electronics MDPI

Article

The Design of Preventive Automated Driving Systems Based on Convolutional Neural Network

Wooseop Lee [1], Min-Hee Kang [1], Jaein Song [2] and Keeyeon Hwang [3,*]

[1] Department of Smart City, Hongik University, Seoul 04066, Korea; wooseabi@naver.com (W.L.); speakingbee@hanmail.net (M.-H.K.)
[2] Research Institute of Science and Technology, Hongik University, Seoul 04066, Korea; wodlsthd@nate.com
[3] Department of Urban Planning, Hongik University, Seoul 04066, Korea
* Correspondence: keith@hongik.ac.kr

Abstract: As automated vehicles have been considered one of the important trends in intelligent transportation systems, various research is being conducted to enhance their safety. In particular, the importance of technologies for the design of preventive automated driving systems, such as detection of surrounding objects and estimation of distance between vehicles. Object detection is mainly performed through cameras and LiDAR, but due to the cost and limits of LiDAR's recognition distance, the need to improve Camera recognition technique, which is relatively convenient for commercialization, is increasing. This study learned convolutional neural network (CNN)-based faster regions with CNN (Faster R-CNN) and You Only Look Once (YOLO) V2 to improve the recognition techniques of vehicle-mounted monocular cameras for the design of preventive automated driving systems, recognizing surrounding vehicles in black box highway driving videos and estimating distances from surrounding vehicles through more suitable models for automated driving systems. Moreover, we learned the PASCAL visual object classes (VOC) dataset for model comparison. Faster R-CNN showed similar accuracy, with a mean average precision (mAP) of 76.4 to YOLO with a mAP of 78.6, but with a Frame Per Second (FPS) of 5, showing slower processing speed than YOLO V2 with an FPS of 40, and a Faster R-CNN, which we had difficulty detecting. As a result, YOLO V2, which shows better performance in accuracy and processing speed, was determined to be a more suitable model for automated driving systems, further progressing in estimating the distance between vehicles. For distance estimation, we conducted coordinate value conversion through camera calibration and perspective transform, set the threshold to 0.7, and performed object detection and distance estimation, showing more than 80% accuracy for near-distance vehicles. Through this study, it is believed that it will be able to help prevent accidents in automated vehicles, and it is expected that additional research will provide various accident prevention alternatives such as calculating and securing appropriate safety distances, depending on the vehicle types.

Keywords: automated driving systems; the design of preventive; CNN; vehicle detection; distance estimation

check for updates

Citation: Lee, W.; Kang, M.-H.; Song, J.; Hwang, K. The Design of Preventive Automated Driving Systems Based on Convolutional Neural Network. *Electronics* **2021**, *10*, 1737. https://doi.org/10.3390/electronics10141737

Academic Editor: Nikolay Hinov

Received: 1 June 2021
Accepted: 17 July 2021
Published: 19 July 2021

1. Introduction

Automated vehicles have been regarded as one of the most important trends in intelligent transportation systems with rapid developments recently, and are evaluated to enhance vehicular traffic, including increased highway capacity and traffic flow and fewer accidents with collision prevention systems [1,2]. Currently, automated vehicles are undergoing various research and development in GM, Waymo, Ford, etc., due to the convergence of ICT, and are focused on commercialization and product production [3]. In particular, as many patents such as collision prevention technology (Automatic Distance Control/ADC) and sensing and tracking technology (Automatic Exposure Control/AEC) have been applied, they are contributing a lot to safety and convenience, and the smart car market is expected to grow at a faster pace in the future [4]. However, for automated vehicles to be successfully introduced into the market, there are various problems such as user acceptance, and safety must be guaranteed for them to be commercialized in everyday traffic [5]. Accordingly, the importance of developing essential automated driving technology is increasing, and the paradigm is shifting to an active automated driving preventive design that can prevent accidents in advance [6]. For this preventive design, it is necessary to accurately recognize the various surrounding environments in which automated vehicles operate [7]. In particular, in environments such as highways, while driving at high speed, the severity of accidents is high compared to the number of accident occurrences, and accidents often occur due to non-compliance of safe driving responsibility and non-secure safety distance, making it very important to recognize surrounding vehicles and estimate the distance between vehicles in automated driving system [8–10].

Object detection is one of the techniques that must be performed for automated driving and traffic safety, mainly through cameras and LiDAR, where visual perception and motion prediction are carried out [11]. However, as LiDAR shows limitations of high cost and short recognition distance, much research has been conducted on improving camera recognition techniques at a relatively low cost [12]. In particular, various research incorporating Artificial Intelligence (AI) has been actively conducted in varied fields, including transportation technology, and has made many advances in video and image processing based on deep learning, a field of AI. CNN, one of the deep learning networks, is an effective model for detecting objects, extracting features, and classifying them, showing good results in image processing and classification [13]. By identifying the vehicle and its surrounding environment through CNN, it is possible to ensure that the automated vehicle can drive safely without colliding with other vehicles on the road, and present solutions to minimize unsafe behavior [14,15]. In particular, R-CNN, one of the CNN models, detected objects through Region Proposal to infer the likely location of objects and then improved performance on models such as Fast R-CNN and Faster R-CNN [16–18]. Among these R-CNN-based models, Faster R-CNN showed high performance in object detection, with higher accuracy and faster processing speed throughput compared to conventional models through end-to-end learning and region proposal network (RPN). Another CNN model, YOLO, is made slightly less accurate by eliminating Region Proposal steps, unlike the existing object detection model, but has been developed rapidly in real-time object detection with fast processing speed, and has recently been released to YOLO V4 and beyond, showing improved performance [19,20].

This study proposes the application of deep learning-based CNN to improve the recognition technique of vehicle-mounted monocular cameras for the design of preventive automated driving systems. To this end, we have introduced a variety of CNN methods and select CNN models suitable for analysis based on the performance and limitations of each model to recognize surrounding vehicles and estimate the distance from them through the suitable model for automated driving systems.

In summary, the main contributions of this work are listed below:

1. It contributes to the prevention of accidents by designing preventive automated driving systems through improving the camera's recognition technique to be suitable for commercialization in terms of cost and recognition distance compared to LiDAR.
2. It applies a better model for automated driving systems through performance comparisons of CNN methods that have recently made significant advances in object detection.
3. Because it is difficult to obtain driving data, black box videos with the most similar collection location to those of automated vehicles and relatively easy to collect data were collected and learned.
4. It can be used as basic materials in calculating the appropriate safety distance between vehicles in the future by estimating the distance according to the coordinates.

This study is conducted in the following order: In Section 2, we draw the differentiation of this study by reviewing the related studies on object detection and distance estimation of automated vehicles and CNN and, in Section 3, explain the methodology benchmarked in the study. Next, after setting up the learning environment in Section 4, we compare and analyze the learning results of each model and estimate the distance from the surrounding vehicles through a more suitable model. Finally, in Section 5, we summarize the learning results and suggest implications and future research.

2. Related Works

Object detection and distance estimation of automated vehicles have recently made significant progress with research on the method of single-use of cameras, LiDAR and Radar sensors [10,21–24], and the method of fusion of cameras and sensors [25–27], as well as AI-based machine learning and deep learning, which has also been applied [1,11,28].

