, which allows automated driving to be compatible with the generation of C-ITS messages from traffic management centres. In this way, all the components that are part of the C-ITS value chain have been implemented and have been applied to the automated and connected vehicle scope. In order to carry out the validation of this architecture, modifications have been made to the guidance system of one of INSIA's automated vehicles, providing it with a module for the interpretation of C-ITS DENM messages that will be received from an ITS G5 OBU installed in the vehicle itself. This setup was validated in a real deployment on the A6 motorway in Madrid by means of a notification of public works on the road issued from the DGT TMC and the response to this incidence of the automated vehicle was analysed. The satisfactory results of this validation have shown that it is possible to achieve a convergence between the C-ITS and connected and automated driving, extending the capabilities of automated and cooperative driving since it is able to respond to the incidents reported by the TMCs in a similar way as human drivers do, in many cases being more efficient. From the point of view of the TMCs, the result of the project is a great step forward since it allows the capacities of its C-ITS to be extended, addressing not only connected but also manually driven vehicles.

Secondly, when the vehicle is in an area in which automated driving is not allowed or the vehicle expects that a situation that it cannot manage is approaching, vehicle control should be transferred back to the driver. In this scenario, the driver should demonstrate that he is alert and in good condition

for taking back control. Although some solutions consider only a minor input (pressing a button, for example), they might be ineffective because they do not guarantee the proper attention level. For this reason, some alternatives are proposed and, finally, our research concludes that the verbal task of reading a word is better than performing an arithmetical calculation to assess driver alertness. In any event, the best solution involves both a motion action and a verbal answer to ensure the driver is in a mental activation state. It should be noted that driver samples are quite small due to practical limitations and this limits extrapolation of these numerical results. However, the statistical analysis has been designed to assess whether or not the differences are significant. The tendencies and general decisions presented in our final results could thus be accepted.

Finally, after taking back vehicle control, some complex scenarios (mainly, changing lanes or merging with another road) may need additional assistance. In fact, such situations cause greater cognitive load and drivers encounter difficulties in capturing all the relevant information, as some results show, making some assistance system advisable. These systems would facilitate decision making and manoeuvres, making manual driving safer and more comfortable. Their importance increases as this safe-critical manoeuvre must be performed just after regaining vehicle control and the driver may not be completely involved in the task (even when his attention has been supervised). As future research, the proposal and testing of an assistance system for safe merging based on V2V communications and the design of a Human Machine Interfaces (HMI) is being carried out, based on the results presented in this paper (HMI location, addition workload, etc.). Furthermore, the cooperative architecture described allows information exchange between vehicles in order to advise drivers on the best way to perform the merging manoeuvre.

**Author Contributions:** F.J. proposed the complete research, supervised tests and results. J.E.N. was responsible for the C-ITS corridor deployment and tests with connected and automated vehicles. S.S. was in charge of tests with drivers and results analysis. F.S. developed interfaces. E.P., M.J.H. and T.R. developed the tests in the driving simulator and provided statistical support for the analysis of results.

**Funding:** This research has been partially funded by the Spanish Direccion General de Trafico—DGT (SICOTRAM project SPIP2017-02324), the Spanish Ministerio de Economia y Competitividad (CAV project-TRA2016-78886-C3-3-R) and the Connected Europe Facility (CEF) Program under grant agreement no. 2015-EU-TM-0243-S (AUTOCITS project).

**Acknowledgments:** We cordially thank all our colleagues from the AUTOCITS project for their collaboration in the C-ITS corridor deployment.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Hobbs, F.D. *Traffic Planning and Engineering*; Pergamon Press: Oxford, UK, 1989.
2. Stern, R.; Cui, S.; Delle Monache, M.L.; Bhadani, R.; Bunting, M.; Churchill, M.; Hamilton, N.; Haulcy, R.; Pohlmann, H.; Wu, F.; et al. Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments. *Transp. Res. Part C* **2018**, *7*, 42–57. [[CrossRef](#)]
3. Aeberhard, M.; Rauch, S.; Bahram, M.; Tanzmeister, G.; Thomas, J.; Pilat, Y.; Homm, F.; Huber, W.; Kaempchen, N. Experience, results and lessons learned from automated driving on Germany's highways. *IEEE Intell. Transp. Syst. Mag.* **2015**, *7*, 42–57. [[CrossRef](#)]
4. Mersky, A.C.; Samaras, C. Fuel economy testing of autonomous vehicles. *Transp. Res. Part C* **2016**, *65*, 31–48. [[CrossRef](#)]
5. Hofflinger, B.; Conte, G.; Esteve, D.; Weisglas, P. Integrated Electronics for Automotive Applications in the EUREKA Program PROMETHEUS. In Proceedings of the Sixteenth European Solid-State Circuits Conference. ESSCIRC'90, Grenoble, France, 19–21 September 1990; pp. 13–17.
6. Naranjo, J.E.; Bouraoui, L.; Garcia, R.; Parent, M.; Sotelo, M.A. Interoperable Control Architecture for Cybercars and Dual-Mode Cars. *IEEE Trans. Intell. Transp. Syst.* **2009**, *10*, 146–154. [[CrossRef](#)]
7. Thrun, S. Stanley: The Robot that Won the DARPA Grand Challenge. *J. Field Robot.* **2006**, *23*, 661–692. [[CrossRef](#)]

8. SAE Automotive. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*; Report J3016\_201609; SAE Automotive: Warrendale, PA, USA, 2016.
9. Harding, J.; Powell, G.; Yoon, R.; Fikentscher, J.; Doyle, C.; Sade, D.; Lukuc, M.; Simons, J.; Wang, J. *Vehicle-to-Vehicle Communications: Readiness of V2V Technology for Application*; Tech. Rep. DOT HS 812 014; National Highway Traffic Safety Administration: Washington, DC, USA, 2014.
10. California DMV. *Autonomous Vehicle Disengagement Reports*; California DMV: Sacramento, CA, USA, 2016.
11. Montemerlo, M.S.; Murveit, H.J.; Urmson, C.P.; Dolgov, D.A.; Nemeč, P. Determining When to Drive Autonomously. U.S. Patent 8718861, 6 May 2014.
12. Urmson, C.P.; Dolgov, D.A.; Chatham, A.H.; Nemeč, P. System and Method of Providing Recommendations to Users of Vehicles. U.S. Patent 9658620, 23 May 2017.
13. Krasniqi, X.; Hajrizi, E. Use of IoT Technology to Drive the Automotive Industry from Connected to Full Autonomous Vehicles. *IFAC-PapersOnLine* **2016**, *49*, 269–274. [[CrossRef](#)]
14. Vanholme, B.; Gruyer, D.; Lusetti, B.; Glaser, S.; Mammar, S. Highly automated driving on highways based on legal safety. *IEEE Trans. Intell. Transp. Syst.* **2013**, *14*, 333–347. [[CrossRef](#)]
15. Castiñeira, R.; Gil, M.; Naranjo, J.E.; Jimenez, F.; Premebida, C.; Serra, P.; Vadejo, A.; Nashashibi, F.; Abualhoul, M.Y.; Asvadi, A. AUTOCITS—Regulation study for interoperability in the adoption of autonomous driving in European urban nodes. In Proceedings of the European Transportation Research Arena 2018, Vienna, Austria, 16–18 April 2018.
16. Nikolaou, S.; Gragapoulos, I. IoT—Based interaction of automated vehicles with Vulnerable Road Users in controlled environments. In Proceedings of the 8th International Congress on Transportation Research—ICTR 2017, Thessaloniki, Greece, 27–29 September 2017.
17. Sakr, A.H.; Bansal, G.; Vladimerou, V.; Kusano, K.; Johnson, M. V2V and onboard sensor fusion for road geometry estimation. In Proceedings of the 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), Yokohama, Japan, 16–19 October 2017; pp. 1–8.
18. Rios-Torres, J.; Malikopoulos, A.A. A Survey on the Coordination of Connected and Automated Vehicles at Intersections and Merging at Highway On-Ramps. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 1066–1077. [[CrossRef](#)]
19. C-ITS Platform. *C-ITS Platform Report Phase I*; European Commission: Brussels, Belgium, January 2016.
20. C-ITS Platform. *C-ITS Platform Report Phase II*; European Commission: Brussels, Belgium, September 2017.
21. Barrachina, J.; Sanguesa, J.A.; Fogue, M.; Garrido, P.; Martinez, F.J.; Cano, J.-C.; Calafate, C.T.; Manzoni, P. V2X-d: A Vehicular Density Estimation System That Combines V2V and V2I Communications. In Proceedings of the 2013 IFIP Wireless Days, Valencia, Spain, 13–15 November 2013.
22. Ilgin Guler, S.; Menendez, M.; Meier, L. Using connected vehicle technology to improve the efficiency of intersections. *Transp. Res. Part C* **2014**, *46*, 121–131. [[CrossRef](#)]
23. Dang, R.; Ding, J.; Su, B.; Yao, Q.; Tian, Y.; Li, K. A lane change warning system based on v2v communication. In Proceedings of the 17th International IEEE Conference Intelligent Transportation Systems (ITSC), Qingdao, China, 8–11 October 2014; pp. 1923–1928.
24. Dey, K.C.; Rayamajhi, A.; Chowdhury, M.; Bhavsar, P.; Martin, J. Vehicle-to-vehicle (v2v) and vehicle-to-infrastructure (v2i) communication in a heterogeneous wireless network performance evaluation. *Transp. Res. Part C* **2016**, *68*, 168–184. [[CrossRef](#)]
25. Kaviani, S.; O'Brien, M.; Van Brummelen, J.; Michelson, D.; Najjaran, H. INS/GPS localization for reliable cooperative driving. In Proceedings of the 2016 IEEE Canadian Conference Electrical and Computer Engineering (CCECE), Vancouver, Canada, 15–18 May 2016.
26. Jin, I.; Ge, J.I.; Avedisov, S.S.; He, C.R.; Qin, W.B.; Sadeghpour, M.; Orosz, G. Experimental validation of connected automated vehicle design among human-driven vehicles. *Transp. Res. Part C* **2018**, *91*, 335–352.
27. Xu, B.; Li, S.E.; Bian, Y.; Li, S.; Ban, X.J.; Wang, J.; Li, K. Distributed conflict-free cooperation for multiple connected vehicles at unsignalized intersections. *Transp. Res. Part C* **2018**, *93*, 322–334. [[CrossRef](#)]
28. Rahman, M.S.; Abdel-Aty, M. Longitudinal safety evaluation of connected vehicles' platooning on Expressways. *Accid. Anal. Prev.* **2018**, *117*, 381–391. [[CrossRef](#)] [[PubMed](#)]
29. van Nunen, E.; Koch, R.; Elshof, L.; Krosse, B. Sensor safety for the European truck platooning challenge. In Proceedings of the 23rd World Congress on Intelligent Transport Systems, Melbourne, Australia, 10–14 October 2016.

