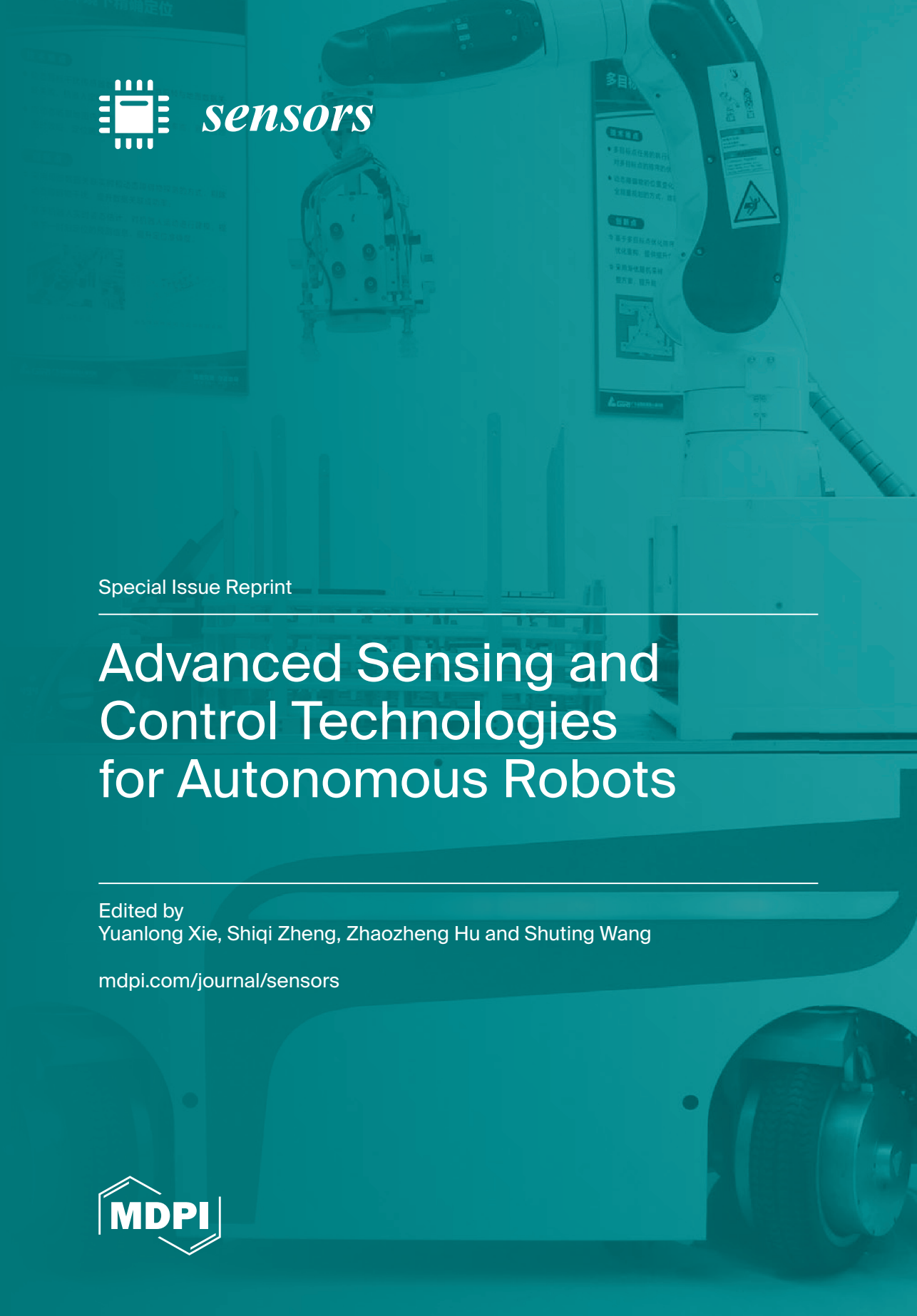


sensors

Special Issue Reprint

Advanced Sensing and Control Technologies for Autonomous Robots

Edited by
Yuanlong Xie, Shiqi Zheng, Zhaozheng Hu and Shuting Wang

mdpi.com/journal/sensors

MDPI

Advanced Sensing and Control Technologies for Autonomous Robots

Advanced Sensing and Control Technologies for Autonomous Robots

Guest Editors

Yuanlong Xie
Shiqi Zheng
Zhaozheng Hu
Shuting Wang

MDPI

Basel • Beijing • Wuhan • Barcelona • Belgrade • Novi Sad • Cluj • Manchester

Guest Editors

Yuanlong Xie
School of Mechanical Science and Engineering
Huazhong University of Science and Technology
Wuhan
China

Shiqi Zheng
School of Automation
China University of Geosciences
Wuhan
China

Zhaozheng Hu
ITS Research Center
Wuhan University of Technology
Wuhan
China

Shuting Wang
School of Mechanical Science and Engineering
Huazhong University of Science and Technology
Wuhan
China

Editorial Office
MDPI AG
Grosspeteranlage 5
4052 Basel, Switzerland

This is a reprint of the Special Issue, published open access by the journal *Sensors* (ISSN 1424-8220), freely accessible at: https://www.mdpi.com/journal/sensors/special_issues/sensing_control_autonomous_robots.

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

Lastname, A.A.; Lastname, B.B. Article Title. *Journal Name* **Year**, *Volume Number*, Page Range.

ISBN 978-3-7258-3183-8 (Hbk)
ISBN 978-3-7258-3184-5 (PDF)
https://doi.org/10.3390/books978-3-7258-3184-5

Cover image courtesy of Yuanlong Xie

Contents

About the Editors

Yuanlong Xie

Prof. Yuanlong Xie began and completed his Ph.D. degree in mechanical engineering from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2014 and 2018, respectively. He was an Academic Visitor with the School of Electronic and Electrical Engineering, University of Leeds, Leeds, U.K., from 2017 to 2018. In Nov. 2018, he joined the HUST as a Postdoctoral Fellow and became a Lecturer in Aug. 2021. He has published more than 100 academic journal and conference papers, and holds more than 50 patents. He serves as a guest editor of some leading international journals, including *Sensors* and *IET Collaborative Intelligent Manufacturing*. His research interests include mobile robots, optimization calculation, and optimal control.

Shiqi Zheng

Prof. Shiqi Zheng received his Ph.D. degree from Huazhong University of Science and Technology, China in 2016. He was a Visiting Scholar with the University of Adelaide, Adelaide, SA, Australia. He is currently a Professor with the China University of Geosciences, Wuhan, China. He is also with the Hubei Key Laboratory of Advanced Control and Intelligent Automation for Complex Systems and the Engineering Research Center of Intelligent Technology for Geo-Exploration, Ministry of Education, Wuhan, China. His research interests include intelligent adaptive control for complex systems, such as fractional-order systems, stochastic systems, and controller tuning. Dr. Zheng is a Member of the Chinese Association of Automation (CAA) and a Committee Member of Fractional Order Systems and Control (FOSC).

Zhaozheng Hu

Prof. Zhaozheng Hu is a distinguished professor and doctoral supervisor of "Chutian Scholars" in Hubei Province of Wuhan University of Technology and is the deputy director of Hubei Provincial Key Laboratory of Transportation Internet of Things Technology. He has published more than 100 academic papers, including more than 30 SCI international journal papers as the first author or corresponding author. In recent years, he has presided over more than 30 projects, such as the National Natural Science Foundation of China, the Hubei Provincial Key R&D Program, the Hubei Provincial Technological Innovation Major Project, and the National Key R&D Program Project/Sub-project. He is mainly engaged in the research of 3D computer vision, intelligent vehicle–road systems, vision and laser positioning (SLAM), intelligent transportation system, etc.

Shuting Wang

Prof. Shuting Wang received a Ph.D. degree from the School of Mechanical Science and Engineering, Huazhong University of Science and Technology (HUST), Wuhan, China, in 2002. He is currently a full professor and vice-president of the School of Mechanical Science and Engineering of HUST. He has published more than 100 articles. His research interests include mobile robots, optimization calculation, optimal control, and topology optimization.

sensors

MDPI

Editorial

Advanced Sensing and Control Technologies for Autonomous Robots

Yuanlong Xie [1,*], Shuting Wang [1], Shiqi Zheng [2] and Zhaozheng Hu [3]

[1] School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China; wangst@hust.edu.cn
[2] School of Automation, China University of Geosciences, Wuhan 430074, China; zhengshiqi@cug.edu.cn
[3] ITS Research Center, Wuhan University of Technology, Wuhan 430063, China; zzhu@whut.edu.cn
* Correspondence: yuanlongxie@hust.edu.cn

1. Introduction

The development of advanced sensing and control technologies provides increased intelligence and autonomy for robots and enhances the robots' agility, maneuverability, and efficiency, which has attracted growing attention in various industries and domains [1–3], including manufacturing [4], logistics [5], and warehousing [6].

Information about the environmental and working conditions is crucial for the safe navigation of autonomous robots [7]. Sensors play a vital role in providing data for navigation, such as map construction [8], self-localization [9], and obstacle detection [10]. The advanced sensing technologies, benefiting from the development of sophisticated sensors and innovative perception algorithms, has enabled autonomous robots to perceive accurate information with faster speed and higher precision [11,12]. This has facilitated the utilization of robots in complex scenarios [13]. A well-designed control algorithm is essential for the precision and stable navigation of an autonomous robot system. Control technology encompasses system modeling [14], parameter identification [15], tracking control [16], and cooperative control [17]. As advanced sensors and control technologies develop, autonomous robots complete tasks individually or in coordinated swarms, offering unprecedented flexibility across diverse fields of application [18–20]. These applications have expanded beyond traditional manufacturing and automation settings.