Kehtarnavaz et al. [22] compared the mono-vision system and stereo-vision system, which are the camera systems used in automated vehicles. In their study, the stereo camera is a more efficient vision system based on the difference that the stereo system detects objects and estimates distance through two cameras, while mono system allows only one camera to detect and classify objects. Radar and LiDAR, other sensors used in automated vehicles, are sensors that detect and rank using radio and light, and various studies have been conducted to apply them [23]. Nabati et al. [24] proposed radar region proposal network (RRPN), a real-time region proposal algorithm based on radar sensors, to compensate for the limitations on slow processing speed of region propositional algorithms, a method that performs object detection by assuming the location of the object. The proposed method showed over 100 times faster and more accurate results than the existing selective search algorithm, and showed good performance even though it was the method of single-use for object detection. Zaarane et al. [10] proposed a method of calculating the distance between vehicles through the location of vehicles, geometric derivations, and specific angles (such as the camera view field angles), etc., using the mounted stereo camera on hosting vehicles for estimating the distance between vehicles, but indicated that the stereo camera has a difficulty measuring real three-dimensional coordinates.

While cameras are effective in detecting and classifying objects, Radar and LiDAR sensors are suitable for detecting objects and obtaining information such as range or geometric structure but have limitations in classifying objects [24]. Accordingly, various methods of fusing both sensors and cameras are being studied. Zhao et al. [26] reduced the average processing time to 66.79 ms/frame by generating fewer but more accurate object-region proposals by fusing 3D Lidar and vision camera information, and the average identification accuracies for vehicles was also showed to have excellent performance with 89.04%. Rashed et al. [27] proposed another fusion method, FuseMODNet. The proposed method is a real-time CNN architecture for moving object detection (MOD) in autonomous driving under low-light conditions by capturing motion information from cameras and LiDAR sensors, demonstrating a 4.25% improvement in automated vehicles by building a new 'Dark-KITTI' dataset in low-light environments based on the standard 'KITTI' dataset, and it was also shown that it can be applied in real-time to autonomous vehicles at 18 fps.

The data collected through cameras, Radar, and LiDAR have been combined with AI-based machine learning and deep learning to enable object detection and distance estimation with more accurate and faster processing speeds, and various analysis models are continuously being developed. Masmoudi et al. [1] described and investigated an image-based object detection model applicable to automated vehicles. In this study, they focused on machine learning-based support vector machine (SVM), deep learning-based YOLO, and the single shot multibox detector (SSD) and compared their performance through simulations. The analysis confirmed that SVM is not suitable for real-time analysis due to its slow processing speed, and YOLO is suitable for real-time processing but has lower performance than multi-scale SSD, so it is necessary to use it differently depending on the purpose of application. Stereo R-CNN [11] was proposed as a method for detecting and localizing 3D objects, showing approximately 30% outperformance in accuracy over the state-of-the-art methods, and confirming that it can be used for multi-object detection and tracking in the future as well as general object detection.

CNN-based object detection models, a branch of AI, are divided into the 1-stage detector that performs detection and classification at once and the 2-stage detector that separates detection and classification, and various models such as simple CNN [29–35] and R-CNN [32,36–41], YOLO [36–38,42,43] have been used, while new models such as lean CNN [44] and convolutional recurrent neural network (CRNN) [45] are also being developed.

The 1-stage detector is suitable for real-time processing because it detects and classifies at once, and has recently shown excellent performance in terms of accuracy. Conversely, the 2-stage detector is a traditional object detection model that has been mainly used for analysis of already collected data due to its relatively slow processing speed but high accuracy. Following these conflicting features, studies have also been conducted to compare the two detector models, along with studies for each model. Madhusri Maity et al. [38] comprehensively reviewed Faster R-CNN and YOLO-based vehicle detection and tracking methods. Benjdira et al. [36] compared the performance of CNN-based Fast R-CNN and YOLOV3 using five metrics: precision, recall, processing speed, etc., for vehicle detection through aerial images. Although both models showed high accuracy, they confirmed that the processing speed of YOLO V3 is higher than that of Fast R-CNN, so it had better performance. Esther Rani and Sri Jamiya [43] proposed the LittleYOLO-SPP Algorithm based on the YOLO v3-tiny network for real-time vehicle detection, achieving 77.44% mAP on the PASCAL VOC dataset. Danilo Avola et al. [41] introduced a multi-stream Fast-RCNN that performs multi-scale image analysis for UAV tracking and confirmed a more accurate detection performance than in existing R-CNN and Faster R-CNN.

CNN-based object detection has been conducted in various studies, such as modifying parameters or combining various analytical methods to improve accuracy and processing speed. Hu et al. [29] proposed a cascade vehicle detection method that combined CNN and various methods such as LBP, Haar-like, and HOG to improve the accuracy of vehicle detection through cameras in complex weather conditions, and 97.32% recall in complex driving environments indicates that the algorithm has good robustness. Another fused CNN method, hybrid CRNN-based network intrusion detection system (HCRN-NIDS) [45], is a detection method used in the field of information security, which combines RNN with CNN, showing excellent performance in detecting both local features and temperature features. Sanchez-Castro et al. [44] proposed lean CNN, which reduces parameters from the existing CNN and compared a total of six models by building a dataset consisting of vehicle types. As a result, the overall model showed more than 80% accuracy, and the optimal model considering accuracy and processing speed was also confirmed. Molina-Cabello et al. [33] proposed five CNN-based object detection and vehicle type classification models for traffic surveillance, and the proposed models consisted of three steps: object detection, tracking, and classification, and used five resizing region proposals. In particular, the centered scale method showed an accuracy of 87% and was found to be the best classification model.

AI-based object detection and distance estimation have also focused on pedestrian protection, serving as key techniques for computer vision-based pedestrian detection (PD) and distance estimation (DE) [34,35]. Dai et al. [40] proposed a 'novel multi-task Fast R-CNN' that simultaneously conducts distance estimation and pedestrian detection using improved ResNet-50 architecture with a processing speed of 7 FPS and with pedestrian detection accuracy of more than 80%. In addition, Strbac et al. [42] performed a camera-based stereoscopy measurement that completely excluded the use of LiDAR sensor through YOLO V3, and showed that it is useful for estimating distance within 20 m, showing that it can be applied to many advanced driver assistance systems (ADAS) such as automatic parking.

Although object detection and distance estimation studies of existing automated vehicles have been conducted through cameras and LiDAR, improving camera recognition techniques has become important due to the cost and recognition distance limitations of LiDAR. Furthermore, although existing research has been mainly conducted through stereo cameras, they fail to accurately measure three-dimensional coordinates. As a result, various research needs to be conducted, such as estimating three-dimensional coordinates through monocular cameras and improving the recognition technology of monocular cameras by applying AI to improve the recognition technology. Moreover, CNN shows good performance in image processing and various methods have been proposed depending on the application. In particular, as various models such as YOLO, a 1-stage detector, and R-CNN model, a 2-stage detector, are developed for object detection and distance estimation, it is believed that model selection suitable for automated driving systems based on accurate comparisons between models will be necessary.

Accordingly, in this study, to explore the application of deep learning-based CNN as one of the alternatives for the preventive design of automated driving systems, we aimed to select a model more suitable for the detection and classification of surrounding vehicles through a comparative analysis of existing CNN models and to contribute to prevention of accidents in automated vehicles through distance estimation between vehicles.