30. O'Brien, M.; Kaviani, S.; Van Brummelen, J.; Michelson, D.; Najjaran, H. Localization Estimation Filtering Techniques for Reliable Cooperative Driving. In Proceedings of the 2016 Canadian Society Mechanical Engineering International Congress, Kelowna, Canada, 2–5 June 2016.
31. Varaiya, P.; Shladover, S.E. Sketch of an IVHS systems architecture. In Proceedings of the Vehicle Navigation and Information Systems Conference, Dearborn, MI, USA, 20–23 October 1991; pp. 909–922.
32. Talebpour, A.; Mahmassani, H.S. Influence of connected and autonomous vehicles on traffic flow stability and throughput. *Transp. Res. Part C* **2016**, *71*, 143–163. [[CrossRef](#)]
33. Lioris, J.; Pedarsani, R.; Tascikaraoglu, F.Y.; Varaiya, P. Platoons of connected vehicles can double throughput in urban Roads. *Transp. Res. Part C* **2017**, *77*, 292–305. [[CrossRef](#)]
34. Talavera, E.; Diaz, A.; Jiménez, F.; Naranjo, J.E. Impact on Congestion and Fuel Consumption of a Cooperative Adaptive Cruise Control System with Lane-Level Position Estimation. *Energies* **2018**, *11*, 194. [[CrossRef](#)]
35. Kyriakidis, M.; de Winter, J.C.; Stanton, N.; Bellet, T.; van Arem, B.; Brookhuis, K.; Martens, M.H.; Bengler, K.; Andersson, J.; Merat, N.; et al. A human factors perspective on automated driving. *Theor. Issues Ergon. Sci.* **2017**, 1–27. [[CrossRef](#)]
36. Takeda, Y.; Sato, T.; Kimura, K.; Komine, H.; Akamatsu, M.; Sato, J. Electrophysiological evaluation of attention in drivers and passengers: Toward an understanding of drivers' attentional state in autonomous vehicles. *Transp. Res. Part F* **2016**, *42*, 140–150. [[CrossRef](#)]
37. Arakawa, T.; Oi, K. Verification of autonomous vehicle over-reliance. In Proceedings of the Measuring Behavior 2016, Dublin, Ireland, 25–27 May 2016; pp. 177–182.
38. Arakawa, T. Trial verification of human reliance on autonomous vehicles from the viewpoint of human factors. *Int. J. Innov. Comput. Inf. Control* **2018**, *14*, 491–501.
39. Arakawa, T.; Hibi, R.; Fujishiro, T. Psychophysical assessment of a driver's mental state in autonomous vehicles. *Transp. Res. Part A* **2018**, in press. [[CrossRef](#)]
40. McCall, R.; McGee, F.; Mirnig, A.; Meschtscherjakov, A.; Louveton, N.; Engel, T.; Tscheligi, M. A taxonomy of autonomous vehicle handover situations. *Transp. Res. Part A* **2018**, in press. [[CrossRef](#)]
41. Politis, I.; Langdon, P.; Bradley, M.; Skrypchuk, L.; Mouzakitis, A.; Clarkson, P.J. Designing autonomy in cars: A survey and two focus groups on driving habits of an inclusive user group, and group attitudes towards autonomous cars. In Proceedings of the International Conference on Applied Human Factors and Ergonomics, Los Angeles, CA, USA, 17–21 July 2017.
42. Wintersberger, P.; Green, P.; Riener, A. Am I driving or are you or are we both? A taxonomy for handover and handback in automated driving. In Proceedings of the 9th International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design, Manchester Village Vermont, VT, USA, 26–29 June 2017.
43. Nilsson, J.; Falcone, P.; Vinter, J. Safe transitions from automated to manual driving using driver controllability estimation. *IEEE Trans. Intell. Transp. Syst.* **2014**, *16*, 1806–1816. [[CrossRef](#)]
44. Recarte, M.A.; Nunes, L.M. Mental workload while driving: Effects on visual search, discrimination, and decision making. *J. Exp. Psychol. Appl.* **2003**, *9*, 119–137. [[CrossRef](#)] [[PubMed](#)]
45. Recarte, M.A.; Nunes, L.M. Effects of verbal and spatial-imagery tasks on eye fixations while driving. *J. Exp. Psychol. Appl.* **2000**, *6*, 31. [[CrossRef](#)] [[PubMed](#)]
46. Schleicher, R.; Galley, N.; Briest, S.; Galley, L.A. Blinks and saccades as indicators of fatigue in sleepiness warnings: Looking tired? *Ergonomics* **2008**, *51*, 982–1010. [[CrossRef](#)] [[PubMed](#)]
47. Dinges, D. *PERCLOS: A Valid Psychophysiology Measure of Alertness as Assessed by Psychomotor Vigilance*; Technical Report; Federal Highway Administration: Washington, DC, USA, 1998.
48. The Royal Society for the Prevention of Accidents. *Road Accidents: A Literature Review and Position Paper*; The Royal Society for the Prevention of Accidents: Birmingham, UK, February 2001.
49. Daza, I.G.; Hernandez, N.; Bergasa, L.M.; Parra, I.; Yebes, J.J.; Gavilan, M.; Quintero, R.; Llorca, D.F.; Sotelo, M.A. Drowsiness monitoring based on driver and driving data fusion. In Proceedings of the 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC), Washington, DC, USA, 5–7 October 2011; pp. 1199–1204.
50. Flores, M.J.; Armingol, J.M.; de la Escalera, A.A. Driver drowsiness warning system using visual information for both diurnal and nocturnal illumination conditions. *EURASIP J. Adv. Signal Process.* **2010**, *2010*, 438205. [[CrossRef](#)]