In the Special Issue titled "Advanced Sensing and Control Technologies for Autonomous Robots", the latest research in cutting-edge fields and emerging topics is presented. This Special Issue invited diverse research contributions from scholars engaged in related fields, including design, modeling, and parameter identification of novel robotic systems; advanced perception, self-localization, and map-building technologies based on neural networks; intelligent path planning and formation control methods for multi-agent systems. After a rigorous review, fourteen papers, including thirteen articles and one review, were selected. They offer original ideas and feasible solutions to address critical issues in the field of autonomous robots and explore the application of the systems in smart manufacturing, autonomous driving, housekeeping, etc. Each article contributes to the advancement of specific areas, and we recommend that readers explore these articles in detail to gain a comprehensive understanding. It is our hope that this Special Issue contributes to the continued development of the robotics field.

2. Overview of the Published Articles

The first contribution presents a new approach to controller design for nonlinear cascade systems. These systems pose a challenge in engineering applications due to their complex dynamics and inherent uncertainty. To address this, the authors estimate the nonlinear characteristics of each subsystem by separating the steady and alternating components, using local tracking errors for model-free adaptive control. Additionally,

Citation: Xie, Y.; Wang, S.; Zheng, S.; Hu, Z. Advanced Sensing and Control Technologies for Autonomous Robots. *Sensors* **2024**, *24*, 5478. https://doi.org/10.3390/s24175478

Received: 3 July 2024
Revised: 29 July 2024
Accepted: 4 August 2024
Published: 23 August 2024

they employ a square error correction procedure to approximate the weight coefficients and mitigate the uncertainty. This straightforward method can be readily implemented in engineering practice, providing a practical alternative to existing schemes.

In Contribution 2, Hongchao Zhang and co-authors present an advanced method for estimating the peak adhesion coefficient on unstructured pavements, which is typically challenging due to uneven surfaces and steep slopes. The authors propose an approach based on the extended Kalman filter, improving the accuracy of identifying road adhesion coefficients by incorporating an equivalent suspension model. This model optimizes the vertical wheel load calculations and adjusts the vehicle acceleration using posture data. Through multi-condition simulations with CarSim, the method demonstrates an accuracy improvement of at least 3.6%, verifying the precision and effectiveness of the designed algorithm. This robust and efficient method proves particularly beneficial for autonomous driving scenarios, enabling accurate ground adhesion data collection and sharing. It showcases the method's practical applicability and significant potential for real-world implementation, making it a valuable contribution to the field of mobile robotics and autonomous driving.

This Special Issue presents a multifaceted exploration of SLAM technology, the foundational technology in robotics. Among the articles, Contribution 3 proposes a solution to SLAM's difficulty in maintaining high localization accuracy over time in complex scenarios. Inspired by human neuroscience, the authors introduce NeoSLAM, a new scheme that incorporates a computational model of the brain into the traditional SLAM framework. To enhance the system's robustness to environmental disturbances, the authors employ a hierarchical temporal memory model, optimizing the real-time performance. This innovative approach improves the adaptability of SLAM systems to dynamic environments.

A novel model for swarm behavior control is introduced through artificial empathy in Contribution 4. The authors emphasize the importance of incorporating empathy into swarm control for optimized performance and the learning mechanisms within the swarm. The research aims to validate the artificial empathy model through simulations and further development, with the goal of accurately recreating human empathy. By utilizing fuzzy set theory and similarity measures, the model emulates human empathetic decision processes in swarm control. It focuses on knowledge representation, decision-making processes, and empathetic communication, providing a comprehensive and effective framework compared to existing models. The research also highlights the significance of an open-source physical-based experimentation platform for evaluating various models and scenarios in robotics. The work lays the foundation for further exploration and development of empathetic swarm control models in diverse environments.

Contribution 5, by Da Jiang and co-authors, addresses the problem of integrated open space risk in architectural scenarios of unmanned vehicles. The authors propose an adaptive dynamic windowing approach that includes a specially designed multi-objective speed sampling window and a layered decision-making mechanism. This method enables obstacle avoidance in multiple driving modes, with speed planning commands constrained within a reasonable range. The core concept of the method is the introduction of an adaptive prediction horizon and a critical rollover speed window, which dynamically adjusts the window in high-risk environments to ensure planning safety. This approach enhances safety and effectively avoids the risks of instability associated with fast steering maneuvers.

Yongchao Zhang and his colleagues propose novel insights to improve the performance of SLAM in complex dynamic environments in Contribution 6. Their primary approach is to impose hierarchical constraints on dynamic features through instance segmentation and multi-view constraints, preserving robust static features. The SOLOv2 instance segmentation algorithm is utilized to eliminate dynamic and potentially dynamic features, retaining reliable static features and generating a robust base matrix. The article also examines how the target semantic information obtained from the instance segmentation algorithm can be fused with a 3D semantic point cloud to create a 3D octane semantic map containing instance-level semantic information. This approach enhances the robot's

perception ability and understanding of the surrounding environments, improving the adaptability to dynamic environments.

As presented in Contribution 7, Zhang and Chen offer valuable insights into the real-time navigation and decision making in unpredictable environments of mobile robots. The authors present an enhanced version of the SAC-LSTM algorithm, incorporating a burn-in mechanism and a prioritized experience replay (PER) mechanism. The effectiveness of this algorithm is illustrated in a simulation environment. The integration of the burn-in training approach to counteract the memory degradation and the utilization of prioritized experience replay to enhance the sampling efficiency results in substantial improvements in the convergence speed and planning accuracy. This research provides a robust framework for mobile robot navigation and sets a new benchmark for future research in autonomous systems, inspiring advances and practical implementations.

The article by Jae-Bong and co-authors (Contribution 8) focuses on quadruped robots with robotic arms. The authors aim to design a quadruped mobile robot system that integrates perception, navigation, movement, and manipulation for item cleanup in households. The designed system, as presented in the article, is divided into three modules: perception, behavior control, and joint control. In the perception module, machine learning methods, specifically YOLOv7, KNN, and RANSAC, are employed to detect objects in real time and generate point clouds, enabling the estimation of optimal grasping postures. The behavior control module utilizes SLAM to construct an environment map, while the AMCL navigation package in ROS facilitates precise navigation. In the joint control module, MPC is employed to regulate the four legs, and the inverse kinematics solution controls the position of the low-degree-of-freedom manipulator. The simulation results in a virtual environment demonstrate that the authors' system achieves an average success rate of 96% for different object classifications, indicating good stability and accuracy. However, it is important to note that the inherent advantages of the quadruped robot are not fully demonstrated due to the flat terrain of the housing environment. It is suggested that the authors consider these relevant aspects in future work.

Ahmed Neaz focuses on the development of an advanced omnidirectional mobile robot designed to enhance industrial logistics tasks in dynamic and heavy-traffic environments. Contribution 9 presents an integrated control system that combines high-level and low-level algorithms with a graphical interface for each system. The low-level motor control is achieved using an efficient microcontroller to ensure high accuracy and robustness. High-level decision making is managed by a Raspberry Pi 4 and a remote PC, utilizing multiple LiDAR sensors, IMU, and odometry data from wheel encoders. The low-level programming is implemented in LabVIEW, while the high-level software architecture uses ROS. The practical operation of the 'MotionBot' robot demonstrates the reliability and effectiveness of the proposed techniques. Enhancements in lower-level control optimize the vibrations and increase the stability with a low computational cost. The robot showcases robustness across various environments and loads, and the fusion of data from three LiDAR sensors significantly improves the localization and positioning accuracy. The robot successfully navigates to the target points while avoiding obstacles in dynamic settings, supported by a user-friendly GUI developed in LabVIEW. Overall, this study presents a comprehensive and robust solution for autonomous navigation and mapping in industrial logistics, with significant potential for real-world applications.

Contribution 10 explores the distributed containment control of continuous-time linear multi-agent systems. The authors introduce a novel parametric dynamically compensated distributed control protocol that incorporates information from both the virtual layer and the actual neighboring agents. By adjusting the dominant poles using MLQR optimal control and Geršgorin's circle criterion, the containment control achieves a specified convergence speed. Furthermore, the authors address the adaptability of the dynamic control protocol in the event of the failure of the virtual layer, ensuring that the convergence speed can still be guaranteed through dominant pole assignment methods. The study emphasizes the importance of dynamic performance tuning, the design of distributed control protocols

for MASs with multiple leaders over fixed topology, and the significance of the convergence speed for the system's stability and performance.

In Contribution 11, Jie Meng and co-authors consider the relationship between perception and control. They propose a novel MPC framework to enhance the tracking accuracy in the presence of localization fluctuations. The prediction horizon is time-varying, following the localization accuracy, mitigating the effects of localization fluctuations, and improving the tracking accuracy and system stability in dynamic environments. The authors also introduce a methodology for assessing the fluctuations in localization, employing fuzzy rules to estimate the variance and entropy, providing a means of quantifying the magnitude of localization fluctuations. To reduce the computational burden and ensure the feasibility of the optimization problem, a Taylor expansion is employed to linearize the kinematic model of the robot. This approach represents a novel conceptual framework compared to the traditional method of decoupling perceptual and control methods, as it effectively enhances the tracking performance of the robot in the presence of perceptual uncertainty.

A simple and easy-to-implement localization system is essential for robots. Therefore, in Contribution 12, a visual tag-based indoor localization system is proposed. It is important to construct the tags straightforwardly to minimize the algorithm's complexity. However, this often leads to non-uniqueness issues with the tags. To address this, the authors propose a methodology for efficient tag matching based on the azimuth angle between the camera and the tag. Additionally, a method for estimating the tag's position is devised to achieve an optimal balance between computational complexity and positioning accuracy.

Huma Mahboob introduces a novel approach to autonomous navigation in unknown environments in Contribution 13. Instead of focusing on the local optimization of individual robots, the proposal aims to optimize the overall energy consumption of a population of robots. An innovative energy- and information-aware management algorithm is proposed, enabling each robot to draw and update a map of the entire environment by receiving information broadcasted by other robots. By comparing this with their own environment perception, the robots can determine their position on the map, enabling real-time localization. By guiding the robots' perception of their optimal paths, sharing sensory information among followers, and comparing energy consumption under information transfer and collaborative perception, the robot swarm can reach its destination with minimal energy consumption. This approach offers valuable insights for the advanced fields, with a notably low overhead in collaborative perception, real-time mapping, localization, and navigation.