3. Methodology

The problem definition of object detection is to localize and classify an object. Traditional object detection methods are divided into region selection, feature extraction, and classification, requiring engineers to manually work on feature extraction themselves, and limited in handling complex and many images [46]. The development of deep learning complemented the limitations of existing methods, enabling deeper learning, and leading to improved performance. The general object detection method is divided into a region proposal-based 2-stage detector that follows the pipeline of localization and classification according to the traditional method and a 1-stage detector that performs detection and classification at once, based on regression [47]. There are R-CNN [16], spatial pyramid pooling in deep convolutional networks for recognition (SPP-Net) [48], Faster R-CNN [18], etc., and the 1-stage detector includes grid CNN(G-CNN) [49], YOLO [19], SSD [50], etc. The two types of object detection models have different processing processes, and they show differences in processing speed and accuracy depending on the data used. Table 1 compares the performance of each method for PASCAL VOC and Microsoft Common Objects in Context (MS COCO) dataset [51,52].

Table 1. The mAP comparison of object detection methods by PASCAL VOC and MS COCO.

Methods		Dataset	mAP
2-stage detector	R-CNN(VGG16)	VOC 2007	66
	SPP-Net(ZF)	VOC 2007	60.9
	Faster R-CNN	VOC 2007 + VOC 2012 + MS COCO	75.9
1-stage detector	G-CNN	VOC 2007	79.4
	YOLO V2	VOC 2007 + VOC 2012 + MS COCO	78.2
	SSD512	VOC 2007 + VOC 2012 + MS COCO	82.2

R-CNN: Regions with convolutional neural network; SPP: spatial pyramid pooling; G-CNN: grid convolutional neural network; MS COCO: microsoft common objects in context.

Although various state-of-the-art models, such as Mask R-CNN [53], have been developed in 2-stage detectors, the purpose of Mask R-CNN is image segmentation rather than object detection, so in this study, we used Faster R-CNN as a comparative model of a 2-stage detector. In addition, a 1-stage detector has a variety of models, including SSD, YOLO was selected as a comparative model of 1-stage detectors in this study because the computation process is easy, but the accuracy is similar to that of Fast R-CNN and the processing speed is slower than YOLO.

Therefore, in this study, we aimed to detect and classify vehicles from road driving videos collected through black boxes using Faster R-CNN, a 2-stage detector, and YOLO, a 1-stage detector, among CNN-based state-of-the-art methods, and to estimate the distance between vehicles using a model that is more suitable for automated driving systems. In this Section, the basic structure and principle of Faster R-CNN and YOLO V2 used in the study is described.

3.1. R-CNN

R-CNN, a network model for leveraging for object detection, has emerged as CNN shows superior performance in image classification. The flowchart of R-CNN, which was proposed by Grishick et al. [16], is shown in Figure 1 and R-CNN detects objects through region proposal, which creates segmentation on the image by using selective search and infers the location by drawing a box where the object is likely to be. Then, feature vectors are extracted from the images through pre-trained CNN, and classified class via each class-specific SVM classifier. However, R-CNN has disadvantages in that it learns in multiple steps, has a lot of computation, and has a slow processing speed as all region proposals have to pass CNN. To compensate for these disadvantages, SPP-Net [48], which extracts features by inputting the entire image into a pre-trained CNN, and Fast R-CNN [17], an end-to-end model that extracts a fixed-size feature vector through region of interest (ROI) pooling and then executes the remaining steps in a single pipeline, have been proposed.

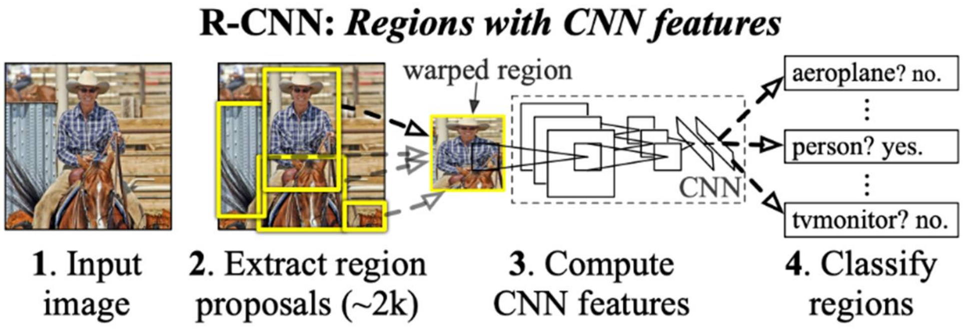

Figure 1. R-CNN flowchart. CNN: convolutional neural network.

3.2. Faster R-CNN

Faster R-CNN, an improved version of Fast R-CNN, is a model that removes selective search and performs the region proposal process through RPN. Faster R-CNN performs fewer operations than the existing selective search through RPN and improves processing speed and accuracy as it enables the use of GPU instead of CPU. In addition, Faster R-CNN simplifies the entire process by using an end-to-end model and has become one of the representative models of the 2-stage detector in which region proposal and classification are sequentially performed. The structure of Faster R-CNN proposed by Grishick et al. [18] is shown in Figure 2.

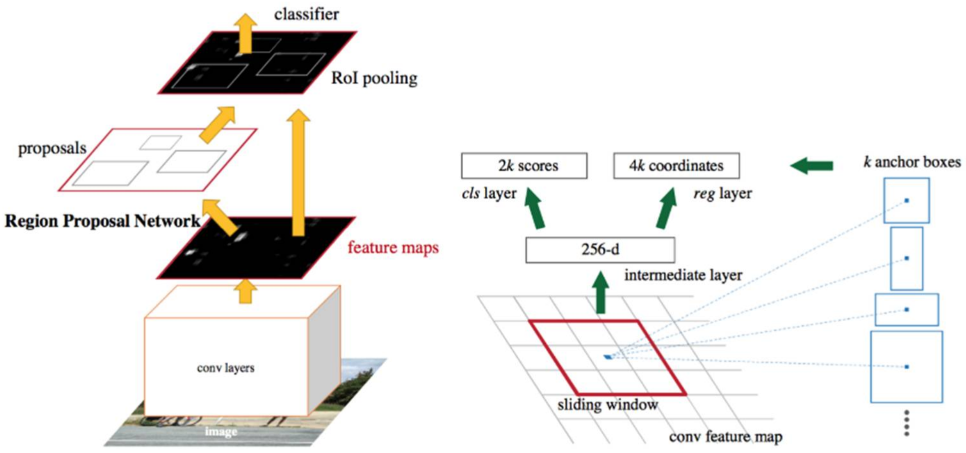

Figure 2. Faster R-CNN structure. R-CNN: Regions with convolutional neural network.