51. Oron-Gilad, T.; Ronen, A.; Shinar, D. Alertness maintaining tasks (AMTs) while driving. *Accid. Anal. Prev.* **2008**, *40*, 851–860. [[CrossRef](#)] [[PubMed](#)]
52. Papadelis, C.; Chen, Z.; Kourtidou-Papadeli, C.; Bamidis, P.; Chouvarda, L.; Bekiaris, E.; Maglaveras, N. Monitoring sleepiness with onboard electrophysiological recordings for preventing sleep-deprived traffic accidents. *Clin. Neurophysiol.* **2007**, *118*, 1906–1922. [[CrossRef](#)] [[PubMed](#)]
53. Wilson, G.F.; O'Donnell, R.D. Measurement of operator workload with the neuropsychological workload test battery. *Adv. Psychol.* **1988**, *52*, 63–100.
54. Wickens, C.D. Multiple resources and performance prediction. *Theor. Issues Ergon. Sci.* **2002**, *3*, 159–177. [[CrossRef](#)]
55. Bailey, B.P.; Iqbal, S.T. Understanding changes in mental workload during execution of goal-directed tasks and its application for interruption management. *ACM Trans. Comput.-Hum. Interact.* **2008**, *14*, 1–28. [[CrossRef](#)]
56. Reimer, B.; Mehler, B.; Coughlin, J.F.; Godfrey, K.M.; Tan, C. An on-road assessment of the impact of cognitive workload on physiological arousal in young adult drivers. In Proceedings of the 1st International Conference on Automotive User Interfaces and Interactive Vehicular Applications, Essen, Germany, 21–22 September 2009; pp. 115–118.
57. Wickens, C.D.; Hollands, J. *Engineering Psychology and Human Performance*, 3rd ed.; Prentice Hall: Upper Saddle River, NJ, USA, 2000.
58. Ahlstrom, U.; Friedman-Berg, F. Using eye movement activity as a correlate of cognitive workload. *Int. J. Ind. Ergon.* **2006**, *36*, 623–636. [[CrossRef](#)]
59. Flemisch, F.; Heesen, M.; Hesse, T.; Kelsch, J.; Schieben, A.; Beller, J. Towards a dynamic balance between humans and automation: Authority, ability, responsibility and control in shared and cooperative control situations. *Cogn. Technol. Work* **2012**, *14*, 3–18. [[CrossRef](#)]
60. Debernard, S.; Chauvin, C.; Pokam, R.; Langlois, S. Designing Human-Machine Interface for Autonomous Vehicles. *IFAC-PapersOnLine* **2016**, *49*, 609–614. [[CrossRef](#)]
61. Rosario, H.D.; Solaz, J.S.; Rodriguez, N.; Bergasa, L.M. Controlled inducement and measurement of drowsiness in a driving simulator. *IET Intell. Transp. Syst.* **2010**, *4*, 280–288. [[CrossRef](#)]
62. Beruscha, F.; Wang, L.; Augsburg, K.; Wandke, H.; Bosch, R. Do drivers steer toward or away from lateral directional vibrations at the steering wheel? In Proceedings of the 2nd European Conference on Human Centred Design for Intelligent Transport Systems, Berlin, Germany, 29–30 April 2010; pp. 227–236.
63. Morrell, J.; Wasilewski, K. Design and evaluation of a vibrotactile seat to improve spatial awareness while driving. In Proceedings of the 2010 IEEE Haptics Symposium, Waltham, MA, USA, 25–26 March 2010; pp. 281–288.
64. Anaya, J.J.; Talavera, E.; Jiménez, F.; Gómez, N.; Naranjo, J.E. A novel Geobroadcast algorithm for V2V Communications over WSN. *Electronics* **2014**, *3*, 521–537. [[CrossRef](#)]
65. Talavera, E.; Anaya, J.J.; Gómez, O.; Jiménez, F.; Naranjo, J.E. Performance comparison of Geobroadcast strategies for winding roads. *Electronics* **2018**, *7*, 32. [[CrossRef](#)]
66. Naranjo, J.E.; Jiménez, F.; Gómez, O.; Zato, J.G. Low level control layer definition for autonomous vehicles based on fuzzy logic. *Intell. Autom. Soft Comput.* **2012**, *18*, 333–348. [[CrossRef](#)]
67. Jiménez, F.; Naranjo, J.E.; Serradilla, F.; Pérez, E.; Hernández, M.J.; Ruiz, T.; Anaya, J.J.; Díaz, A. Intravehicular, Short- and Long-Range Communication Information Fusion for Providing Safe Speed Warnings. *Sensors* **2016**, *16*, 131. [[CrossRef](#)] [[PubMed](#)]
68. TobiiAB. *Tobii Pro Glasses 2 API Developer's Guide v.1.26.0*; TobiiAB: Stockholm, Sweden, 2016; p. 40.
69. Poock, G.K. Information processing vs pupil diameter. *Percept. Mot. Ski.* **1973**, *37*, 1000–1002. [[CrossRef](#)]



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Article

# Performance Comparison of Geobroadcast Strategies for Winding Roads

Edgar Talavera <sup>1</sup> José J. Anaya <sup>2</sup> Oscar Gómez <sup>2</sup>, Felipe Jiménez <sup>2,\*</sup> and José E. Naranjo <sup>1,2</sup>

<sup>1</sup> Artificial Intelligence Department, Escuela Técnica Superior de Ingeniería de Sistemas Informáticos (ETSISI), Universidad Politécnica de Madrid, 28031 Madrid, Spain; edgartamu@hotmail.com (E.T.); joseeugenio.naranjo@upm.es (J.E.N.)

<sup>2</sup> University Institute for Automobile Research (INSIA), Universidad Politécnica de Madrid, Campus Sur UPM, Carretera de Valencia km.7, 28031 Madrid, Spain; jjanaya.c@gmail.com (J.J.A.); labie.insia@upm.es (O.G.)

\* Correspondence: felipe.jimenez@upm.es; Tel.: +34-913-365-300; Fax: +34-913-365-302

Received: 16 December 2017; Accepted: 1 March 2018; Published: 3 March 2018

**Abstract:** Vehicle-to-X (V2X) communications allow real-time information sharing between vehicles and Roadside Units (RSUs). These kinds of technologies allow for the improvement of road safety and can be used in combination with other systems. Advanced Driver Assistance Systems (ADAS) are an example and can be used along with V2X communications to improve performance and enable Cooperative Systems. A key element of vehicular communications is that the information transmitted through the network is always linked to a GPS position related to origin and destination (GeoNetworking protocol) in order to adjust the data broadcast to the dynamic road environment needs. In this paper, we present the implementation and development of Institute for Automobile Research (INSIA) V2X communication modules that follow the European vehicular networking standards in a close curve in a winding road where poor visibility causes a risk to the safety of road users. The technology chosen to support these communications is ETSI ITS-G5, which has the capability to enable specific services that support GeoNetworking protocols, specifically the Geobroadcast (GBC) algorithm. These functionalities have been implemented and validated in a real environment in order to demonstrate the performance of the communication devices in real V2V (Vehicle-to-Vehicle) and V2I (Vehicle-to-Infrastructure) situations. GBC messages are also compared with two different configurations of emission area. A comparison with/without RSU modules in critical areas of the road with previous knowledge of the road cartography has also been made.

**Keywords:** Vehicle-to-X communications; Intelligent Transport Systems; VANET; DSRC; Geobroadcast

## 1. Introduction

Vehicular ad hoc networks (VANETs) allow communication between vehicles, which are considered as nodes in the system. Due to the limited range of the communication channel (1 km [1]) they are considered short-range networks and their physical communications protocol is defined in the standard ETSI-G5 [2], which is the European standard equivalent to IEEE 802.11p [3]. The standards define the operation and implementation that the communications between vehicles must follow for the transmission of packages in the 5.9 GHz band (802.11p protocol).

As these networks work in variable environments and with high mobility, they require specific routing algorithms which are defined in the standards. These algorithms can manage information routing among the network nodes (vehicles and Roadside Units -RSUs-) to guarantee data availability in real time. Also, these kinds of services can enable different cooperative systems [4,5].

Thus, the geographical positioning of each network node is essential for its configuration. Given the nature of these communication networks, the most common type of transmission used is broadcast protocols. However, one of the classical limitations of broadcast-based communications is the appearance

of situations such as network saturation or flooding—particularly dangerous in this case [6]. The routing algorithms of VANETs define geographical boundaries to limit the broadcast of messages [7] that are used to specify their relevance while these kinds of boundaries also prevent flooding effects.

Thus, three GeoNetworking protocol configurations that must be supported in vehicular operations are defined [8]:

- Geounicast: conveys a message to the vehicle that is in a particular geographic position;
- Geobroadcast: conveys a message to all vehicles that are in a particular geographic area;
- Geocast: conveys a message to the nearest vehicle that is in a particular geographic area.

These operations are supported by specific GeoRouting algorithms. To guarantee the correct performance of all these algorithms, each network node maintains a neighbor location table. This table contains a time-stamped address, position, and speed for ITS (Intelligent Transport Systems) stations (nodes) in its vicinity. GeoNetworking forwarding algorithms use this neighbor location table to make forwarding decisions [9].

Although the algorithms are defined and differentiated in the standard, there are multiple changes that have been presented to improve their efficiency. Korkmaz et al. [10] presented their own model of Geobroadcast known as Urban Multihop broadcast (UMB). Since vehicle mobility is high and vehicles leave and enter the network frequently, the topology of this network changes quickly. Therefore, the UMB protocol is designed to operate without exchanging location information among nearby nodes.