The final contribution is a review by Yu Liu and co-authors, which presents the technologies used for perception by indoor autonomous mobile robots. The authors emphasize the significance of perception in mobile robotics and the need for accurate and efficient sensing capabilities to make informed decisions. The review provides a systematic literature review, covering various techniques for robot localization, including inertial navigation, GPS, navigation based on beacons or landmarks, and model matching. The article also discusses map-building techniques, with a focus on SLAM methods, including filter-based and graph-optimization-based algorithms. Additionally, the importance of LiDAR and vision cameras in SLAM, as well as the processing of SLAM optimization algorithms, is highlighted. In conclusion, the review emphasizes the importance and application of perception techniques, localization methods, and map-building algorithms for mobile robots operating in indoor environments.

3. Conclusions

This collection of papers on advanced sensing and control technologies for autonomous robots provides valuable insights into the current state of research and the state of the art in the field. The authors' contributions provide distinctive perspectives and innovative solutions to address pressing challenges in robotics. Building on the preceding discourse, several potential avenues warrant further exploration to establish robust and reliable navigation capabilities in sensing and control technologies. To this end, the following recommendations are presented for consideration:

- **Multi-sensor data fusion:** The inherent limitations of a single sensor pose significant challenges in the deployment of autonomous robots in complex scenarios, such as intelligent transport systems. Consequently, the integration and analysis of data from multiple sensor sources are imperative to generate comprehensive, accurate, and reliable information. This will involve addressing data heterogeneity and uncertainties from a broad range of sensors, ensuring the integrity and trustworthiness of the fused information, and advancing the development of productive fusion algorithms to enable a rapid response when needed.
- **Perception-control coupling method:** Autonomous robots complete some specific tasks in which accurate control is required, such as safe navigation under interference, the sensor faults of aircrafts, and the palletizing tasks of industrial robots. In current robotics systems, the perception and control layers often operate as distinct components. Notably, the localization results of the robot are typically considered accurate within the control framework. However, in practical applications, external errors persist, whether associated with LiDAR-based or vision-based localization strategies. In the absence of an appropriate strategy on the part of the controller, these errors can lead to severe system instability or substantial tracking deviations. Therefore, formulating a well-structured perception-control method becomes paramount to effectively mitigate such problems.

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

List of Contributions

1. Liu, Z.; Jin, H.; Zhao, J. An Adaptive Control Scheme Based on Non-Interference Nonlinearity Approximation for a Class of Nonlinear Cascaded Systems and Its Application to Flexible Joint Manipulators. *Sensors* **2024**, *24*, 3178. https://doi.org/10.3390/s24103178.
2. Zhang, H.; Song, B.; Xu, J.; Li, H.; Li, S. Adhesion Coefficient Identification of Wheeled Mobile Robot under Unstructured Pavement. *Sensors* **2024**, *24*, 1316. https://doi.org/10.3390/s24041316.
3. Pizzino, C.A.P.; Costa, R.R.; Mitchell, D.; Vargas, P.A. NeoSLAM: Long-Term SLAM Using Computational Models of the Brain. *Sensors* **2024**, *24*, 1143. https://doi.org/10.3390/s24041143.
4. Siwek, J.; Żywica, P.; Siwek, P.; Wójcik, A.; Woch, W.; Pierzyński, K.; Dyczkowski, K. Implementation of an Artificially Empathetic Robot Swarm. *Sensors* **2024**, *24*, 242. https://doi.org/10.3390/s24010242.
5. Jiang, D.; Wang, M.; Chen, X.; Zhang, H.; Wang, K.; Li, C.; Li, S.; Du, L. An Integrated Autonomous Dynamic Navigation Approach toward a Composite Air–Ground Risk Construction Scenario. *Sensors* **2024**, *24*, 221. https://doi.org/10.3390/s24010221.
6. Zhang, Y.; Li, Y.; Chen, P. TSG-SLAM: SLAM Employing Tight Coupling of Instance Segmentation and Geometric Constraints in Complex Dynamic Environments. *Sensors* **2023**, *23*, 9807. https://doi.org/10.3390/s23249807.
7. Zhang, Y.; Chen, P. Path Planning of a Mobile Robot for a Dynamic Indoor Environment Based on an SAC-LSTM Algorithm. *Sensors* **2023**, *23*, 9802. https://doi.org/10.3390/s23249802.
8. Yi, J.-B.; Nasrat, S.; Jo, M.-s.; Yi, S.-J. A Software Platform for Quadruped Robots with Advanced Manipulation Capabilities. *Sensors* **2023**, *23*, 8247. https://doi.org/10.3390/s23198247.
9. Neaz, A.; Lee, S.; Nam, K. Design and Implementation of an Integrated Control System for Omnidirectional Mobile Robots in Industrial Logistics. *Sensors* **2023**, *23*, 3184. https://doi.org/10.3390/s23063184.
10. Yan, F.; Feng, S.; Liu, X.; Feng, T. Parametric Dynamic Distributed Containment Control of Continuous-Time Linear Multi-Agent Systems with Specified Convergence Speed. *Sensors* **2023**, *23*, 2696. https://doi.org/10.3390/s23052696.

11. Meng, J.; Xiao, H.; Jiang, L.; Hu, Z.; Jiang, L.; Jiang, N. Adaptive Model Predictive Control for Mobile Robots with Localization Fluctuation Estimation. *Sensors* **2023**, *23*, 2501. https://doi.org/10.3390/s23052501.
12. Gao, F.; Ma, J. Indoor Location Technology with High Accuracy Using Simple Visual Tags. *Sensors* **2023**, *23*, 1597. https://doi.org/10.3390/s23031597.
13. Mahboob, H.; Yasin, J.N.; Jokinen, S.; Haghbayan, M.-H.; Plosila, J.; Yasin, M.M. DCP-SLAM: Distributed Collaborative Partial Swarm SLAM for Efficient Navigation of Autonomous Robots. *Sensors* **2023**, *23*, 1025. https://doi.org/10.3390/s23021025.
14. Liu, Y.; Wang, S.; Xie, Y.; Xiong, T.; Wu, M. A Review of Sensing Technologies for Indoor Autonomous Mobile Robots. *Sensors* **2024**, *24*, 1222. https://doi.org/10.3390/s24041222.

References

1. Zhao, J.; Liu, S.; Li, J. Research and Implementation of Autonomous Navigation for Mobile Robots Based on SLAM Algorithm under ROS. *Sensors* **2022**, *22*, 4172. [CrossRef]
2. Yue, X.; Li, H.; Shimizu, M.; Kawamura, S.; Meng, L. YOLO-GD: A Deep Learning-Based Object Detection Algorithm for Empty-Dish Recycling Robots. *Machines* **2022**, *10*, 294. [CrossRef]
3. Zou, Q.; Sun, Q.; Chen, L.; Nie, B.; Li, Q. A Comparative Analysis of LiDAR SLAM-Based Indoor Navigation for Autonomous Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 6907–6921. [CrossRef]
4. Arents, J.; Greitans, M. Smart Industrial Robot Control Trends, Challenges and Opportunities within Manufacturing. *Appl. Sci.* **2022**, *12*, 937. [CrossRef]
5. Jefroy, N.; Azarian, M.; Yu, H. Moving from Industry 4.0 to Industry 5.0: What Are the Implications for Smart Logistics? *Logistics* **2022**, *6*, 26. [CrossRef]
6. Vorasawad, K.; Park, M.; Kim, C. Efficient Navigation and Motion Control for Autonomous Forklifts in Smart Warehouses: LSPB Trajectory Planning and MPC Implementation. *Machines* **2023**, *11*, 1050. [CrossRef]
7. Huang, J.; Junginger, S.; Liu, H.; Thurow, K. Indoor Positioning Systems of Mobile Robots: A Review. *Robotics* **2023**, *12*, 47. [CrossRef]
8. Steenbeek, A.; Nex, F. CNN-Based Dense Monocular Visual SLAM for Real-Time UAV Exploration in Emergency Conditions. *Drones* **2022**, *6*, 79. [CrossRef]
9. Meng, J.; Wang, S.; Jiang, L.; Hu, Z.; Xie, Y. Accurate and Efficient Self-Localization of AGV Relying on Trusted Area Information in Dynamic Industrial Scene. *IEEE Trans. Veh. Technol.* **2023**, *72*, 7148–7159. [CrossRef]
10. Bovcon, B.; Muhovič, J.; Vranac, D.; Mozetič, D.; Perš, J.; Kristan, M. MODS-A USV-Oriented Object Detection and Obstacle Segmentation Benchmark. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 13403–13418. [CrossRef]
11. Halwani, M.; Ayyad, A.; AbuAssi, L.; Abdulrahman, Y.; Almaskari, F.; Hassanin, H.; Zweiri, Y. A Novel Vision-Based Multi-Functional Sensor for Normality and Position Measurements in Precise Robotic Manufacturing. *Precis. Eng.* **2024**, *88*, 367–381. [CrossRef]
12. Girbés-Juan, V.; Armesto, L.; Hernández-Ferrándiz, D.; Dols, J.F.; Sala, A. Asynchronous Sensor Fusion of GPS, IMU and CAN-Based Odometry for Heavy-Duty Vehicles. *IEEE Trans. Veh. Technol.* **2021**, *70*, 8617–8626. [CrossRef]
13. Xiao, Y.; Yan, Y.; Yu, Y.; Wang, B.; Liang, Y. Research on pose adaptive correction method of indoor rail mounted inspection robot in GIS Substation. *Energy Rep.* **2022**, *8*, 696–705. [CrossRef]
14. Morales-Díaz, A.B.; Gómez-Casas, J.; Treesatayapun, C.; Muñiz-Valdez, C.R.; Galindo-Valdés, J.S.; Martínez-Villafañe, J.F. Data-Driven Adaptive Modelling and Control for a Class of Discrete-Time Robotic Systems Based on a Generalized Jacobian Matrix Initialization. *Mathematics* **2023**, *11*, 2555. [CrossRef]
15. Cheng, J.; Bi, S.; Yuan, C.; Chen, L.; Cai, Y.; Yao, Y. A Graph Theory-Based Method for Dynamic Modeling and Parameter Identification of 6-DOF Industrial Robots. *Appl. Sci.* **2021**, *11*, 10988. [CrossRef]
16. Mancilla, A.; García-Valdez, M.; Castillo, O.; Merelo-Guervós, J.J. Optimal Fuzzy Controller Design for Autonomous Robot Path Tracking Using Population-Based Metaheuristics. *Symmetry* **2022**, *14*, 202. [CrossRef]
17. Guo, Z.; Ren, H.; Li, H.; Zhou, Q. Adaptive-Critic-Based Event-Triggered Intelligent Cooperative Control for a Class of Second-Order Constrained Multiagent Systems. *IEEE Trans. Artif. Intell.* **2023**, *4*, 1654–1665. [CrossRef]
18. Cardona, G.A.; Calderon, J.M. Robot Swarm Navigation and Victim Detection Using Rendezvous Consensus in Search and Rescue Operations. *Appl. Sci.* **2019**, *9*, 1702. [CrossRef]