In addition, their proposed loss function formulation of Faster R-CNN is as follows:

$$L(\{p_i\},\ \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \quad (1)$$

Faster R-CNN is optimized for a multi-task loss function, which combines the losses of classification and bounding box regression. According to the authors, i is the anchor index in a mini-batch, p_i is the predicted probability that anchor i will be an object, and t_i is the vector representing the 4 parameterized coordinates of the predicted bounding box. * means the ground-truth label, where the region ratio intersects the prediction label can be calculated to determine the performance of the prediction through the intersection over union (IoU) formula. The classification loss L_{cls} is log loss over two classes (object vs. not object) and, for the regression loss, they used L_{reg} (t_i, t_i^*) = R(t_i − t_i^*) where R is the robust loss function. The two terms are normalized by N_{cls} and N_{reg} and weighted by a balancing parameter λ (defalut:10). Furthermore, to solve the class imbalance problem, which detects backgrounds more frequently than objects, the authors set the IoU criterion of non-positive anchors to 0.3, applying negative labeling, and random sampling IoU lower bound for all of these ground truth boxes. The layer parameters of ResNet-101 Architecture, the backbone network of Faster R-CNN, are shown in Table 2 [54].

Table 2. ResNet-101 Architectures layer parameters description.

Layer Name	No. of Units	Output Size
Convolution 1	[7 × 7, stride 2, channel 64] × 1	112 × 112
Convolution 2	[3 × 3 max pooling, stride 2] × 1	56 × 56
	[1 × 1, channel 64] × 3	
	[3 × 3, channel 64] × 3	
	[1 × 1, channel 256] × 3	
Convolution 3	[1 × 1, channel 128] × 4	28 × 28
	[3 × 3, channel 128] × 4	
	[1 × 1, channel 512] × 4	
Convolution 4	[1 × 1, channel 256] × 23	14 × 14
	[3 × 3, channel 256] × 23	
	[1 × 1, channel 1024] × 23	
Convolution 5	[1 × 1, channel 512] × 3	7 × 7
	[3 × 3, channel 512] × 3	
	[1 × 1, channel 2048] × 3	
Output	Average pooling, 1000d-fc, softmax	1 × 1

3.3. YOLO

In contrast to Faster R-CNN, which proceeds with region proposal and classification sequentially, YOLO is a 1-stage detector model that performs both processes simultaneously. YOLO predicts bounding boxes (B) and their confidence for each grid after dividing the input image into S × S grids. At this point, confidence is defined by multiplying Pr(object), the probability of an object being present, and IoU, which is the ratio of intersection area between the predicted Bounding box and ground truth. Each grid predicts one class (C) probability per grid regardless of the number of bounding boxes and multiplies the individual box confidence and conditional class probability. Finally, the multiplied class-specific confidence is encoded as an S × S × (B * 5 + C) tensor to check how well it fits the object and builds a network. The processing system of YOLO proposed by Redmon et al. [19] is shown in Figure 3.

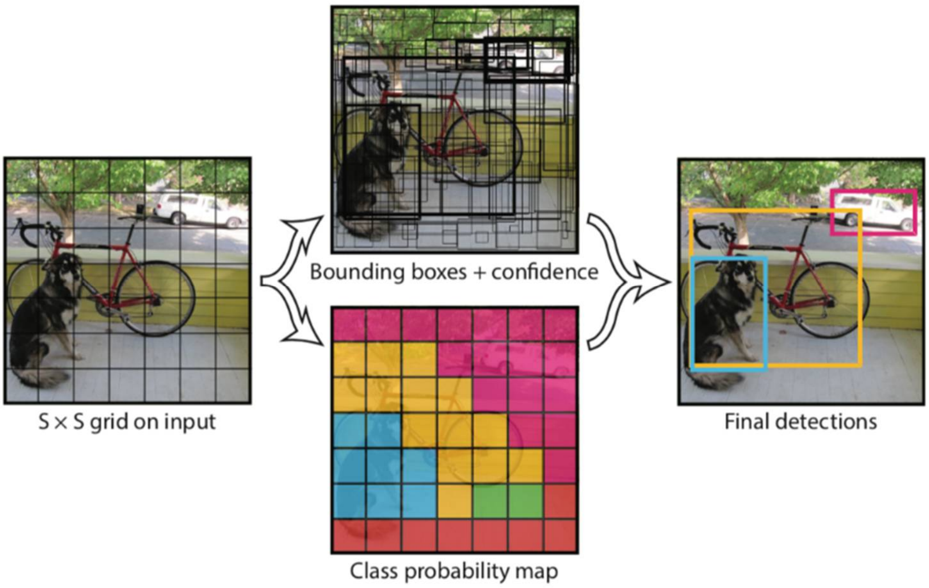

Figure 3. The processing system of YOLO. YOLO: You Only Look Once.

In addition, their proposed loss function formulation of YOLO is as follows:

$$\begin{aligned}
&\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\lessdot_{ij}^{obj}[(x_i-\hat{x}_i)^2+(y_i-\hat{y}_i)^2]\\
&+\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\lessdot_{ij}^{obj}\left[(\sqrt{w_i}-\sqrt{w_i})^2+\left(\sqrt{h_i}-\sqrt{\hat{h}_i}\right)^2\right]\\
&+\sum_{i=0}^{S^2}\sum_{j=0}^{B}\lessdot_{ij}^{obj}(C_i-\hat{C}_i)^2\\
&+\lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\lessdot_{ij}^{noobj}(C_i-\hat{C}_i)^2\\
&+\sum_{i=0}^{S^2}\lessdot_{ij}^{obj}\sum_{c\in classes}(p_i(c)-\hat{p}_i(c))^2
\end{aligned} \quad (2)$$

YOLO finds the bounding box contained in the final prediction, and uses sum-squared error between the predictions and the ground truth to calculate the loss. YOLO composes of localization loss, confidence loss, and classification loss. The localization loss, first and second line of (2), measures the errors in the locations and sizes of the predictive bounding box by counting the box responsible for detecting the object. In the formula, $\lessdot_{ij}^{obj}$ means that the j_{st} bounding box of the i featuring the object has produced the final prediction, otherwise, it is displayed as 0, and x, y, w, h refers to the x, y coordinates, width, and height of the bounding box, respectively. The authors predicted square roots of the width and height of the bounding box instead of width and height to differentiate the weight absolute errors of large boxes and small boxes and also multiplied the loss by λ_{coord} (default: 5) to further emphasize the boundary box accuracy. At this time, λ_{coord} increased the weight for the loss bounding box coordinates. In the formula, values x and y are simply calculated for simple differences, but since w and h are ratios, the difference is calculated by adding root. The confidence loss, third and fourth line of (2), measures the objectness of the box. Since most boxes do not contain objects, there is a class imbalance problem that detects backgrounds more frequently than objects. To address this issue, weight this loss down a λ_{noobj} factor with a default value of 0.5. In the formula, $\boldsymbol{C}$ is the box confidence score. The classification loss, the last line of (2), is the squared error of the class conditional probabilities for each class if an object is detected. the difference between the predicted value and the actual value for all classes is added to the exponential *i* where all objects are judged to be present. In the formula, *p(c)* denotes the conditional class probability for class ***c***.

YOLO has difficulty predicting small objects that appear in groups because the bounding box predicted by the grid can have only one class, and errors occur due to inaccurate localization. However, YOLO has made many advances in real-time object detection by showing faster processing speed than existing models. Since then, YOLO V2, which improves accuracy and enables more detection and classification by modifying the network and using fine-tuning and anchor box, was proposed [55]. Afterward, YOLO V3, which improves the backbone network structure for pre-learning, and YOLO V4, which combines various deep learning techniques, have been released, continuing to develop into an improved model. Since there is no significant change in the overall structure from YOLO V3, YOLO V2 was used in this study, and the layer parameters of DarkNet-19 Architecture, the backbone network of YOLO V2, are shown in Table 3 [55].