Furthermore, there are other works that consider an improved Geobroadcast algorithm, such as BROADCAST [11]. This protocol is based on Geographical Routing and improves the quality of inter-vehicular broadcast communications by keeping a low load on the network.

Another improvement for a Geobroadcast algorithm is the proposal by Yu'Chun Liu et al. [12]. They proposed a Software-Defined Network (SDN) architecture for Geobroadcast in VANETs, and showed an implementation of a system that automatically manages the geographic position of each node. The results of the implementation were shown using a simulation in OpenNet.

In order to reduce the number of accidents and improve road safety, several European projects have arisen, such as GEONET and COMPASS4D. The COMPASS4D project proposed the Energy Efficiency Intersection (EEI) which uses a two-way communication system to optimize the communication of vehicles at an intersection. Another work of COMPASS4D is the Road Hazard Warning which uses communication systems between vehicles to provide a safe warnings system to the vehicles. Likewise, GEONET uses communications between vehicles to send safety messages over IPv6 in certain areas.

Some of the most novel works done with VANETs are based on simulations, like those of Tobias Queck et al. [13] where simulations of different scenarios were presented, or S. Djahel et al. [14], who showed a simulation to reduce traffic congestion. In another case, Victor Sandonis et al. [15] presented an adaptation of Proxy Mobile IPv6 (PMIPv6) with ETSI TC ITS GeoNetworking protocols to improve the overall performance. Liu Zhenyu et al. [16] proposed a communications module OBU (On Board Unit) through 802.11p communications to implement a safety system for pedestrians and vehicles. However, a few works have more advanced systems, such as [17], which implemented communications modules with the IEEE 1609.3 standard and IEEE 802.11p.

Despite the algorithms described above, we have focused on incorporating the Geobroadcast algorithm into the communication module because it adapts better to the test conditions. These modules have been implemented using GeoNetworking algorithms and multihop behavior following European standards. For this purpose, we have demonstrated its full operation with tests in a real and complex environment, understanding as complex a close curve in a winding road where poor visibility causes a risk to the safety of road users. ETSI EN 636-4-1 [18] defines the rules that communication modules follow when receiving Geobroadcast packages. The Geobroadcast algorithm transmits its messages to a specific geographical area; this area of interest is defined in the same standard where



three possible types are listed: circular, rectangular, or ellipsoidal. The ETSI EN 302 931 [19] specifies the characteristics of the areas in greater depth as well as the equations for each one.

However, the standards are still in an interim state, without the specification of some facilities layers or the security layer. However, even today there are many manufacturers that have some V2X products available on the market. Those products are not designed under a closed specification and are subject to change. In this context, a proprietary V2X communication module has been designed at the Institute for Automobile Research (INSIA) in order to support the research and deployment of V2X technology. This unit supports the current European standards in V2X communications and has passed interoperability tests in the AUTOCITS project [20] with other manufacturers (Yogoko and Cohda communication modules).

In this paper the development of the INSIA-V2X communications module is presented, including hardware and software, open to be adapted to the necessities of any cooperative systems to be developed. It includes validation testing in hazardous areas, on rural roads, specifically in complex scenarios. Thus, its operation is demonstrated in real environments like rural winding roads with adverse situations for communications and it is shown to continue operating through different configurations of emission and incorporating an RSU module in points of difficult diffusion. Two different ways of issuing Geobroadcast (GBC) messages are compared, focusing the emission area on different objectives and the same scenario with/without using a RSU module.

The following work is divided into five sections. The first section, V2X Communication System, shows the hardware and software used in each of the modules. In the section following (Implementation), the Geobroadcast algorithm used is shown in greater depth. The third section (Tests and Results) shows the results obtained during the tests performed. In the last two sections (Discussion and Conclusion), the results obtained are evaluated, comparing different configurations.

## 2. V2X Communication System

### 2.1. V2X Communication Modules

The communication modules are the INSIA-V2X (Figure 1) [21], developed internally at INSIA. Likewise, DSRC (Dedicated Short-Range Communications) INSIA-V2X modules are based on an AR9220 chipset that uses the ath9k modified driver to allow 802.11p bands. This card is configured to work in the 5855–5925 GHz bands.



**Figure 1.** DSRC INSIA-V2X (Dedicated Short-Range Communications) module.

The operating system that incorporates the modules is Debian Wheezy with kernel version 3.19.0. The kernel has been configured to allow Outside the Context of a BSS -Basic Service Set- (OCB) mode and the ath9k driver. The ath9k driver has been modified to allow the 802.11p protocol.

Each INSIA-V2X module includes the Global Navigation Satellite System NV08C-CSM chipset. It is an integrated GLONASS + GPS + GALILEO + SBAS satellite navigation receiver for use

in various applications demanding low cost (~380€ per module), low power consumption, and uncompromised performance.

2.2. Ad Hoc Network Protocol

In 1977, due to the need to create a standard capable of encompassing all computer networks, a subcommittee (SC16) was created to address Open System Interconnection (OSI) [22]. In 1997 this model was adapted to run in wireless environments, becoming the IEEE 802.11 protocol; later, in 2010, its amended version was published as the IEEE 802.11p standard to run in vehicular environments. The European mirror of this protocol is the ETSI ITS-G5.

The philosophy of vehicular communications (V2X) is that each vehicle and RSU element is considered as a network dynamic node and each node can receive and relay messages from the network. The OSI model for communications is set according to the ETSI EN 636-4-1 [18] (Figure 2).

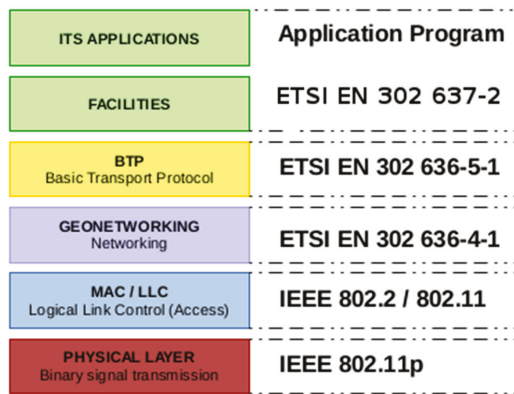


Figure 2. Open System Interconnection (OSI) model of Geonetworking protocol. MAC: Medium Access Control.

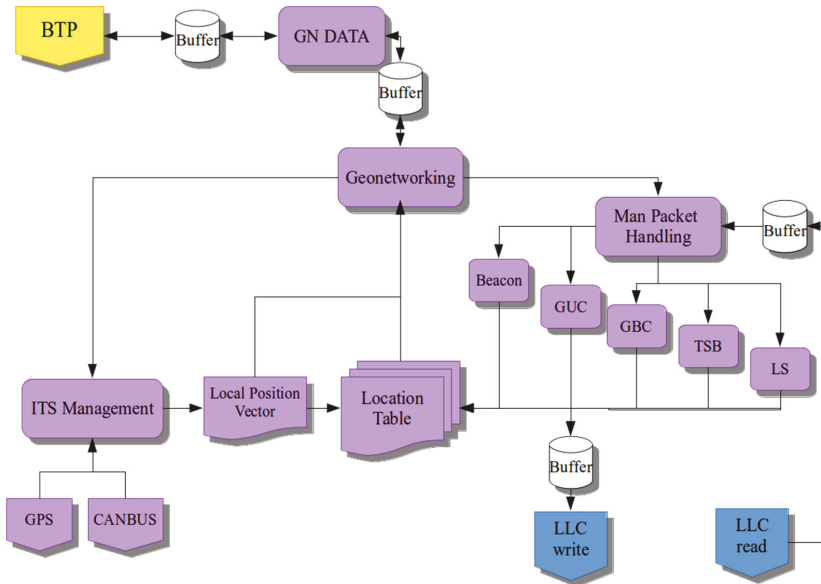
As shown in Figure 2, the Logical Link Control (LLC) corresponds to the data link layer for local area networks. The LLC layer is responsible for packing the frames with the destination MAC (Medium Access Control); upon receiving a message, it checks its MAC identifier and sends it up to the GeoNetworking layer. The network layer includes the GeoNetworking capabilities where Geobroadcast messages are processed. Inside the GeoNetworking layer, the information of the module and nearby nodes is stored in a database. Each entry in the database contains the node’s GPS position as well as the time, error, speed, etc. The protocol sends beacons from time to time to communicate its information to the rest of the system.

The Basic Transport Protocol (BTP) layer defined in ETSI [23] has also been implemented in the DRSC INSIA-V2X modules, and enables communication with external devices or applications. The link with high-level safety or efficiency applications is possible through the BTP protocol, enabling the use of this module as a V2X network access provider for vehicles and RSUs.