19. Singh, Y.; Bibuli, M.; Zereik, E.; Sharma, S.; Khan, A.; Sutton, R. A Novel Double Layered Hybrid Multi-Robot Framework for Guidance and Navigation of Unmanned Surface Vehicles in a Practical Maritime Environment. *J. Mar. Sci. Eng.* **2020**, *8*, 624. [CrossRef]
20. Yan, Y.; Zhang, B.; Zhou, J.; Zhang, Y.; Liu, X. Real-Time Localization and Mapping Utilizing Multi-Sensor Fusion and Visual–IMU–Wheel Odometry for Agricultural Robots in Unstructured, Dynamic and GPS-Denied Greenhouse Environments. *Agronomy* **2022**, *12*, 1740. [CrossRef]

sensors

MDPI

Article

An Adaptive Control Scheme Based on Non-Interference Nonlinearity Approximation for a Class of Nonlinear Cascaded Systems and Its Application to Flexible Joint Manipulators

Zhangxing Liu, Hongzhe Jin * and Jie Zhao

School of Mechatronics Engineering, Harbin Institute of Technology, Harbin 150001, China; liuzhangxinghit@163.com (Z.L.); jzhao@hit.edu.cn (J.Z.)
* Correspondence: hongzhejin@hit.edu.cn

Abstract: Control design for the nonlinear cascaded system is challenging due to its complicated system dynamics and system uncertainty, both of which can be considered some kind of system nonlinearity. In this paper, we propose a novel nonlinearity approximation scheme with a simplified structure, where the system nonlinearity is approximated by a steady component and an alternating component using only local tracking errors. The nonlinearity of each subsystem is estimated independently. On this basis, a model-free adaptive control for a class of nonlinear cascaded systems is proposed. A squared-error correction procedure is introduced to regulate the weight coefficients of the approximation components, which makes the whole adaptive system stable even with the unmodeled uncertainties. The effectiveness of the proposed controller is validated on a flexible joint system through numerical simulations and experiments. Simulation and experimental results show that the proposed controller can achieve better control performance than the radial basis function network control. Due to its simplicity and robustness, this method is suitable for engineering applications.

Keywords: model-free adaptive control; nonlinearity approximation; cascaded system; flexible joint

Citation: Liu, Z.; Jin, H.; Zhao, J. An Adaptive Control Scheme Based on Non-Interference Nonlinearity Approximation for a Class of Nonlinear Cascaded Systems and Its Application to Flexible Joint Manipulators. *Sensors* **2024**, *24*, 3178. https://doi.org/10.3390/s24103178

Academic Editors: Yuanlong Xie, Shiqi Zheng, Zhaozheng Hu and Shuting Wang

Received: 4 February 2024
Revised: 12 May 2024
Accepted: 13 May 2024
Published: 16 May 2024

1. Introduction

The control problem of nonlinear cascaded systems commonly exists in engineering. Mechanical joints in robot manipulators are driven by motor currents [1–3]. The path tracking control of mobile robots is realized by adjusting wheel velocities [4–7]. Gyroscopic precession can be integrated into one-wheeled robots for steering control [8]. Flight dynamics in unmanned aerial vehicles (UAVs) can be stabilized through attitude adjustment [9–12]. Although these systems vary in physical assumptions, all of them can be modeled as nonlinear systems with a cascaded structure. The control design for such systems is challenging due to complicated system nonlinearity and uncertainty.

Disturbance rejection is a common approach to addressing the effects of unknown system nonlinearity and uncertainty. References [13,14] apply H-infinity optimal control for the linear system to suppress the effects of unknown disturbances. However, for systems with strong uncertainties, linear H-infinity control may lead to conservative performance. Hence, some researchers develop H-infinity controllers based on nonlinear system models [15–17]. Compared to the linear version, nonlinear H-infinity control allows for greater system nonlinearity under fine-tuning conditions and can delay control degradation and instability risks [17]. However, solving for nonlinear H-infinity controllers is usually complex and time-consuming [15,17,18]. In addition, invariant ellipsoid techniques are also introduced to optimize the robustness of control systems to unknown disturbances [19]. The invariant ellipsoid method simplifies the optimal controller to finding the smallest invariant ellipsoid of the closed-loop dynamic system [20]. A typical way is to apply the invariant ellipsoid method to suppress persistent disturbances through state-feedback control via LMI techniques [21–23]. It needs to quantitatively evaluate the effects of disturbances

on the system output; thus, accurate system information is required. Other methods, such as the generalized fractional equation [24], are also introduced to model complex, uncertain systems.

Obtaining optimal control solutions for nonlinear systems with complex uncertainties is often challenging. Hence, researchers have proposed to combine the aforementioned disturbance rejection methods with nonlinearity estimation approaches that are free from system models, such as artificial neural networks (ANN) [25,26], fuzzy networks [27], and disturbance observers [18,28,29].

Artificial intelligence networks, such as fuzzy systems and neural networks, are commonly used for nonlinearity approximation [30–37]. In [31], a fuzzy approximation-based adaptive backstepping controller was developed to assist in the movement of an upper-limb exoskeleton robot. References [33–35] present observer-based fuzzy neural-network output feedback control algorithms for underactuated nonlinear systems. These studies combine the adaptive backstepping technique with artificially intelligent networks to achieve a high-performance approximation-based controller. Reference [38] proposes a reinforcement learning-based method to ensure asymptotic tracking control of continuous-time systems. However, the application of these approaches is hindered by complex control objects with a high degree of freedom (DOF), structural uncertainty, and system nonlinearity [36]. For artificial intelligence networks with complex topological structures, the learning process degrades the transient performance of the system and requires high calculation efficiency. For real-time control systems, their high computational cost is an inevitable challenge. References [32,36,37] stated that these factors impede the development of intelligence networks-based adaptive control, especially in real-time control applications.

High-gain disturbance observer (HGDOB) and sliding mode control (SMC) are also effective methods to deal with systems with parametric uncertainties and unmodeled nonlinearities. In [39], a HGDOB is designed to estimate the system disturbance caused by friction, load force, and the parameter disturbance for electro-hydraulic systems. However, the high gain observer is sensitive to measurement noise and delayed outputs [40]. To solve this problem, Reference [41] designed time-varying gains relying on the generalization of the Halanay-type inequalities. Reference [42] tried to lower the observer gain by introducing artificial delays and Taylor's series. Similarly, the SMC is limited by chattering and peak phenomena in control signals [43]. In [44], a radial basis function neural network (RBFNN)-based soft computing strategy is applied to avoid the high switching gain that leads to chattering amplification. In [45], an adaptive sliding mode control method (ASMC) for robot manipulators is introduced. It utilizes the Taylor expansion to achieve a less conservative sign-function gain that enables chattering attenuation. The above approaches reduce chattering by applying extra-complicated policies. An interesting work is presented in [46] that presents a finite-time SMC (FT-SMC) and suppresses the peak phenomenon and chattering with an asymptotically convergent differentiator.

As can be seen from the previous discussion, in order to deal with unknown disturbances while avoiding problems caused by high control gains, controllers tend to become more and more complex and bloated. It is particularly unfriendly for engineering applications. Therefore, a simplified controller that is robust to unknown system nonlinearities and possesses mild control input is valuable for engineering applications.

Hence, this study aims to provide a simplified adaptive controller for a class of nonlinear cascaded systems. We first propose a so-called non-interference nonlinearity approximation (NINA) technique. It is based on the following system theory: For stable closed-loop systems, a bounded and continuous system nonlinearity can always be decomposed into steady and alternating components [47]. Furthermore, the output errors incorporated information relating to the system nonlinearity. Therefore, the unknown system nonlinearity can be modeled as a hierarchical form of a steady component and an alternating component. In addition, each nonlinearity can be approximated independently, using only local tracking errors. Thus, the proposed scheme is called non-interference nonlinearity approximation. Due to the simplified and decoupled approximation structure, the

computational complexity of NINA is significantly reduced. Based on NINA, a model-free adaptive control is proposed. It is convenient for engineering applications because it avoids the fussy process of system modeling and parameter identification. In addition, it is also robust to external disturbance and parameter perturbation due to accurate nonlinearity approximation and compensation, which are verified by numerical simulations and experiments. Finally, its control inputs are milder than those of SMC and HGDOB-based control.