Table 3. DarkNet-19 Architecture layer parameters description.

Layer Type	Filters	Size/Stride	Output Size
Convolutional	32	3 × 3	224 × 224
Maxpool		2 × 2/2	112 × 112
Convolutional	64	3 × 3	
Maxpool		2 × 2/2	56 × 56
Convolutional	128	3 × 3	
Convolutional	64	1 × 1	
Convolutional	128	3 × 3	
Maxpool		2 × 2/2	28 × 28
Convolutional	256	3 × 3	
Convolutional	128	1 × 1	
Convolutional	256	3 × 3	
Maxpool		2 × 2/2	14 × 14
Convolutional	512	3 × 3	
Convolutional	256	1 × 1	
Convolutional	512	3 × 3	
Convolutional	256	1 × 1	
Convolutional	512	3 × 3	
Maxpool		2 × 2/2	7 × 7
Convolutional	1024	3 × 3	
Convolutional	512	1 × 1	
Convolutional	1024	3 × 3	
Convolutional	512	1 × 1	
Convolutional	1024	3 × 3	
Convolutional	1000	1 × 1	7 × 7
Avgpool		Global	1000
Softmax			

4. Vehicle Detection and Distance Estimation

In this Section, we performed vehicle detection and classification using Faster R-CNN and YOLO V2, and the learning environment is shown in Table 4.

4.1. Data Collecting and Pre-Training

Before this study, the accuracy and processing speed were compared by training the PASCAL VOC data set for model performance comparison. There was no significant difference in the accuracy of the training results, but YOLO V2 showed faster processing speed and was analyzed as shown in Table 5, respectively, in the real-time object detection part for the VOC 2007 dataset [51].

Table 4. Analysis Environment.

	Performance
Hardware	· Intel(R) Core(TM) i5-9400F · RAM 16 GB · Nvidia GeForce GTX 1650
Software	· Windows 10 · Python 3.7 · OpenCV 4.5 · Tensorflow 1.14
Model	· Faster R-CNN, YOLO V2

Table 5. Comparison Performance of Faster R-CNN and YOLO V2.

Model	Backbone	Dataset	mAP	FPS
Faster R-CNN	ResNet-101	Pascal VOC 2007	76.4	5
YOLO V2	DarkNet-19		78.6	40

The driving videos of automated vehicles are the most basic with which to detect a front object from various angles, but since it is difficult to obtain driving data, black box videos with relatively easy to collect data and those with the most similar collection location to those of automated vehicles were obtained and learned. The black box videos used for learning were collected under weekday daytimes with sunny weather conditions, consisting of 30 frames/s of 1920 × 1080 px size, and the extracted examples are shown in Figure 4. Before model comparison learning, we used the VOC dataset for pre-training, and, to handle the class imbalances of the dataset, the classes of dataset consisting of 20 classes and 20 K of data were removed except for the car, bus, and truck class, which may exist on the highway.

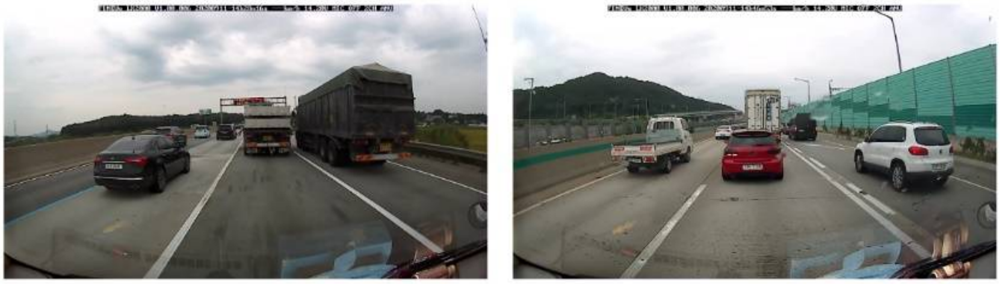

Figure 4. The example of Extracted Data.

4.2. Vehicle Detection and Classification

We input the collected data into the pre-trained Faster R-CNN and YOLOV2 models and set the threshold to 0.5 and 0.9, respectively, to proceed with comparative learning. The threshold is the critical value for the class-specific confidence of the bounding box and anchor box detected during the learning process, and only the bounding box, which is over the threshold, was outputted. As a result of the learning, as shown in Figures 5 and 6, both models did not detect the vehicles accurately when the threshold was set to 0.5, and when the threshold was set to 0.9, the vehicle could not be detected except for nearby vehicles. Therefore, based on the previous learning, we proceeded with the learning by setting the threshold as 0.7 finally, the median value.

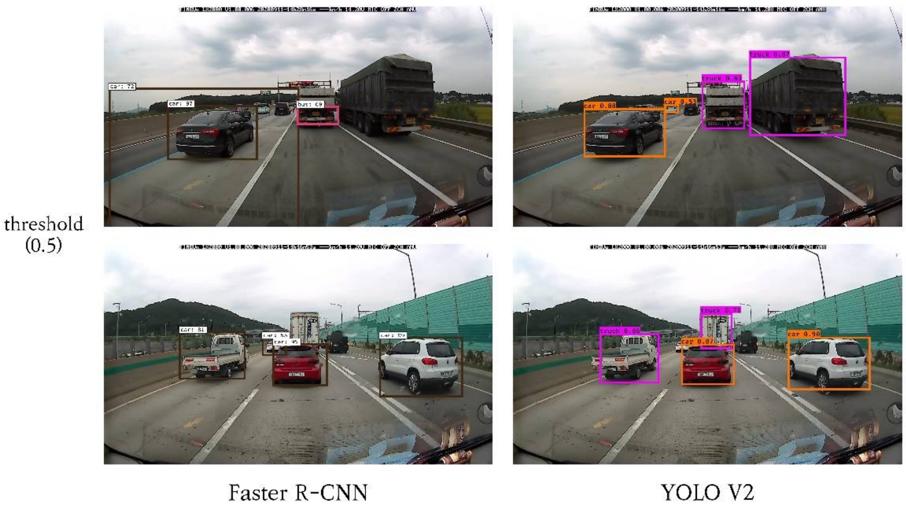

Figure 5. Learning results of object detection and classification—threshold 0.5.