3. Implementation

The implementation of a DSRC INSIA-V2X module that follows the ETSI EN 636-4-1 [18] standard is presented. In the standard, different communication algorithms are defined, such as Geoanycast, Geounicast, Topology Scope Broadcast, Geobroadcast, etc. In this paper we have focused on the operation of the Geobroadcast algorithm and calculation of the defined areas for each of the possible cases.

In Figure 3, the scheme of the different logical units that operate in the presented V2X modules is shown. In the figure, the colors are associated with the layers of the OSI model in [22]. In the center of the figure is shown the Local Position Vector and Location Table responsible for storing local information and the known nodes of the module, respectively.



**Figure 3.** Software diagram. GeoUnicast (GUC), GeoBroadcast (GBC), Topology Scope Broadcast (TSB), or Location Service (LS).

The Man Packet Handling box processes the messages received by the LLC layer or the GeoNetworking box itself. In the first case, the Man Packet Handling evaluates the message, stores the information in the Location Table, and forwards it according to the protocol. In the second case, it receives the message with the data module through the GeoNetworking layer, and then packs and sends it according to the protocol (Beacon, GeoUnicast (GUC), GeoBroadcast (GBC), Topology Scope Broadcast (TSB), or Location Service (LS)).

In addition to the GeoNetworking features of the message through the GBC algorithm, the GeoNetworking layer has been designed to make a controlled and directed diffusion of the data to a limited geographical area to avoid network congestion. When treating a message, the GBC logical unit must calculate whether the node belongs to the emitting area. If the node is in the emitting area, the module manages the message and records it in the Location Table to have the last position of the node.

If the node is outside the emitting area, the next optimal jump to forward the message must be calculated. Standard ETSI EN 636-4-1 [18] defines the greedy algorithm which calculates the optimal node (closest) to send the message.

The standard defines how to analyze the area of interest and transmit the GBC messages. These formulas are applied to Cartesian Coordinates, which are not intended for the global positioning system as they begin to cause problems, especially in the rectangular and ellipsoidal area.

In the case of INSIA V2X modules, the calculation has been done using polar coordinates to simplify operations. Then, the distance between two points in geographic coordinates given the central point of the area and the position of the node is given by Equation (1).

$$D_{0N} = \text{Harversine}(P_0(\text{lat}, \text{long}), P_{\text{Node}}(\text{lat}, \text{long})) \quad (1)$$

The Harversine function [24] calculates the distance between two geographic points with the Harversine Equation (2) where the  $P_0$  and  $P_n$  parameters are the geographic positions of the focal and node points, respectively. In the case of the circle, the Harversine function is enough, but for the rectangle and the ellipse it is necessary to calculate the Northing angle to which the area is oriented.

$$\text{Harversine}_{0N} = \text{Harversine}(\text{lat}_0 - \text{lat}_N) + \cos(\text{lat}_0) \times \cos(\text{lat}_N) \times \text{Harversine}(\text{long}_0 - \text{long}_N) \quad (2)$$

The ITS management layer is responsible for constantly updating the module data obtained from can-BUS and GPS. The BTP layer is responsible for sending the information of the module to the connected applications. In this way, it communicates with the GN data program from which it requests information.

#### 4. Tests and Results

In the tests carried out, different situations were considered to test the communication modules and the GeoNetworking algorithms; for this case, only the Geobroadcast algorithm has been tested because it adapts better to the characteristics of the environment. The tests were performed in the town of Morata de Tajuña in the Madrid Region (Spain) on a road outside the urban core that includes bends which lack visibility. Figure 4 shows one of these bends at ground level for a better appreciation of the situation, including a detail of a traffic sign at the midpoint on the outside of the bend where the RSU communications module was installed. In addition, the fact that the road is very narrow should also be highlighted while the dividing line simply marks the center of the track.



Figure 4. Bend of the winding road where the Roadside Unit (RSU) was placed.

In this environment, it is possible to test the two main behaviors of the V2X modules: V2V and V2I. The former involves sending and receiving data directly from two moving vehicles in a determined geographical area to feed cooperative systems. The latter involves the use of an intermediate relay node to support multihop behavior when there is no direct link between the vehicles in the geographic destination area. In this second case, an additional RSU module was added.

The INSIA vehicles used were a Peugeot 307 and a Kymco Super Dink 125i motorcycle, each one equipped with a DRSC INSIA-V2X Communications Module. The devices used as data sources were a Motorola Nexus 6 smartphone with Android 5.1.1 and a BQ Curie 2 Quad Core tablet with Android 4.2.2. The smartphone was placed on the motorcycle due to its smaller size. Each device runs an application (app) developed to interact with the DSRC module and send its own warning messages. This app allows the sending of specific GBC messages to the onboard communications module. It also receives messages that are addressed to the module and is able to extract the information stored in the connected module.

Eight experiments were performed in total (Table 1). Those experiments are divided into two blocks. In the first block, four experiments are shown comparing two different emission centers for the area of the GBC algorithm. In the second block, the same experiments were carried out including an RSU module.

**Table 1.** Distribution experiments.

	Vehicle	GBC Characteristics	RSU
Test 1A	Stopped	Over the motorcycle	NO
Test 1B	Stopped	Over the motorcycle	YES
Test 2A	Moving	Over the motorcycle	NO
Test 2B	Moving	Over the motorcycle	YES
Test 3A	Stopped	Over the vertex of the curve	NO
Test 3B	Stopped	Over the vertex of the curve	YES
Test 4A	Moving	Over the vertex of the curve	NO
Test 4B	Moving	Over the vertex of the curve	YES

In each one of these four cases, only the motorcycle emits GBC messages. The experiments performed to compare the GBC algorithm emission area present two situations: the GBC emission area focuses on the position of the motorcycle, and the focus of the area of emission is in the center of the poor visibility bend. As a function of the cooperative system's necessities, the messages can switch from one configuration to another. Finally, to observe the operation of the algorithm in each configuration, the car may remain stationary or be in motion for each type of emission.

The results of each test are shown as graphs for a better understanding, presented in Figures 5–12. In each case, four graphs are used:

- Trajectory (top left): This graph shows the trajectory of the car (C) and the motorcycle (M) in UTM (Universal Transverse Mercator) coordinates (meters). The green line represents when both vehicles are connected (C knows M's position) and the blue one when there is no connection between them. This disconnection may be caused because the car was out of the area defined in the GBC or because this message has not been received. The black point represents the position of the RSU module.
- GBC–Vehicle distance (top right): This graph shows the distance in meters between each vehicle with respect to the center of the GBC reception area. The green line represents the distance when the motorcycle has a connection with the car. The purple line represents the range of the GBC area, set at 100 m from the center in every case. The blue line is the evolution of the distance to the center of the motorcycle during the test and the red line is the evolution of this distance of the car. Finally, the red line represents the distance from the RSU module (when available) to the center of the reception area of the GBC.
- Vehicle speed (bottom left): This graph displays the speed of each vehicle in m/s. The red line represents the speed of the motorcycle and the blue line the speed of the car. Similar to the previous graphs, the color of the motorcycle line changes to green when there is communication with the car.
- Vehicle distance (bottom right): This graph shows the distance in meters between the vehicles and the RSU. The blue line represents the distance between the car and the motorcycle. The black line

is the distance between the motorcycle and the RSU and the green line is the distance between the car and the RSU.

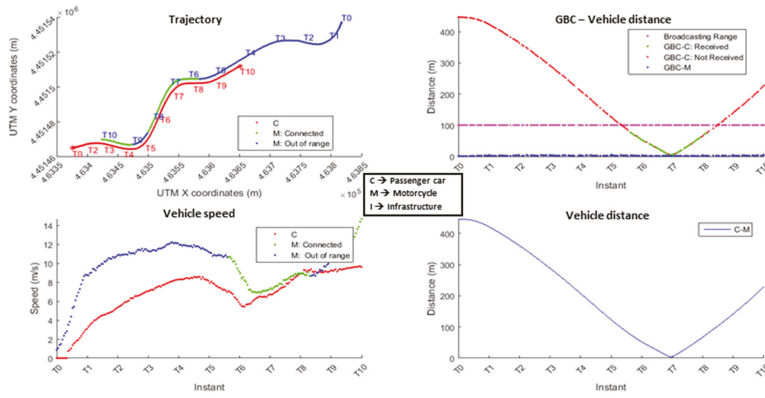


Figure 5. Results of Test 1 without RSU.