In summary, the contributions of this work are as follows:

(1) A novel NINA scheme that has a simplified hierarchical structure is proposed. Based on only local tracking errors, the NINA technique can approximate the unknown system nonlinearity regardless of its internal complexity. Saturation functions with adjustable shaping factors help balance fast convergence against measurement noise, thereby providing a mild control input.
(2) A model-free adaptive control based on the NINA technique is proposed. Its uniformly ultimate boundedness (UUB) is proven by the Lyapnuov theory. The effectiveness and robustness have been validated by simulations and experiments on a flexible-joint manipulator system.
(3) Compared with the intelligence network-based control, the proposed method possesses a simplified structure and requires less computational costs. Compared with the SMC, the proposed method can perform fast trajectory tracking with mild control inputs. Hence, it is convenient for engineering applications.

Reference [48] introduces an adaptive weighted saturation function to suppress system uncertainty in a stabilization problem. The approach was applied to flexible manipulator control by [49,50]. Different from previous work, this paper approximates the nonlinearity of the closed-loop system using trajectory tracking errors instead of relying on system states. Furthermore, a hierarchical approximation structure is introduced in this paper. The steady component aims to achieve fast tracking for the major part of the nonlinearity, while the alternating component is designed to supplementarily track its high-frequency fluctuations. In addition, this paper conducted an elaborate theoretical analysis that not only proves the effectiveness of the proposed approximation method but also provided the upper bound of the approximation error. The convergence of the weighted parameters was also analyzed. Hence, this work can be viewed as an extension of the approach in [48] to some degree.

The remainder of this paper is organized as follows: Section 2 formulates the dynamic model for a class of nonlinear cascaded systems. Section 3 presents a decoupled control framework. Section 4 describes the NINA technique. On this basis, Section 5 proposes NINA-based adaptive control. Numerical simulations and experiments on the flexible joint system are presented in Sections 6 and 7, respectively. Conclusions are provided in Section 8.

2. Preliminaries

Mathematical Description of the Generalized Dynamics

First, we consider a class of nonlinear cascaded systems with n-DOF whose dynamics are given by:

$$\begin{bmatrix} m_\alpha(\alpha,\beta) & m_{\alpha\beta}(\alpha,\beta) \\ m_{\beta\alpha}(\alpha,\beta) & m_\beta(\alpha,\beta) \end{bmatrix} \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} + \begin{bmatrix} n_\alpha\left(\alpha,\dot{\alpha},\beta,\dot{\beta}\right) \\ n_\beta\left(\alpha,\dot{\alpha},\beta,\dot{\beta}\right) \end{bmatrix} = \begin{bmatrix} 0 \\ \tau_\beta \end{bmatrix} \tag{1}$$

where $\alpha, \beta \in \mathbb{R}^n$ represent the coordinates. $m_\alpha, m_{\alpha\beta}, m_{\beta\alpha}, m_\beta \in \mathbb{R}^{n\times n}$ form the system inertia matrix. $n_\alpha, n_\beta \in \mathbb{R}^n$, represent the system nonlinearity that captures centrifugal and Coriolis forces, viscous and frictions, gravitation, unmodeled system dynamics, and external disturbances. $\tau_\beta \in \mathbb{R}^n$ represents the control inputs. The first and second rows in

(1) represent the unactuated and actuated subsystems, respectively. For the convenience of distinguishing, they are denoted as the α-system and β-system.

For cascaded systems [1–12], the nonlinearity of the α-system n_α usually contains a dynamic coupling term that coordinates the behavior of the actuated and unactuated subsystems. Hence, n_α can be modeled as the combination of a known dynamic coupling term and a residual term, i.e.,

$$n_\alpha\left(\alpha, \dot{\alpha}, \beta, \dot{\beta}\right) = \delta_\alpha\left(\alpha, \dot{\alpha}, \beta, \dot{\beta}\right) - u_\alpha\left(\alpha, \dot{\alpha}, \beta, \dot{\beta}\right) \tag{2}$$

where u_α is the known dynamic coupling term and δ_α is the unmolded system nonlinearity.

Substituting (2) into (1), the dynamic model can be represented as

$$\begin{bmatrix} m_\alpha(\alpha,\beta) & m_{\alpha\beta}(\alpha,\beta) \\ m_{\beta\alpha}(\alpha,\beta) & m_\beta(\alpha,\beta) \end{bmatrix} \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} + \begin{bmatrix} \delta_\alpha\left(\alpha, \dot{\alpha}, \beta, \dot{\beta}\right) \\ n_\beta\left(\alpha, \dot{\alpha}, \beta, \dot{\beta}\right) \end{bmatrix} = \begin{bmatrix} u_\alpha\left(\alpha, \dot{\alpha}, \beta, \dot{\beta}\right) \\ \tau_\beta \end{bmatrix} \tag{3}$$

where the behavior of the α-system is indirectly regulated by the dynamic coupling term u_α. Given the states of the α-system, the value of u_α depends on the states of the β-system. Therefore, the control objective is to perform trajectory tracking control of the α-system by regulating the behavior of the β system.

Assumption 1. *Let* $x = \left(\alpha, \dot{\alpha}\right)$, $y = \left(\beta, \dot{\beta}\right)$. *For any given* $y_1, y_2 \in \mathbb{R}^n$, u_α *satisfies the following Lipschitz condition:*

$$\|u_\alpha(x, y_2) - u_\alpha(x, y_1)\| \leq \gamma \|y_2 - y_1\|, \tag{4}$$

where $\gamma > 0$ *is a finite constant.*

Assumption 2. *Let* S_1, S_2, *and* S_3 *be the ranges of* x, y, *and* u_α, *respectively. We have* $u_\alpha : (S_1 \times S_2) \to S_3$. *Given* $x \in S_1$, *for any desired* $u_\alpha \in S_3$, *there exists* $y_r \in S_2$ *satisfying the following inverse mapping:*

$$u_\alpha^{-1} : (u_{\alpha r} \times x) \to y_r, \tag{5}$$

where $u_{\alpha r}$ *is the desired value of* u_α, *and* y_r *is the desired value of* y. *This assumption is summarized from real systems* [1–12].

Remark 1. *Assumptions 1 and 2 guarantee the maneuverability of the α-system. If we take* $u_\alpha \in S_3$ *as the virtual control and using (4) and (5), the error between* $u_{\alpha r}$ *and* u_α *is bounded by*

$$\|u_\alpha(x, y_r) - u_\alpha(x, y)\| \leq \gamma \|y - y_r\|. \tag{6}$$

We have $u_\alpha(x, y) \to u_\alpha(x,\ y_r)$ *as* $y \to y_r$. *It indicates that the α-system can be indirectly regulated by the β-system via the dynamic coupling term* u_α.

Remark 2. *Equations (1) and (2) with assumptions 1 and 2 represent a class of nonlinear cascaded systems where the unactuated subsystems are indirectly regulated by the behaviors of the actuated subsystems through dynamic coupling. Some examples are provided as follows: For the flexible joint manipulator, the flexibility torque connects the dynamic behavior of the load and motor sides [3]. The gyro moment is used to maintain the lateral balance of the gyroscopic pendulum robot [7]. Dynamic coupling between attitude regulation torque and thrust force is widely utilized for the path tracking control of UAVs [9–12]. In the above examples, flexibility torque, the gyro moment, and aerodynamics can be viewed as the known dynamic coupling terms that can be used for controller design.*

3. Decoupled Control Framework

Considering system (1), there are two types of dynamic coupling: first, the dynamic coupling between the actuated and unactuated subsystems; and second, the dynamic

coupling between different degrees of freedom (DOFs). To address the problem mentioned above, a decoupled control framework is proposed in this paper, as shown in Figure 1. To deal with the dynamic coupling between the actuated and unactuated subsystems, we introduce a cascaded control framework where an α-controller is placed in the outer layer to stabilize the unactuated subsystem and a β-controller is positioned in the inner layer to regulate the actuated subsystem. The two sub-controllers are linked through the inverse mapping of the dynamic coupling term $\sqcap_\alpha$. In addition, the dynamic coupling between different DOFs is considered to be an unknown disturbance and is compensated by the proposed NINA technique presented in the next section.

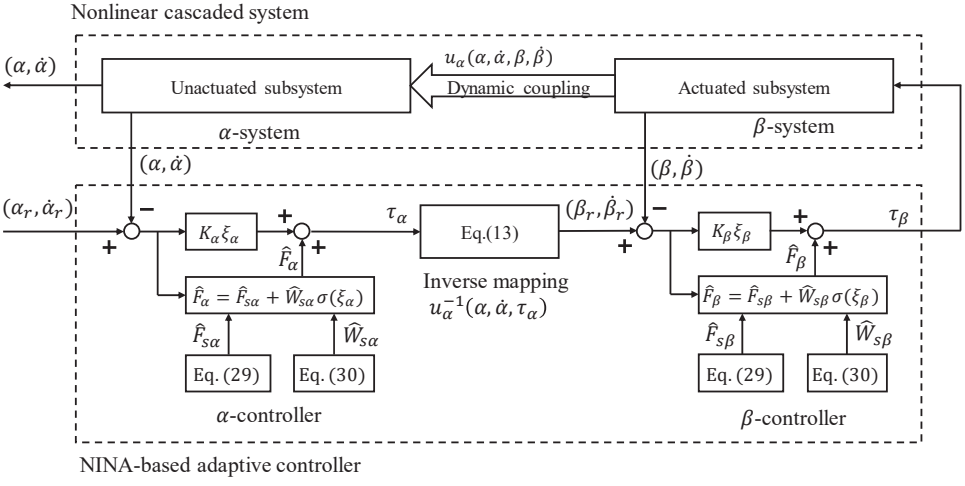


Figure 1. Decoupled cascaded control framework.