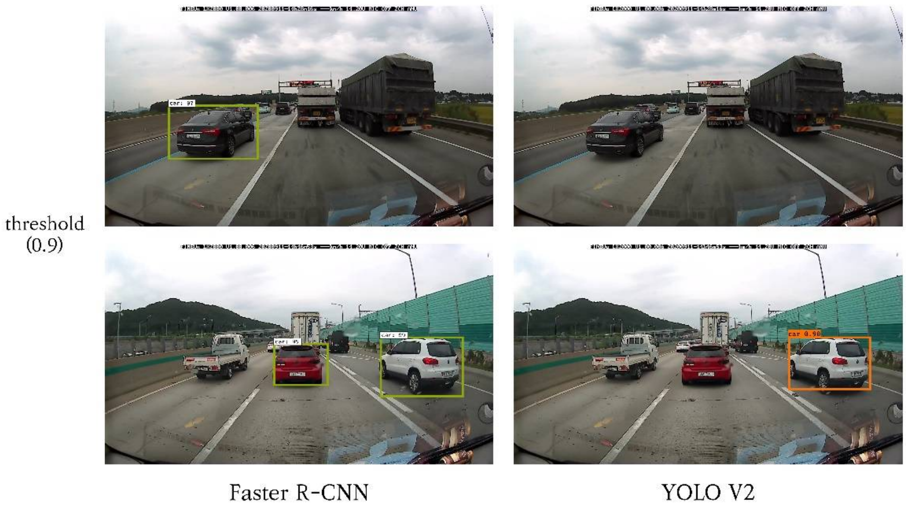

Figure 6. Learning results of object detection and classification—threshold 0.9.

As a result of model learning, Faster R-CNN showed high accuracy in classifying detected vehicles but had difficulty in detecting vehicles, and the processing speed was also analyzed approximately ten times slower compared to YOLO V2. YOLO V2 detected most of the vehicles on the frame and showed more than 90% accuracy for nearby vehicles, which was higher than Faster R-CNN. In particular, it is determined that the YOLO V2 is more suitable for classifying vehicles by detecting objects in real-time in automated driving systems, with a better processing speed of 5.5 FPS compared to low hardware performance, and thus, we want to further proceed with distance estimation through YOLO V2. The results of the comparative learning of the two models are shown in Figure 7.

4.3. Distance Estimation

To implement the distance estimation model using YOLO V2 selected by classification learning, we used previously collected black box videos and utilized video data without image extraction to implement real-time distance estimation. After entering the images into the pre-trained YOLO v2 model, we classified the probability of an object out of 80 classes using five anchor boxes and then output classes and accuracy for objects with a threshold of 0.7 or higher. To estimate the distance from the detected object, we used camera calibration [56] to calculate the parameter values of the 2D converted image captured by the camera, as shown in Figure 8, and re-extracted the 2D coordinate values to the 3D coordinate values using the calculated parameters and perspective transform. Next, we estimated the distance between the camera and the extracted 3D coordinate values through image warping and pixels per meter [57].

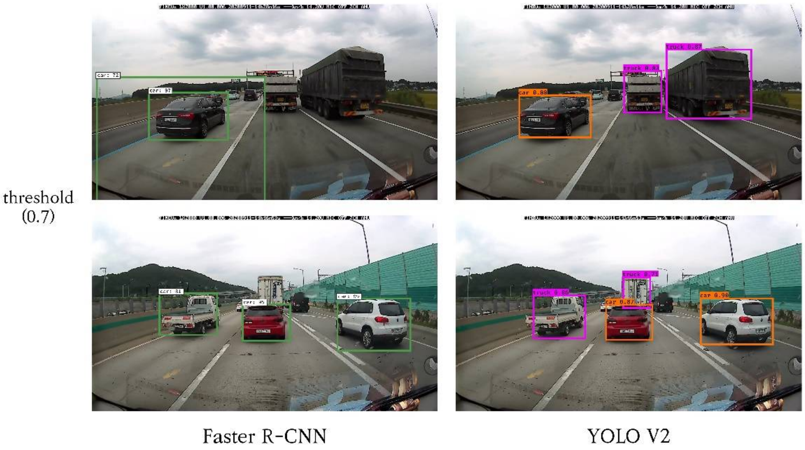

Figure 7. Learning results of object detection and classification—threshold 0.7.

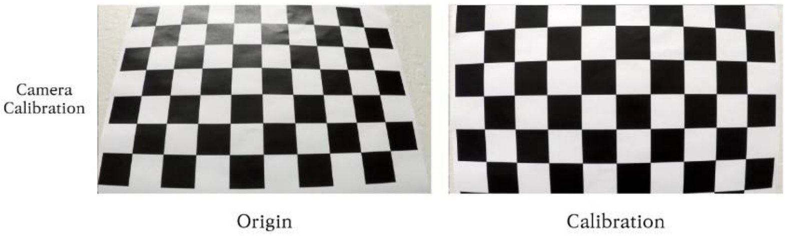

Figure 8. The process of distance estimation—camera calibration.

(i) After extracting the bounding box coordinates of the detected object, calculate the parameters using camera calibration.

(ii) Reshape the coordinate value to (1, 1, 2) for matrix operation, transform the coordinate value back to 3D through perspective transform, and then reshape it to (2,1).

(iii) Estimate and output the distance between the detected object and the camera through image warping and pixels per meter.

As a result, objects at a distance were difficult to detect with a threshold of 0.7, but object detection and classification were performed with more than 80% accuracy for objects at relatively close, and thus distance estimation was also performed accurately for detected objects. The final analysis results are the same as Figure 9.

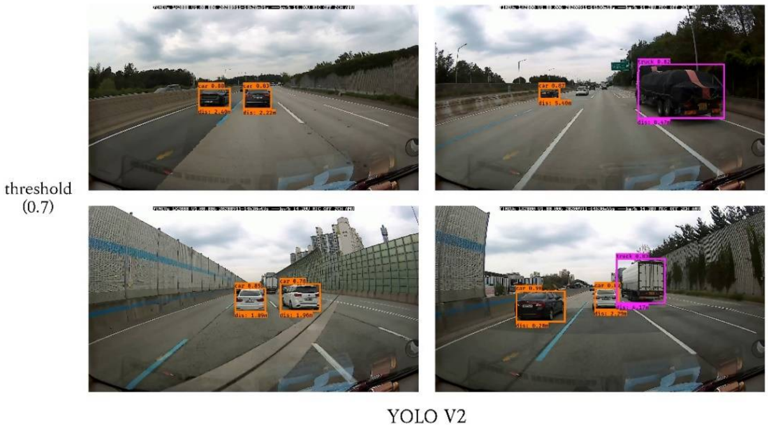

Figure 9. Learning Result of distance estimation.

5. Conclusions

To improve the recognition technique of vehicle-mounted monocular-camera for the design of preventive automated vehicles, this study employed CNN-based Faster R-CNN and YOLO V2 to recognize surrounding vehicles in black box highway driving videos and estimate distances from surrounding vehicles through more suitable models for automated driving systems. For analysis, black box videos of driving directly under sunny weather conditions during weekdays were collected, and pre-training was conducted. The analysis showed that Faster R-CNN had a similar accuracy, with a mAP of 76.4, as YOLO V2, with mAP of 78.6, but had a slower processing speed with an FPS of 5 compared to YOLOV2 with an FPS of 40, which had difficulty detecting. As a result, YOLO V2 was determined to be a more suitable model for real-time vehicle detection and classification, and further learned to estimate the distance between vehicles. For distance estimation, we conducted coordinate value conversion through camera calibration and perspective transform, set the threshold to 0.7, and performed object detection and distance estimation, showing more than 80% accuracy for near-distance vehicles.