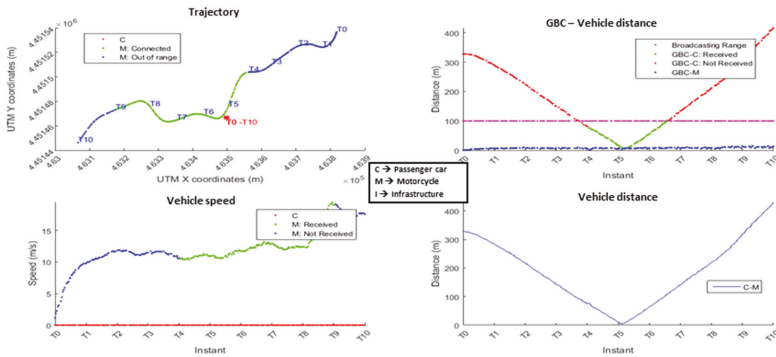


Figure 6. Results of Test 2 without RSU.

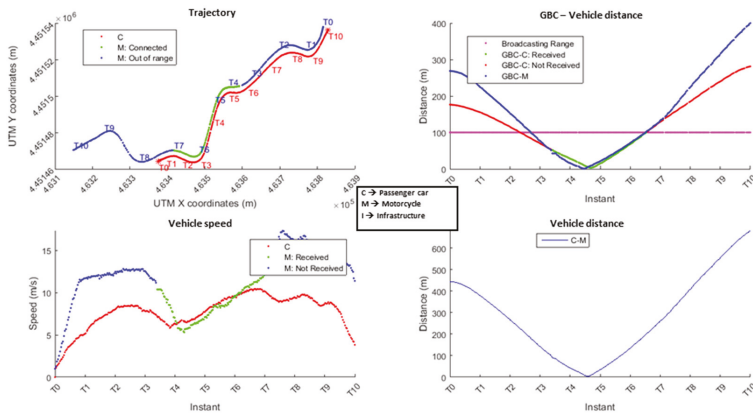


Figure 7. Results of Test 3 without RSU.

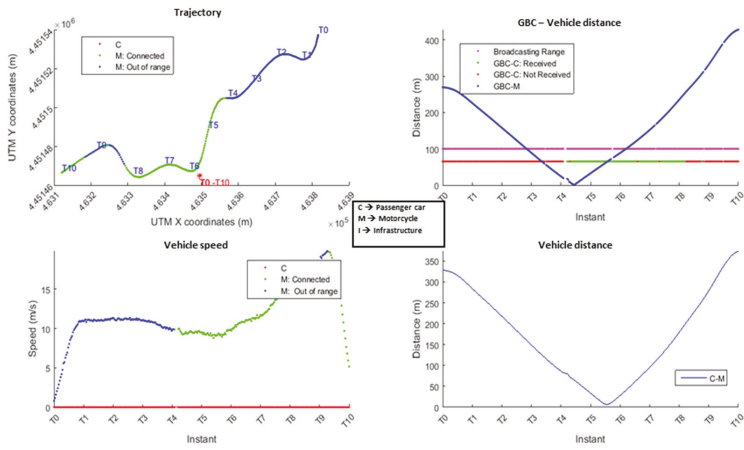


Figure 8. Results of Test 4 without RSU.

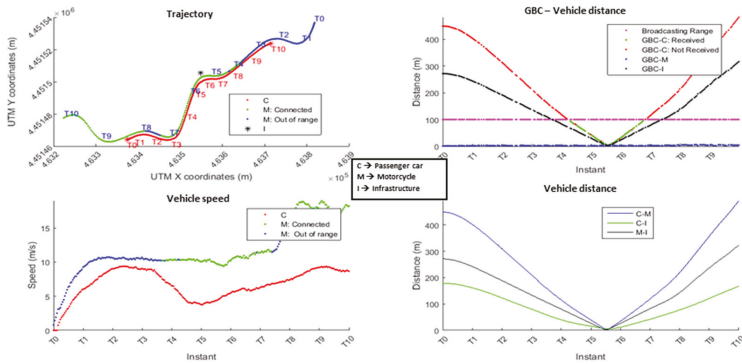


Figure 9. Results of Test 1 with RSU.

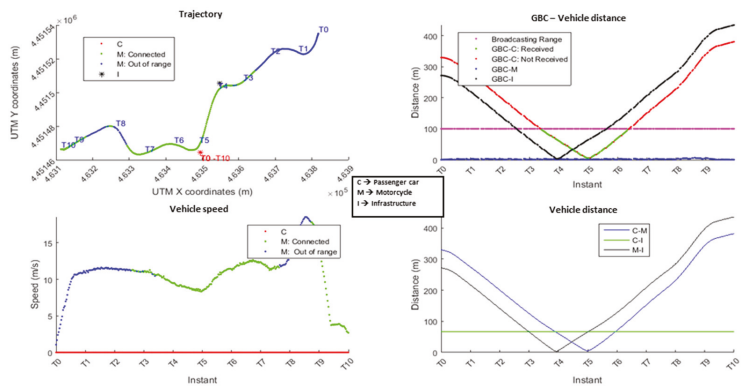


Figure 10. Results of Test 2 with RSU.

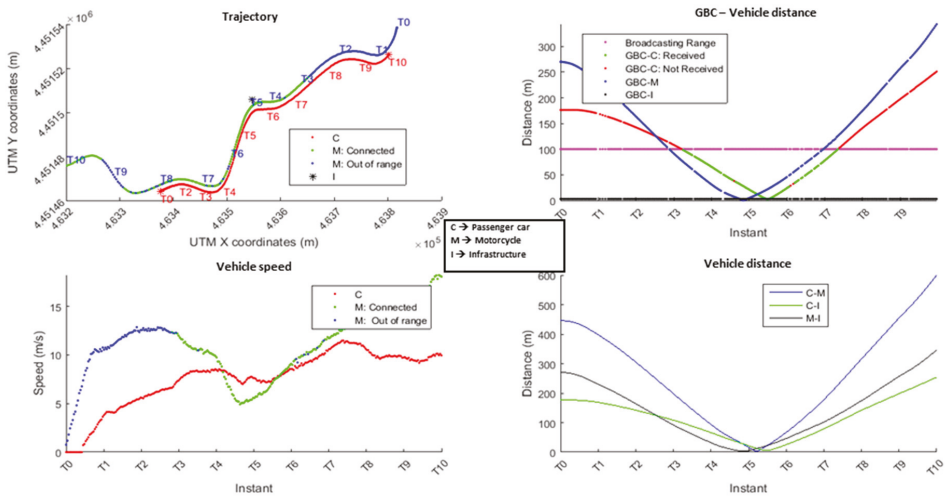


Figure 11. Results of Test 3 with RSU.

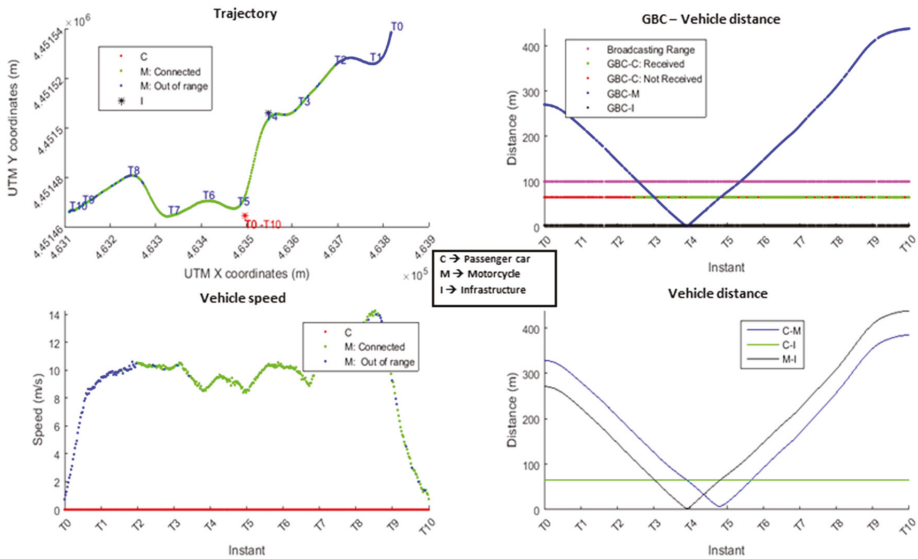


Figure 12. Results of Test 4 with RSU.

In the experiments the time has been represented as  $T_i$  in order to synchronize the trajectories of both vehicles. Time  $T_0$  is considered when the experiment begins and  $T_{10}$  when it finishes. For all tests performed, the message transmission frequency was 4 Hz (4 frames per second).

#### 4.1. Validation without RSU Module

In these tests, two different types of configuration for GBC messages were tested. Therefore, for each type of configuration a test was performed with a static car and another simulating a situation of real driving. The four tests were configured as follows:



- Test 1 (Figure 5): The circular area 100 m in radius was focused on the motorcycle and the car remained immobile throughout the test.
- Test 2 (Figure 6): The configuration of the GBC messages was the same as for Test 1 but the car was moving in the opposite direction.
- Test 3 (Figure 7): The circular area 100 m in radius was focused on the vertex of the curve and the car remained immobile throughout the test.
- Test 4 (Figure 8): The configuration of the GBC messages was the same as for Test 3 but the car was moving in the opposite direction.