The control framework is derived below. Let $\alpha_r(t)$ and $\beta_r(t)$ be the reference trajectory of the α- and β-systems, which are assumed to be bounded and to have finite first- and second-order time derivatives. Let $e_\alpha = \alpha_r - \alpha$ and $e_\beta = \beta_r - \beta$ be the position tracking errors. Then, the following synthetic tracking errors are introduced:

$$\begin{matrix} \xi_\alpha = \Lambda_\alpha e_\alpha + \dot{e}_\alpha \\ \xi_\beta = \Lambda_\beta e_\beta + \dot{e}_\beta \end{matrix}. \tag{7}$$

where $\Lambda_\alpha, \Lambda_\beta > 0$ are diagonal positive gain matrices. Substituting (7) into model (3) and applying $\tau_\alpha = u_{\alpha r}$ as the virtual control, the error dynamics can be expressed as

$$\begin{bmatrix} m_\alpha(\alpha,\beta) & m_{\alpha\beta}(\alpha,\beta) \\ m_{\beta\alpha}(\alpha,\beta) & m_\beta(\alpha,\beta) \end{bmatrix} \begin{bmatrix} \dot{\xi}_\alpha \\ \dot{\xi}_\beta \end{bmatrix} = \begin{bmatrix} m_\alpha(\alpha,\beta) & m_{\alpha\beta}(\alpha,\beta) \\ m_{\beta\alpha}(\alpha,\beta) & m_\beta(\alpha,\beta) \end{bmatrix} \begin{bmatrix} \Lambda_\alpha \dot{e}_\alpha + \ddot{\alpha}_r \\ \Lambda_\beta \dot{e}_\beta + \ddot{\beta}_r \end{bmatrix} + \begin{bmatrix} v_\alpha - \tau_\alpha \\ n_\beta - \tau_\beta \end{bmatrix}, \tag{8}$$

where $v_\alpha = \delta_\alpha + \widetilde{u}_\alpha$ represents a lumped nonlinearity. $\widetilde{u}_\alpha = u_{\alpha r} - u_\alpha$ is the distortion between the desired control input and its actual value. Such a distortion is mainly caused by state tracking errors, parameter perturbations, and the model uncertainty of u_α.

Let us analyze (8) by choosing the following Lyapunov function:

$$V_1 = \frac{1}{2}\xi^T M \xi, \tag{9}$$

with

$$\xi = \begin{bmatrix} \xi_\alpha \\ \xi_\beta \end{bmatrix}, M = \begin{bmatrix} m_\alpha(\alpha,\beta) & m_{\alpha\beta}(\alpha,\beta) \\ m_{\beta\alpha}(\alpha,\beta) & m_\beta(\alpha,\beta) \end{bmatrix}.$$

Considering the time derivative of (9) and substituting (8), we obtain

$$\dot{V}_1 = \xi^T(F - \tau), \tag{10}$$

with

$$\tau = \begin{bmatrix} \tau_\alpha \\ \tau_\beta \end{bmatrix},$$

$$F = \begin{bmatrix} F_\alpha \\ F_\beta \end{bmatrix} = \frac{1}{2}\begin{bmatrix} \dot{m}_\alpha(\alpha,\beta) & \dot{m}_{\alpha\beta}(\alpha,\beta) \\ \dot{m}_{\beta\alpha}(\alpha,\beta) & \dot{m}_\beta(\alpha,\beta) \end{bmatrix}\begin{bmatrix} \xi_\alpha \\ \xi_\beta \end{bmatrix} + \begin{bmatrix} m_\alpha(\alpha,\beta) & m_{\alpha\beta}(\alpha,\beta) \\ m_{\beta\alpha}(\alpha,\beta) & m_\beta(\alpha,\beta) \end{bmatrix}\begin{bmatrix} \Lambda_\alpha \dot{e}_\alpha + \ddot{\alpha}_r \\ \Lambda_\beta \dot{e}_\beta + \ddot{\beta}_r \end{bmatrix} + \begin{bmatrix} v_\alpha \\ n_\beta \end{bmatrix} \tag{11}$$

where F is an integrated system nonlinearity. Considering F as an unknown disturbance and compensating via nonlinearity estimation, a simplified control law can then be designed as

$$\tau = K\xi + \hat{F}, \tag{12}$$

where K is a positive, definite diagonal gain matrix. $\hat{F}$ is the estimation of F applied for nonlinearity compensation.

Substituting (12) into (10), $\dot{V}_1$ becomes

$$\dot{V}_1 = -\xi^T K\xi + \xi^T(F - \hat{F}). \tag{13}$$

Ideally, if $\hat{F} = F$, ξ is asymptotically convergent to zero. If $\widetilde{F} = F - \hat{F}$ is bounded, ξ will be ultimately bounded. It can be seen that the stability of the closed-loop system is determined by the nonlinearity approximation process. In the next section, a simplified NINA technique is proposed for the nonlinearity approximation.

Remark 3. *As shown in* Figure 1*, the reference of the α-system* $(\alpha_r, \dot{\alpha}_r)$ *is given by users, while the reference of the β-system* $\left(\beta_r, \dot{\beta}_r\right)$ *is generated to guide the tracking of the virtual control* $u_\alpha \to u_{\alpha r} = \tau_\alpha$.

Given the states of the α-system $(\alpha, \dot{\alpha})$ *and the desired value of virtual control,* $u_{\alpha r}$*, we have*

$$u_\alpha\left(\alpha, \dot{\alpha}, \beta, \dot{\beta}\right) \to u_{\alpha r}\left(\alpha, \dot{\alpha}, \beta_r, \dot{\beta}_r\right) \text{ as } \left(\beta, \dot{\beta}\right) \to \left(\beta_r, \dot{\beta}_r\right). \tag{14}$$

Hence, $\left(\beta_r, \dot{\beta}_r\right)$ *can be obtained by solving the inverse mapping of* $u_\alpha\left(\alpha, \dot{\alpha}, \beta_r, \dot{\beta}_r\right)$ *with respect to* $\left(\beta_r, \dot{\beta}_r\right)$*, i.e.,*

$$\left(\beta_r, \dot{\beta}_r\right) = u_\alpha^{-1}(\alpha, \dot{\alpha}, u_{\alpha r}). \tag{15}$$

An example of such inverse mapping about the elastic torque of the flexible joint manipulator is given in Equation (51).

4. Principles of NINA

In this section, a simplified nonlinearity approximation scheme is presented. The nonlinearity of each subsystem can be estimated independently by simply utilizing the local tracking error.

Declaration 1. *Considering the nonlinearity approximation by each subsystem, we adopt the following symbolic notation: for a vector V or a diagonal matrix V, the j-th element is marked by* V_j*, where j = 1, 2, . . ., 2n.*

Let F_j represents the system nonlinearity and is assumed to be a bounded continuous time function. In real-world applications, most of the plants are controlled by digital controllers. Therefore, F_j can be viewed as a piecewise time-varying function within successive control cycles. Mathematically, such a piecewise time-varying function can

always be expressed as the synthetic form of steady and alternating components [47]. Hence, the system nonlinearity can be modeled in the time-domain as

$$F_j(t) \equiv F_{sj} + \Delta_j(t), (t_I \le t \le t_I + \mu T_c) \tag{16}$$

where $\mu > 1$ is a positive integer. t_I is the initial moment, and T_c is the control cycle. F_{sj} and Δ_j denote a bounded steady component and a bounded alternating component, respectively.

In addition, for closed-loop control systems, the tracking error reflects the combined effect of system nonlinearities. Therefore, we introduce the following structure to approximate system nonlinearities with the synthetic tracking error ξ_j:

$$F_{Aj}(\xi_j) = F_{sj} + W_{sj}\sigma(\xi_j),\ (|\Delta_j| < W_{sj} < \infty), \tag{17}$$

where F_{Aj} is the approximation of F_j. F_{sj} is the steady component in (16), and $W_{sj}\sigma(\xi_j)$ is introduced to approximate the alternating component Δ_j, where W_{sj} is a dynamically adjusted weight coefficient and $\sigma(\xi_j)$ is a saturation function expressed as

$$\sigma(\xi_j) = \xi_j / (\vartheta_{sj} + |\xi_j|),\ (0 < \vartheta_{sj} < \infty), \tag{18}$$

where ϑ_{sj} is a shaping factor. When $\vartheta_{sj} \to 0$, $\sigma(\xi_j)$ acts as a signed switching function that is highly sensitive to the variation of ξ_j around zero. By contrast, when $\vartheta_{sj} \to \infty$, $\sigma(\xi_j)$ tends to zero and becomes insensitive to the changes of ξ_j. Compared with the linear or polynomial approximation, the introduced saturation function enables a wide range of sensitivity adjustment w.r.t. ξ_j via only one parameter. It is more convenient and adaptable.

From (16) and (17), the approximation error between $F_j(t)$ and F_{Aj} can be calculated as

$$E(\Delta_j, \xi_j) = F_j(t) - F_{Aj}(\xi_j) = \Delta_j(t) - W_{sj}\sigma(\xi_j). \tag{19}$$

Theorem 1. *For a bounded continuous nonlinearity $F_j(t)$, there exist optimized parameters F_{sj}, W_{sj}, and ϑ_{sj} for the approximation structure (17), (18) that satisfy the identity $E(\Delta_j, \xi_j) = 0$ and the synthetic tracking error is ultimately bounded by*

$$|\xi_j| \le \vartheta_{sj}(|\Delta_j / W_{sj}|) / [1 + |\Delta_j / W_{sj}|], \tag{20}$$

where $|\xi_j| \to 0$, as $\vartheta_{sj} \to 0$ and $W_{sj} > |\Delta_j|$. It illustrates that the proposed structure can effectively approximate the system nonlinearity while maintaining a small synthetic tracking error.

Proof . Let $\hat{F}_j(t) = F_A(\xi_j)$ and substitute (19) into (13). The first derivative of the Lyapunov function in (13) becomes

$$\dot{V}_1 = -\sum_j K_j \xi_j^2 + \sum_j E(\Delta_j, \xi_j)\xi_j, \tag{21}$$

where $-K_j\xi_j^2 \le 0$. The property of $E(\Delta_j, \xi_j)\xi_j$ is discussed below.