In this study, 20 classes were simply reduced to three classes (car, bus, truck) which may exist on the highway, as there was a class imbalance problem with incorrect classification when pre-training with a set class of dataset. However, in the future, it is deemed necessary to use subdivided classes according to road environments and consider using an improved network that reflects additional data sampling methods (e.g., focal loss [58], gradient harmonizing mechanism [59]) to deal with class imbalance problems and learn more accurately. In addition, there are not many open source driving videos of automated vehicles, so the front black box videos of general vehicles, which are expected to be the most similar to the front camera of automated vehicles, were used instead, but it is expected that additional videos (e.g., the rear, the side black boxes, etc.) taken from various angles can be further trained to more accurately detect and classify vehicles, as well as to estimate the distance. In particular, in estimating the distance, the frame in the video is extracted from the camera to estimate the projected coordinate distance with the detected object for the stationary screen, and the speed of the currently running vehicle is not taken into account because the frame is again merged into the video and output. So, there is a limit in which the precision is somewhat lower in estimating the distance in real-time, and an error may occur in the estimated distance depending on the lane of the detected vehicle. Therefore, in the future, it is necessary to increase the precision of distance estimation by sensor fusion by conducting experiments on vehicles equipped with both cameras and LiDAR, and it is determined that additional research is needed on coordinate projection and distance estimation methods. The goal of future research includes the task of collecting data from vehicles containing both LiDAR and a black box to compare the accuracy of distance estimation.

This study is significant in that vehicle detection, classification, and distance estimation were performed by applying CNN, a deep learning network based on the most basic monocular camera data, for the preventive design of automated driving systems, and this study is expected to contribute to the commercialization of automated driving systems in the future, as basic materials for poor weather conditions (e.g., rain, snow, fog, etc.) to derive differentiation from LiDAR and radar sensors. Through this, it is believed that it will be able to help prevent accidents in automated vehicles, and it is expected that various accident prevention alternatives such as calculating and securing an appropriate safe distance according to vehicle type can be prepared through additional research in the future. In addition, it is expected to contribute to smooth traffic operations by quickly handling unexpected situations such as abnormal vehicle access by using CCTV or drones on the highway as well as cameras mounted on automated vehicles.

Author Contributions: Conceptualization, W.L.; software, W.L.; data curation, W.L.; investigation, M.-H.K., and J.S.; methodology, W.L.; project administration, M.-H.K.; writing—original draft, W.L.; writing—review and editing, M.-H.K.; K.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Ministry of Land, Infrastructure and Transport of Korea (grant 21AMDP-C161754-01) and 2021 Hongik University Research Fund.

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

References

1. Masmoudi, M.; Ghazzai, H.; Frikha, M.; Massoud, Y. Object detection learning techniques for autonomous vehicle applications. In Proceedings of the 2019 IEEE International Conference on Vehicular Electronics and Safety (ICVES), Cairo, Egypt, 4–6 September 2019; pp. 1–5.
2. Luettel, T.; Himmelsbach, M.; Wuensche, H.J. Autonomous ground vehicles—Concepts and a path to the future. *Proc. IEEE* **2012**, *100*, 1831–1839. [CrossRef]
3. Abuelsamid, S.; Alexander, D.; Jerram, L. *Navigant Research Leaderboard Report: Automated Driving*; Navigant Consulting, Inc.: Boulder, CO, USA, 2017.
4. Lee, B.Y. Domestic and foreign autonomous vehicle technology development trends and prospects. *Information and Communications Magazine*, 31 March 2016; Volume 33, 10–16.
5. Kaan, J. User Acceptance of Autonomous Vehicles: Factors & Implications. Master's Thesis, Delft University of Technology, Delft, The Netherlands, 28 August 2017.
6. Kim, K.; Kim, B.; Lee, K.; Ko, B.; Yi, K. Design of integrated risk management-based dynamic driving control of automated vehicles. *IEEE Intell. Transp. Syst. Mag.* **2017**, *9*, 57–73. [CrossRef]
7. Arnold, E.; Al-Jarrah, O.Y.; Dianati, M.; Fallah, S.; Oxtoby, D.; Mouzakitis, A. A survey on 3d object detection methods for autonomous driving applications. *IEEE Trans. Intell. Transp. Syst.* **2019**, *20*, 3782–3795. [CrossRef]
8. Lee, J.Y.; Chung, J.H.; Son, B.S. Analysis of traffic accident severity for Korean highway using structural equations model. *J. Korean Soc. Transp.* **2008**, *26*, 17–24.
9. Chen, Y.-L.; Wang, C.-A. Vehicle safety distance warning system: A novel algorithm for vehicle safety distance calculating between moving cars. In Proceedings of the 2007 IEEE 65th Vehicular Technology Conference-VTC2007-Spring, Dublin, Ireland, 22–25 April 2007; pp. 2570–2574.
10. Zaarane, A.; Slimani, I.; Al Okaishi, W.; Atouf, I.; Hamdoun, A. Distance measurement system for autonomous vehicles using stereo camera. *Array* **2020**, *5*, 100016. [CrossRef]
11. Li, P.; Chen, X.; Shen, S. Stereo r-cnn based 3d object detection for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 7644–7652.
12. Chen, X.; Kundu, K.; Zhang, Z.; Ma, H.; Fidler, S.; Urtasun, R. Monocular 3d object detection for autonomous driving. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 2147–2156.
13. LeCun, Y.; Bengio, Y. Convolutional networks for images, speech, and time series. *Handb. Brain Theory Neural Netw.* **1995**, *3361*, 1995.
14. Tarmizi, I.A.; Abd Aziz, A. Vehicle Detection Using Convolutional Neural Network for Autonomous Vehicles. In Proceedings of the 2018 International Conference on Intelligent and Advanced System (ICIAS), Kuala Lumpur, Malaysia, 13–14 August 2018; pp. 1–5.
15. Babiker, M.A.; Elawad, M.A.; Ahmed, A.H. Convolutional Neural Network for a Self-Driving Car in a Virtual Environment. In Proceedings of the 2019 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE), Khartoum, Sudan, 21–23 September 2019; pp. 1–6.
16. 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, 23–28 June 2014; pp. 580–587.
17. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1440–1448.
18. 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. Available online: https://arxiv.org/abs/1506.01497 (accessed on 4 June 2015). [CrossRef]
19. 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.
20. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934. Available online: https://arxiv.org/abs/2004.10934 (accessed on 23 April 2020). [CrossRef]
21. Bhaskar, H.; Dwivedi, K.; Dogra, D.P.; Al-Mualla, M.; Mihaylova, L. Autonomous detection and tracking under illumination changes, occlusions and moving camera. *Signal Process.* **2015**, *117*, 343–354. [CrossRef]
22. Kehtarnavaz, N.; Griswold, N.C.; Eem, J.K. Comparison of mono-and stereo-camera systems for autonomous vehicle tracking. In Proceedings of the Applications of Artificial Intelligence IX, Orlando, FL, USA, 1 March 1991; Volume 1468, pp. 467–478.
23. Grimes, D.M.; Jones, T.O. Automotive radar: A brief review. *Proc. IEEE* **1974**, *62*, 804–822. [CrossRef]
24. Nabati, R.; Qi, H. Rrpn: Radar region proposal network for object detection in autonomous vehicles. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 3093–3097.
25. Kocić, J.; Jovičić, N.; Drndarević, V. Sensors and sensor fusion in autonomous vehicles. In Proceedings of the 2018 26th Telecommunications Forum (TELFOR), Belgrade, Serbia, 20–21 November 2018; pp. 420–425.