With the results obtained, it can be seen that the car takes time to obtain information about the motorcycle. This is due to the fact that the terrain orography prevents communication between modules. Even with the orography, both drivers are warned about the position of the other vehicle before they have eye contact. However, this warning is not carried out enough in advance and this fact justifies placing a module as an RSU to solve this problem.

#### 4.2. Validation with RSU Module

In this section, the same cases as in the previous one are presented but including a module as an RSU. This verifies how the V2I communications affect the system and if they are able to improve the delay of the previous cases. The four tests were configured as follows:

- Test 1B (Figure 9): The circular area 100 m in radius was focused on the motorcycle and the car remained immobile throughout the test.
- Test 2B (Figure 10): The configuration of the GBC messages was the same as for Test 1 but the car was moving in the opposite direction.
- Test 3B (Figure 11): The circular area 100 m in radius was focused on the vertex of the curve and the car remained immobile throughout the test.
- Test 4B (Figure 12): The configuration of the GBC messages was the same as for Test 3 but the car was moving in the opposite direction.

With the results obtained, a significant improvement can be verified in all cases. In the GBC–Vehicle distance graph, it can be seen that there is no loss of frames. At the moment the vehicle is within the area of interest, it receives information from the motorcycle. Therefore, the RSU module allows problems in communications due to orography to be solved, acting as a relay for the GBC messages by placing its broadcast area in zones where there is no node.

In these tests performed with RSU modules, the differences between the two GBC message settings can be seen. In the configuration where we fix the center of the emission area at the vertex of the curve, information of the motorcycle is obtained in advance. This advantage allows the driver of the car to anticipate any situation with more time.

The problem associated with this type of configuration is that the situation of the road must be known in order to accurately position the broadcast areas. This inconvenience can only be solved by having the cartography.

## 5. Discussion

During the tests, we collected all the packets emitted by the motorcycle, regardless of whether they were received by the car in or out of the GBC circular reception area. The information obtained from the reception of these messages allows us to analyze the behavior of the network, maximum range, effective range, position of the vehicles, or transmission time in the different configurations. Table 2 shows the comparison of the range in the 8 tests. The distance is presented as a straight line between the vehicle and the motorcycle when they connect for the first time, whether or not they are in the GBC coverage range.

**Table 2.** Comparison among the different ranges of the system in the tests.

	Distance (Meters)	RSU	Transmission Delay
Test 1A	80 m	No	<1 ms
Test 1B	150 m	Yes	<1 ms
Test 2A	60 m	No	<1 ms
Test 2B	150 m	Yes	<1 ms
Test 3A	74 m	No	<1 ms
Test 3B	190 m	Yes	<1 ms
Test 4A	70 m	No	<1 ms
Test 4B	210 m	Yes	<1 ms

Results shown in Table 2 corroborate the conclusions described in the previous section, verifying the improved behavior of centering the GBC destination area in the middle of the poor visibility bend. This guarantees coverage of messages as soon as possible when an emitting vehicle is approaching the dangerous area. It is also important to remark that the inclusion of intermediate relay RSU modules has no influence on the transmission latency of the messages. This means that the developed INSIA-V2X communication modules do not introduce delays into the system.

Analysis of the GBC packets (sent and received) to obtain a packet loss rate was also done to validate the performance of the system. These results are shown in Table 3 as well as the time for which the vehicles were connected during the tests.

**Table 3.** Packets sent and received in the experiments.

	GBC Packets Sent	GBC Packets Sent When the Car Is Inside the Transmission Area	GBC Packets Received by the Car	GBC Packets Not Received Rate	Time Connected
Test 1A	247	87	65	25.3%	29.8 s
Test 1B	301	100	88	12%	43 s
Test 2A	179	59	43	27.1%	15.6 s
Test 2B	224	61	53	13.1%	29.4 s
Test 3A	308	308	124	59.7%	36 s
Test 3B	365	365	253	30.6%	56.2 s
Test 4A	246	127	61	51.9%	22.4 s
Test 4B	272	113	109	4.4%	34.6 s

Keep in mind that the packet delivery rate emitted by the motorcycle is 4 Hz (4 frames per second) and most of the packets not received are mainly due to the orography.

The sent packets column of Table 3 corresponds to the total GBC packets sent from when the test started until it was finished. The next column corresponds to the number of data packets sent by the motorcycle when the car was in the GBC circular reception area and the fourth column represents the GBC packets received by the car when it was within the GBC circular reception area. In addition, these packages were received and processed by the car to be within the destination area of transmission. Finally, the last column represents the time during which both vehicles were connected, interchanging messages.

The results shown in Table 3 are discussed separately in order to understand the differences between each test:

- Test 1A: The rate of non-received packets during the test is high—about 25%. This is because the car remains in the same position throughout the test and does not receive messages from the motorcycle until it is within the line of sight of the bend, although the car is in the GBC range. Figure 5 shows that these losses occur mainly at the beginning and end of the entry into the GBC range, as the two vehicles are driven on the low visibility curve.
- Test 1B: By including the RSU module in the test, the number of packets that are not received is reduced by more than half, producing only sporadic packet losses; as can be seen in Figure 6, the car messages are received when the motorcycle enters the coverage area.

- Test 2A: In this case, the rate of non-received packets is also high, at around 27%. As in Test 1A, orography plays an important role in the test and the car takes longer to receive the first data from the motorcycle (it is at the opposite side of the curve) and there is no direct line of sight with the car. This data loss occurs at the beginning and at the end of the test, as shown in Figure 7.
- Test 2B: When including the RSU module, the number of packets that are not received is reduced by more than half, with the appearance of sporadic losses as shown in Figure 8.
- Test 3A: By selecting the center of the GBC range at the apex of the curve, the results are apparently worse since the rate of non-received packets during the test is 59%. This test is the one with the highest number of packets not received by the car. This is because the configuration of the GBC message causes the emission area to focus on the curve and, therefore, the car will always be within the broadcast area of the total packets emitted from the start of the trial.
- Test 3B: An RSU module is included, solving a part of the communication problem of Test 3A. The number of packets not received by the car is considerably reduced, although it is still high because they are calculated from the beginning of the trial, since the car is always stopped within the coverage area. As shown in Figure 10, the time during which the car receives messages is higher than for any of the above tests, as the car receives data packets at the same instant that the motorcycle has the direct vision line to the RSU module.
- Test 4A: In this case, the rate of non-received messages is high—around 51%—for the same reason as in Test 3A. As in other tests, the orography prevents communication between vehicles and this causes many messages to not reach the car.
- Test 4B: Finally, the best results occur with configuration 4B, where the rate of non-received packets is 4%. This gives the best result because the RSU module allows more efficient communication. It increases the range of communications and the car is in the coverage area when the motorcycle is connected with the RSU module, receiving the data. This setting is most appropriate when deploying a cooperative system and ensures that there will be maximum connectivity at the moment when there is greatest risk of an accident.

It is an obvious conclusion that the best configuration of V2X architecture is when an RSU module is installed at the roadside. When RSUs are used, the number of received packets increases; this is due to the fact that the RSU retransmits the packets it receives and it may be the case that messages are repeated. However, the standard indicates that when a module receives a duplicate of a package it has already received, this will be wasted. There is a second factor that definitively influences the performance of this configuration: the position of the center of Geobroadcast areas. As shown in the figures, the location of this center in the critical zone of the RSU definitively improves the performance and range of the communication. Furthermore, a priori knowledge of the location of this critical area is necessary to enable this configuration and this information is outside the GBC standard, depending only on the C-ITS applications. C-ITS applications should then have digital cartography data about the critical areas where the location of the GBC centers will improve their functionality.

## 6. Conclusions

In this paper, we have shown the correct operation of DSRC modules with the Geobroadcast algorithm in a close curve in a winding road where poor visibility causes a risk to the safety of road users. Communications were made through the 802.11p protocol and tested using real instrumented vehicles. With the results obtained in the tests, the system was shown to be robust and well functioning.

The work carried out in [25] was complemented by testing the Geobroadcast algorithm and the system under a situation of scarce coverage for communications. In addition, there is a problem derived from centering the area of emission in a curve in which no node is found. Two different emitting configurations were provided for this situation by focusing the area on a vehicle or including an RSU module in the vertex of the curve.

The system was tested in a scenario where the orography plays an important role in the communications, as detailed in [26]. With the orography situation chosen, it is not possible to predict which would be the best way to center the emission area if the road conditions are not previously known. In this case, it is demonstrated that the solution is given by placing an RSU module.