Step 1: The second derivatives of $E(\Delta_j, \xi_j)\xi_j$ with respect to ξ_j is presented as

$$\frac{\partial^2 E(\Delta_j, \xi_j)\xi_j}{\partial^2 \xi_j} = 2W_{sj}(\sigma(\xi_j) - 1)\sigma'(\xi_j), \tag{22}$$

where $\sigma'(\xi_j)$ is the first derivatives of $\sigma(\xi_j)$ with respect to ξ_j. It can be verified segmentally that $\frac{\partial^2 E(\Delta_j, \xi_j)\xi_j}{\partial^2 \xi_j} < 0$ for any $\xi_j \in R$ and $W_{sj} > 0$. Hence, $E(\Delta_j, \xi_j)\xi_j$ is an open downward convex function with respect to ξ_j. This can be further verified by the profile diagram shown in Figure 2.

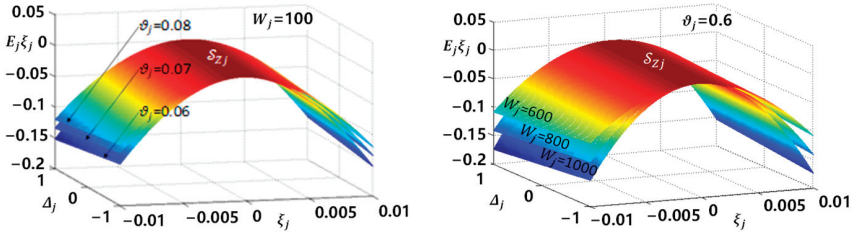


Figure 2. Changes in the profile of $E_j\left(\Delta_j, \xi_j\right)\xi_j$ caused by parameter drifts.

Step 2: Given that $E(\Delta_j, \xi_j)\xi_j$ is an open downward convex function with respect to ξ_j, its sign will vary around the nonzero solution $\xi_j = \xi_z(\Delta_j)$ of equation $E(\Delta_j, \xi_j)\xi_j = 0$. With (19), the value of $\xi_z(\Delta_j)$ can be calculated as

$$\xi_z(\Delta_j) = \begin{cases} \xi_z^+ = \vartheta_{sj}(\Delta_j/W_{sj})/\left[1 - (\Delta_j/W_{sj})\right], (\Delta_j \geq 0) \\ \xi_z^- = \vartheta_{sj}(\Delta_j/W_{sj})/\left[1 + (\Delta_j/W_{sj})\right], (\Delta_j \leq 0) \end{cases}, \tag{23}$$

Step 3: The following two compact sets are defined accordingly:

$$\mathcal{S}_{zj}^+ \triangleq \left\{ (\xi_j, \Delta_j) \in \mathbb{R}^2 | 0 < \xi_j < \xi_z^+, \Delta_j > 0 \right\},$$

$$\mathcal{S}_{zj}^- \triangleq \left\{ (\xi_j, \Delta_j) \in \mathbb{R}^2 | \xi_z^- < \xi_j < 0, \Delta_j < 0 \right\}.$$

Step 4: Substituting $E(\Delta_j, \xi_j)\xi_j$ around the domain $\mathcal{S}_{zj}^+ \cup \mathcal{S}_{zj}^-$, we can verify that, for $(\xi_j, \Delta_j) \notin \mathcal{S}_{zj}^+ \cup \mathcal{S}_{zj}^-$, there is

$$E(\Delta_j, \xi_j)\xi_j < 0, \tag{24}$$

and for $(\xi_j, \Delta_j) \in \mathcal{S}_{zj}^+ \cup \mathcal{S}_{zj}^-$, there is

$$E(\Delta_j, \xi_j)\xi_j > 0. \tag{25}$$

Using (21) and considering the extreme case when $K_j \to 0$, if $(\xi_j, \Delta_j) \in \mathcal{S}_{zj}^+ \cup \mathcal{S}_{zj}^-$, $\dot{V}_1$ tends to be positive and ξ_j diverges from $\mathcal{S}_{zj}^+ \cup \mathcal{S}_{zj}^-$. By contrast, if $(\xi_j, \Delta_j) \notin \mathcal{S}_{zj}^+ \cup \mathcal{S}_{zj}^-$, $\dot{V}_1 < 0$ and ξ_j converges back to $\mathcal{S}_{zj}^+ \cup \mathcal{S}_{zj}^-$. This variation proves that ξ_j is ultimately restricted within $\mathcal{S}_{zj}^+ \cup \mathcal{S}_{zj}^-$, which provides

$$|\xi_j| \leq \vartheta_{sj}|\Delta_j/W_{sj}|/\left[1 + |\Delta_j/W_{sj}|\right]. \tag{26}$$

According to (23) and (26), if the candidates are chosen as $\vartheta_{sj} \ll 1$ and $W_{sj} \gg |\Delta_j|$, there exists $\xi_z(\Delta_j)$ that tends to zero and satisfies the identity: $E(\Delta_j, \xi_z(\Delta_j)) \equiv 0$. It illustrates that $F_{Aj}(\xi_j)$ in (17) can effectively approximate the system nonlinearity $F_j(t)$ around the domain $\mathcal{S}_{zj}^+ \cup \mathcal{S}_{zj}^-$, while maintaining a small synthetic tracking error. □

5. Adaptive Control Based on NINA

In this section, an adaptive control utilizing NINA is fulfilled, and the stability analysis is carried out.

5.1. Adaptive Law

Given that F_j and W_{sj} are optimal candidates for the approximation structure in (17), the integrated error $\iint E^2(\Delta_j, \xi_j)\mathrm{d}\Delta_j\mathrm{d}\xi_j$ is minimized. The estimation of $F_j(\mathrm{t})$ is defined as

$$\hat{F}_j(t) = \hat{F}_{Aj}(\xi_j) = \hat{F}_{sj} + \hat{W}_{sj}\sigma(\xi_j). \tag{27}$$

where $\hat{F}_{sj}$ and $\hat{W}_{sj}$ are the estimations of F_j and W_{sj}, respectively. Subtracting (27) from (16), the error between $F_j(t)$ and $\hat{F}_j(t)$ is represented as

$$\widetilde{F}_j(t) = \widetilde{F}_{sj} + \widetilde{W}_{sj}\sigma(\xi_j) + E(\Delta_j, \xi_j), \tag{28}$$

with $\widetilde{F}_{sj} = F_{sj} - \hat{F}_{sj}$ and $\widetilde{W}_{sj} = W_{sj} - \hat{W}_{sj}$. The adaptive law of $\hat{F}_{sj}$ and $\hat{W}_{sj}$ is given by the following squared-error correction procedures:

$$\dot{\hat{W}}_{sj} = -A_j\xi_j^2\hat{W}_{sj} + B_j\sigma(\xi_j)\xi_j \tag{29}$$

$$\dot{\hat{F}}_{sj} = -A_j\xi_j^2\hat{F}_{sj} + B_j\xi_j \tag{30}$$

where $A_j, B_j > 0$ are the adaptive gains. The term $A_j\xi_j^2$ plays a role in preventing the divergences of $\hat{W}_{sj}$ and $\hat{F}_{sj}$.

Using (29) and (30), the transient performance of $\widetilde{F}_{sj}$ and $\widetilde{W}_{sj}$ is analyzed next. Applying $\dot{\widetilde{W}}_{sj} = -\dot{\hat{W}}_{sj}$ and $\dot{\widetilde{F}}_{sj} = -\dot{\hat{F}}_{sj}$, (29) and (30) can then be represented as

$$\dot{\widetilde{W}}_{sj} = -A_j\xi_j^2\widetilde{W}_{sj} + \rho(\xi_j), \tag{31}$$

$$\dot{\widetilde{F}}_{sj} = -A_j\xi_j^2\widetilde{F}_{sj} + \varrho(\xi_j), \tag{32}$$

where

$$\rho = \left(A_j\xi_j W_{sj} - B_j\sigma(\xi_j)\right)\xi_j, \varrho = \left(A_j\xi_j F_{sj} - B_j\right)\xi_j.$$

The solutions of (31) and (32) are represented as

$$\widetilde{W}_{sj}(t) = \phi(t, t_I)\widetilde{W}_{sj}(t_I) + \int_{t_I}^{t}\phi(t,\tau)\rho(\tau)\mathrm{d}\tau, \tag{33}$$

$$\widetilde{F}_{sj}(t) = \phi(t, t_I)\widetilde{F}_{sj}(t_I) + \int_{t_I}^{t}\phi(t,\tau)\varrho(\tau)\mathrm{d}\tau, \tag{34}$$

where

$$\phi(t, t_I) = \exp\left(-A_j\int_{t_I}^{t}\xi_j^2(\tau)\mathrm{d}\tau\right). \tag{35}$$

Supposing that the persistent excitation condition holds for ξ_j, $\phi(t, t_I)$ asymptotically converges to zero with the increase in t, proving that (31) and (32) are bounded-input-bounded-output stable. The expressions of ρ and ϱ show that ξ_j is a unique source that changes $\widetilde{W}_{sj}$ and $\widetilde{F}_{sj}$ in the steady state. Therefore, $\widetilde{W}_{sj}$ and $\widetilde{F}_{sj}$ can be restricted within a small area along with the convergence of ξ_j. We can also conclude from (35) that $\widetilde{W}_{sj}$ and $\widetilde{F}_{sj}$ obtain fast convergence as long as A_j is set to make the steady-state time of $\phi(t, t_I)$ much smaller than μT_c.