26. Zhao, X.; Sun, P.; Xu, Z.; Min, H.; Yu, H. Fusion of 3D LIDAR and camera data for object detection in autonomous vehicle applications. *IEEE Sens. J.* **2020**, *20*, 4901–4913. [CrossRef]
27. Rashed, H.; Ramzy, M.; Vaquero, V.; El Sallab, A.; Sistu, G.; Yogamani, S. Fusemodnet: Real-time camera and lidar based moving object detection for robust low-light autonomous driving. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, Seoul, Korea, 27 October–2 November 2019.
28. Lai, Y.K.; Chou, Y.H.; Schumann, T. Vehicle detection for forward collision warning system based on a cascade classifier using adaboost algorithm. In Proceedings of the 2017 IEEE 7th International Conference on Consumer Electronics-Berlin (ICCE-Berlin), Berlin, Germany, 3–6 September 2017; pp. 47–48.
29. Hu, J.; Sun, Y.; Xiong, S. Research on the Cascade Vehicle Detection Method Based on CNN. *Electronics* **2021**, *10*, 481. [CrossRef]
30. Molina-Cabello, M.A.; Luque-Baena, R.M.; López-Rubio, E.; Thurnhofer-Hemsi, K. Vehicle type detection by convolutional neural networks. In Proceedings of the International Work-Conference on the Interplay Between Natural and Artificial Computation (IWINAC), Corunna, Spain, 19–23 June 2017; pp. 268–278.
31. Gao, H.; Cheng, B.; Wang, J.; Li, K.; Zhao, J.; Li, D. Object classification using CNN-based fusion of vision and LIDAR in autonomous vehicle environment. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4224–4231. [CrossRef]
32. Murali, A.; Nair, B.B.; Rao, S.N. Comparative Study of Different CNNs for Vehicle Classification. In Proceedings of the 2018 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Madurai, India, 13–15 December 2018; pp. 1–4.
33. Molina-Cabello, M.A.; Luque-Baena, R.M.; Lopez-Rubio, E.; Thurnhofer-Hemsi, K. Vehicle type detection by ensembles of convolutional neural networks operating on super resolved images. *Integr. Comput. Aided Eng.* **2018**, *25*, 321–333. [CrossRef]
34. Joung, J.; Jung, S.; Chung, S.; Jeong, E.R. CNN-based Tx–Rx distance estimation for UWB system localisation. *Electron. Lett.* **2019**, *55*, 938–940. [CrossRef]
35. Mukherjee, A.; Adarsh, S.; Ramachandran, K.I. ROS-Based Pedestrian Detection and Distance Estimation Algorithm Using Stereo Vision, Leddar and CNN. In *Intelligent System Design*; Springer: Singapore, 2020; pp. 117–127.
36. Benjdira, B.; Khursheed, T.; Koubaa, A.; Ammar, A.; Ouni, K. Car detection using unmanned aerial vehicles: Comparison between faster r-cnn and yolov3. In Proceedings of the 2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS), Muscat, Oman, 5–7 February 2019; pp. 1–6.
37. Ammar, A.; Koubaa, A.; Ahmed, M.; Saad, A. Aerial images processing for car detection using convolutional neural networks: Comparison between faster r-cnn and yolov3. *arXiv* **2019**, arXiv:1910.07234. Available online: https://arxiv.org/abs/1910.07234 (accessed on 16 October 2019). [CrossRef]
38. Maity, M.; Banerjee, S.; Chaudhuri, S.S. Faster R-CNN and YOLO based Vehicle detection: A Survey. In Proceedings of the 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 8–10 April 2021; pp. 1442–1447.
39. Hsu, S.C.; Huang, C.L.; Chuang, C.H. Vehicle detection using simplified fast R-CNN. In Proceedings of the 2018 International Workshop on Advanced Image Technology (IWAIT), Chiang Mai, Thailand, 7–9 January 2018; pp. 1–3.
40. Dai, X.; Hu, J.; Zhang, H.; Shitu, A.; Luo, C.; Osman, A.; Sfarra, S.; Duan, Y. Multi-task faster R-CNN for nighttime pedestrian detection and distance estimation. *Infrared Phys. Technol.* **2021**, *115*, 103694. [CrossRef]
41. Avola, D.; Cinque, L.; Diko, A.; Fagioli, A.; Foresti, G.L.; Mecca, A.; Pannone, D.; Piciarelli, C. MS-Faster R-CNN: Multi-Stream Backbone for Improved Faster R-CNN Object Detection and Aerial Tracking from UAV Images. *Remote Sens.* **2021**, *13*, 1670. [CrossRef]
42. Strbac, B.; Gostovic, M.; Lukac, Z.; Samardzija, D. YOLO Multi-Camera Object Detection and Distance Estimation. In Proceedings of the 2020 Zooming Innovation in Consumer Technologies Conference (ZINC), Novi Sad, Serbia, 26–27 May 2020; pp. 26–30.
43. Rani, E.; Jamiya, S. LittleYOLO-SPP: A delicate real-time vehicle detection algorithm. *Optik* **2021**, *225*, 165818.
44. Sanchez-Castro, J.J.; Rodríguez-Quiñonez, J.C.; Ramírez-Hernández, L.R.; Galaviz, G.; Hernández-Balbuena, D.; Trujillo-Hernández, G.; Flores-Fuentes, W.; Mercorelli, P.; Hernández-Perdomo, W.; Sergiyenko, O.; et al. A Lean Convolutional Neural Network for Vehicle Classification. In Proceedings of the 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, The Netherlands, 17–19 June 2020; pp. 1365–1369.
45. Khan, M.A. HCRNNIDS: Hybrid Convolutional Recurrent Neural Network-Based Network Intrusion Detection System. *Processes* **2021**, *9*, 834. [CrossRef]
46. O'Mahony, N.; Campbell, S.; Carvalho, A.; Harapanahalli, S.; Hernandez, G.V.; Krpalkova, L.; Riordan, D.; Walsh, J. Deep learning vs. traditional computer vision. In Proceedings of the Science and Information Conference, Las Vegas, NV, USA, 2–3 May 2019; pp. 128–144.
47. Zhao, Z.; Zheng, P.; Xu, S.; Wu, X. Object Detection with Deep Learning: A Review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3212–3232. [CrossRef] [PubMed]
48. 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*, 1–14. [CrossRef]
49. Lu, Q.; Liu, C.; Jiang, Z.; Men, A.; Yang, B. G-CNN: Object detection via grid convolutional neural network. *IEEE Access* **2017**, *5*, 24023–24031. [CrossRef]
50. 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.

51. 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]
52. 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; pp. 740–755.
53. 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. 2961–2969.
54. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Cmputer Vsion and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
55. 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.
56. Schoepflin, T.N.; Dailey, D.J. Dynamic camera calibration of roadside traffic management cameras. In Proceedings of the IEEE 5th International Conference on Intelligent Transportation Systems, Singapore, 3–6 September 2002; pp. 25–30.
57. Wolberg, G. *Digital Image Warping*; IEEE Computer Society Press: Los Alamitos, CA, USA, 1990.
58. 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; pp. 2980–2988.
59. Li, B.; Liu, Y.; Wang, X. Gradient harmonized single-stage detector. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 8577–8584.