As reflected in the tests performed, it is important to know the conditions of the road in order to choose the best strategy of emission and configuration of the system. The RSUs are an important part of the system, preventing blind spots in communications, but it is not the only tool, as demonstrated in this paper. Therefore, this paper has demonstrated two different ways of issuing GBC messages as well as the addition of RSU communication modules in critical risky areas.

As the tests have been oriented to a specific scenario, there are aspects of the communication modules that have not been tested. In future work, the facilities layer defined in the standard could be developed and tested as to how its operation affects the system. In addition, it should be noted that it would be subject to study as to how scalability affects the system with a different number of modules emitting at different frequencies.

**Acknowledgments:** This work has received the support of the Spanish Ministerio de Economía y Competitividad MINECO (TRA2016-78886-C3-3-R) and Madrid Region Excellence Network SEGVAUTO-TRIES (S2013/MIT-2713).

**Author Contributions:** All authors contributed to the paper. E. Talavera and J. J. Anaya developed the communications modules. O. Gómez prepared the tests. F. Jimenez and J. E. Naranjo proposed the system and the scenario and analyzed the final results.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Jakubiak, J.; Koucheryavy, Y. State of the art and research challenges for VANETS. In Proceedings of the 5th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, USA, 10–12 January 2008; p. 912.
2. Intelligent Transport Systems (ITS). *European Profile Standard for the Physical and Medium Access Control Layer of Intelligent Transport Systems Operating in the 5 GHz Frequency Band*, v1.1.0 edition; European Telecommunications Standards Institute: Sophia Antipolis Cedex, France, 2009.
3. *802.11p Local and Metropolitan Area Networks-Specific Requirements-part11: Wireless Lan Medium Access Control (mac) and Physical Layer (phy) Specifications Amendment 6: Wireless Access in Vehicular Environments*; Technical Report; IEEE: Piscataway, NJ, USA, 2010.
4. Hussain, R.; Son, J.; Eun, H.; Kim, S.; Oh, H. Traffic information system: A lightweight geocast-based piggybacking strategy for cooperative awareness in VANET. In Proceedings of the 2013 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 11–14 January 2013; pp. 614–615.
5. Atéchian, T.; Torbey, Z.; Bennani, N.; Brunie, L. CoFFee: Cooperative and inInfrastructure-Free peer-to-peer system for VANET. In Proceedings of the 2009 9th International Conference on Intelligent Transport Systems Telecommunications, Lille, France, 20–22 October 2009; pp. 510–515.
6. Prasetijo, A.B.; Alwakeel, S.S.; Altwaijry, H.A. Effects of VANET's attributes on network performance. In Proceedings of the 2014 1st International Conference on Information Technology, Computer and Electrical Engineering (ICITACEE), Semarang, Indonesia, 8–9 November 2014; pp. 303–308.
7. Meneguette, R.; Villas, L. An Autonomic Algorithm for Data Dissemination in Vehicular Ad Hoc Networks. *IEEE Lat. Am. Trans.* **2014**, *12*, 430–435. [[CrossRef](#)]
8. Festag, A. Cooperative intelligent transport systems standards in Europe. *IEEE Commun. Mag.* **2014**, *52*, 166–172. [[CrossRef](#)]
9. *Observations on GeoNetworking, Document HTG1&3-3*; Version: 2012-11-12, EU-US ITS Task Force, Standards Harmonization Working Group, Harmonization Task Groups 1 & 3; U.S. Department of Transportation: Washington, DC, USA, 2011.
10. Korkmaz, G.; Ekici, E.; Özgüner, U. Urban multi-hop broadcast protocol for inter-vehicle communication systems. In Proceedings of the 1st ACM International Workshop on Vehicular Ad Hoc Networks, VANET '04, New York, NY, USA, 1 October 2004; pp. 76–85.

11. Durresti, M.; Durresti, A.; Barolli, L. Emergency broadcast protocol for intervehicle communications. In Proceedings of the 11th International Conference on Parallel and Distributed Systems, Fukuoka, Japan, 20–22 July 2005; Volume 2, pp. 402–406.
12. Liu, Y.C.; Chen, C.; Chakraborty, S. A Software Defined Network architecture for GeoBroadcast in VANETs2. In Proceedings of the 2015 IEEE International Conference on Communications (ICC), London, UK, 8–12 June 2015; pp. 6559–6564.
13. Queck, T.; Schünemann, B.; Radusch, I.; Meinel, C. Realistic Simulation of V2X Communication Scenarios. In Proceedings of the IEEE Asia-Pacific Services Computing Conference, Yilan, Taiwan, 9–12 December 2008.
14. Djahel, S.; Jabeur, N.; Barrett, R.; Murphy, J. Toward V2I communication technology-based solution for reducing road traffic congestion in smart cities. In Proceedings of the 2015 International Symposium on Networks, Computers and Communications (ISNCC), Hammamet, Tunisia, 13–15 May 2015; pp. 1–6.
15. Sandonis, V.; Calderon, M.; Soto, I.; Bernardos, C.J. Design and performance evaluation of a PMIPv6 solution for geonetworking-based VANETs. *Ad Hoc Netw.* **2013**, *11*, 2069–2082. [[CrossRef](#)]
16. Zhenyu, L.; Lin, P.; Konglin, Z.; Lin, Z. Design and evaluation of V2X communication system for vehicle and pedestrian safety. *J. China Univ. Posts Telecommun.* **2015**, *22*, 18–26. [[CrossRef](#)]
17. Meng, Z.; Liu, Z.; Gu, X.; Pu, L.; Yang, X.; Qin, Z.; Zhu, K.; Zhang, L. Guaranteed V2V QoS services implementation and field measurements in hybrid WAVE/TE environments. In Proceedings of the TENCON 2015—2015 IEEE Region 10 Conference, Macao, China, 1–4 November 2015; pp. 1–6.
18. Intelligent Transport System (ITS). *Vehicular Communications; GeoNetworking; Part 4: Geographical Addressing and Forwarding for Point-to-Point and Point-to-Multipoint Communications; Sub-Part 1: Media-Independent Functionality*, v1.2.1 edition; European Telecommunications Standards Institute: Sophia Antipolis Cedex, France, 2014.
19. Intelligent Transport System (ITS). *Vehicular Communications; Geographical Area Definition*, v1.1.1 edition; European Telecommunications Standards Institute: Sophia Antipolis Cedex, France, 2014.
20. Jiménez, F.; Naranjo, J.E.; Castiñeira, R.; Gil, M. Cooperative systems for supporting autonomous driving in European urban nodes. In Proceedings of the 15th EAEC European Automotive Congress, Madrid, Spain, 3–5 October 2017.
21. Anaya, J.J.; Talavera, E.; Giménez, D.; Gómez, N.; Jiménez, F.; Naranjo, J.E. Vulnerable Road Users Detection using V2X Communications. In Proceedings of the 2015 IEEE 18th International Conference on Intelligent Transportation Systems, Gran Canaria, Spain, 15–18 September 2015.
22. ISO/TC97/SC16. *Reference Model of Open Systems Interconnection*; ISO: London, UK, 1979.
23. Intelligent Transport System (ITS). *Vehicular Communications; GeoNetworking; Part 5: Transport Protocols; Sub-Part 1: Basic Transport Protocol*, v1.2.1 edition; European Telecommunications Standards Institute: Sophia Antipolis CEDEX, France, 2014.
24. Robusto, C.C. The Cosine-Haversine Formula. *Am. Math. Mon.* **1957**, *64*, 38–40. [[CrossRef](#)]
25. Anaya, J.J.; Talavera, E.; Jimenez, F.; Serradilla, F.; Naranjo, J.E. Vehicle to vehicle geonetworking using wireless sensor networks. *Ad Hoc Netw.* **2015**, *27*, 133–146. [[CrossRef](#)]
26. Naranjo, J.E.; Jimenez, F.; Anaya, J.J.; Talavera, E.; Gomez, O. Application of vehicle to another entity (V2X) communications for motorcycle crash avoidance. *J. Intell. Transp. Syst.* **2016**. [[CrossRef](#)]



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

MDPI  
St. Alban-Anlage 66  
4052 Basel  
Switzerland  
Tel. +41 61 683 77 34  
Fax +41 61 302 89 18  
[www.mdpi.com](http://www.mdpi.com)

*Electronics* Editorial Office  
E-mail: [electronics@mdpi.com](mailto:electronics@mdpi.com)  
[www.mdpi.com/journal/electronics](http://www.mdpi.com/journal/electronics)





MDPI  
St. Alban-Anlage 66  
4052 Basel  
Switzerland

Tel: +41 61 683 77 34  
Fax: +41 61 302 89 18

[www.mdpi.com](http://www.mdpi.com)



ISBN 978-3-03921-376-4