5.2. Adaptive Controller and Stability Analysis

Combining the control structure in (12), and approximating the system nonlinearity using (27), (29), and (30), the NINA-based adaptive control law is presented as

$$\begin{cases} \tau_j = K_j\xi_j + \hat{F}_{sj} + \hat{W}_{sj}\sigma(\xi_j) \\ \dot{\hat{W}}_{sj} = -A_j\xi_j^2\hat{W}_{sj} + B_j\sigma(\xi_j)\xi_j \\ \dot{\hat{F}}_{sj} = -A_j\xi_j^2\hat{F}_{sj} + B_j\xi_j \end{cases}. \tag{36}$$

Theorem 2. *Consider the nonlinear system represented by (3) and use the control law in (36). If control gainK_j is selected in accordance with the following inequality*

$$K_j > \frac{1}{4}\left[A_j\left(W_{sj}^2 + F_{sj}^2\right)/B_j\right], \tag{37}$$

the synthetic tracking $\xi_j$is uniformly and ultimately bounded by

$$|\xi_j|^2 \le W_{sj}\vartheta_{sj}/G_j$$

with

$$G_j = K_j - \frac{1}{4}\left[A_j\left(W_{sj}^2 + F_{sj}^2\right)/B_j\right]$$

Moreover, the synthetic tracking error $\xi_j$can be restricted to a small area around zero as $0 < (\vartheta_{sj}, A_j) \ll 1$ and $(K_j,\ B_j) \gg 1$.

Proof. A Lyapunov function candidate is selected as

$$V_2 = V_1 + \frac{1}{2B_j}\sum_j\left(\widetilde{W}_{sj}^2 + \widetilde{F}_{sj}^2\right), \tag{38}$$

Using (13) and differentiating (38) with respect to time, we have

$$\dot{V}_2 = \sum_j\left[\xi_j\widetilde{F}_j(t) + \left(\widetilde{W}_{sj}\dot{\widetilde{W}}_{sj} + \widetilde{F}_{sj}\dot{\widetilde{F}}_{sj}\right)/B_j - K_j\xi_j^2\right], \tag{39}$$

Applying $\dot{\widetilde{W}}_{sj} = -\dot{\hat{W}}_{sj}$ and $\dot{\widetilde{F}}_{sj} = -\dot{\hat{F}}_{sj}$ with (29) and (30), $\dot{V}_2$ becomes

$$\dot{V}_2 = \sum_j\left[\begin{array}{c} A_j\xi_j^2\left(\widetilde{W}_{sj}\hat{W}_{sj} + \widetilde{F}_{sj}\hat{F}_{sj}\right)/B_j \\ +E_j(\Delta_j,\xi_j)\xi_j - K_j\xi_j^2 \end{array}\right]. \tag{40}$$

where $\widetilde{W}_{sj}\hat{W}_{sj}$ and $\widetilde{F}_{sj}\hat{F}_{sj}$ are bounded by

$$\widetilde{W}_{sj}\hat{W}_{sj} \equiv \widetilde{W}_{sj}\left(W_{sj} - \widetilde{W}_{sj}\right) \le W_{sj}^2/4, \tag{41}$$

$$\widetilde{F}_{sj}\hat{F}_{sj} \equiv \widetilde{F}_{sj}\left(F_{sj} - \widetilde{F}_{sj}\right) \le F_{sj}^2/4. \tag{42}$$

Substituting (41) and (42) into (40), we can obtain the following inequality:

$$\dot{V}_2 \le -\sum_j G_j\xi_j^2 + \sum_j E(\Delta_j,\xi_j)\xi_j, \tag{43}$$

where $G_j = K_j - \frac{1}{4}\left[A_j\left(W_{sj}^2 + F_{sj}^2\right)/B_j\right]$.

According to the conclusion in (22) that $E(\Delta_j, \xi_j)\xi_j$ is an open-downward convex function with respect to ξ_j, $E(\Delta_j, \xi_j)\xi_j$ will reach its maximum value as

$$\frac{\partial E(\Delta_j, \xi_j)\xi_j}{\partial \xi_j} = 0. \tag{44}$$

Substituting the solution of (44), the maximum value of $E(\Delta_j, \xi_j)\xi_j$ can be calculated as

$$E(\Delta_j, \xi_j)\xi_j \le W_{sj}\sigma^2(\xi_j)\vartheta_{sj} = W_{sj}\vartheta_{sj}\xi_j^2/\left(\vartheta_{sj} + |\xi_j|\right)^2. \tag{45}$$

Applying (45), (43) becomes

$$\dot{V}_2 \le -\sum_j G_j\xi_j^2 + \sum_j \left[W_{sj}\vartheta_{sj}/\left(\vartheta_{sj} + |\xi_j|\right)^2\right]\xi_j^2. \tag{46}$$

If $|\xi_j|^2 > \frac{W_{sj}\vartheta_{sj}}{G_j}$, we have $\dot{V}_2 < 0$. Hence, the synthetic tracking error ξ_j is uniformly and ultimately bounded by

$$|\xi_j|^2 \le W_{sj}\vartheta_{sj}/G_j. \tag{47}$$

The synthetic tracking error ξ_j will be restricted into a small area around zero as $0 < (\vartheta_{sj}, A_j) \ll 1$ and $(K_j, B_j) \gg 1$. This completes the proof. □

6. Numerical Simulation

In this section, the proposed method is verified by the trajectory tacking control of a two-link flexible-joint manipulator. The simulations of the manipulator under step change, different link lengths, and joint stiffness are performed to evaluate the robustness of the proposed method. The simulations are conducted utilizing the fourth-order Runge–Kutta method.

The finite-time sliding mode control (FT-SMC) in [46] and the RBFN-based control in [51] are also simulated for comparison. These two controllers are selected as representatives of sliding mode control and RBFN-based control methods. They exhibit relatively simple yet representative architectures and are also model-free methods, which makes them suitable as benchmarks for comparison.

6.1. Simulation Setup

The configuration of the manipulator is depicted in Figure 3, and its parameters are listed in Table 1. Referring to [52], the system dynamics can be modeled as

$$\begin{bmatrix} m_{11}(\alpha_1, \alpha_2) & m_{12}(\alpha_1, \alpha_2) \\ m_{21}(\alpha_1, \alpha_2) & m_{22}(\alpha_1, \alpha_2) \end{bmatrix}\begin{bmatrix} \ddot{\alpha}_1 \\ \ddot{\alpha}_2 \end{bmatrix} + \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} + \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{48}$$

$$\begin{bmatrix} J_1(\beta_1) & 0 \\ 0 & J_2(\beta_2) \end{bmatrix}\begin{bmatrix} \ddot{\beta}_1 \\ \ddot{\beta}_2 \end{bmatrix} + \begin{bmatrix} f_{d1} \\ f_{d2} \end{bmatrix} - \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \tau_3 \\ \tau_4 \end{bmatrix} \tag{49}$$

with

$$\begin{cases} u_1 = k_{s1}(\alpha_1 - \beta_1) + k_{d1}\left(\dot{\alpha}_1 - \dot{\beta}_1\right) \\ u_2 = k_{s2}(\alpha_2 - \beta_2) + k_{d2}\left(\dot{\alpha}_2 - \dot{\beta}_2\right) \end{cases} \tag{50}$$

where $\alpha = (\alpha_1, \alpha_2)^{\mathrm{T}}$ and $\beta = (\beta_1, \beta_2)^{\mathrm{T}}$ are the position vectors of the load and motor sides, respectively; m_{11}, m_{12}, m_{21}, m_{22} are the elements of the load side inertial matrix, J_1 and J_2 are the elements of the motor side inertial matrix; h_1 and h_2 consist of Coriolis and centrifugal terms; G_1 and G_2 contain gravitational terms; u_1 and u_2 are the elastic

torque terms; f_{d1} and f_{d2} are the damping terms of the motor side; and τ_3 and τ_4 are the motor torques.

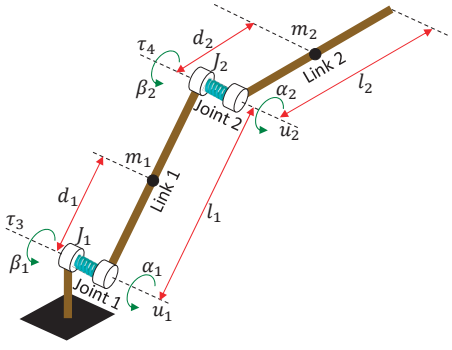


Figure 3. Architecture of the two-link robot manipulator with joint flexibility.

Table 1. Parameters of the two-link robot manipulator system.

Parameters	Value	Parameters	Value	Unit	Means
m_1	6.0	m_2	4.0	kg	mass of the link
l_1	0.6	l_2	0.4	m	length of the link
d_1	0.3	d_2	0.2	m	length of the mass center
J_1	2.2×10^{-3}	J_2	6.4×10^{-4}	kg.m^2	motor inertia
f_{d1}	0.088	f_{d2}	0.057	N.m/rad/s	motor damping
k_{s1}	120.0	k_{s2}	100.0	N.m/rad	joint stiffness
k_{d1}	6.79×10^{-2}	k_{d2}	3.69×10^{-2}	N.m/rad/s	elasticity damping

The control law presented in Theorem 2 is applied. The reference trajectory of the load-side is given by the user command. The reference trajectory of the motor side is generated by solving the following differential equation:

$$\begin{cases} \dot{\beta}_{r1} = -k_{s1}\beta_{r1}/k_{d1} + \left[(\tau_{\alpha 1} + k_{s1}\alpha_1)/k_{d1} + \dot{\alpha}_1\right] \\ \dot{\beta}_{r2} = -k_{s2}\beta_{r2}/k_{d2} + \left[(\tau_{\alpha 2} + k_{s2}\alpha_2)/k_{d2} + \dot{\alpha}_2\right] \end{cases}, \tag{51}$$

where $\left(\beta_{r1}, \beta_{r2}, \dot{\beta}_{r1}, \dot{\beta}_{r2}\right)$ represent the command trajectory of the motor side, and $\tau_{\alpha 1}$ and $\tau_{\alpha 2}$ are the desired values of u_1 and u_2, respectively. This formula is an inverse mapping of $u_1\left(\alpha_1, \dot{\alpha}_1, \beta_{1r}, \dot{\beta}_{1r}\right)$ and $u_2\left(\alpha_2, \dot{\alpha}_2, \beta_{2r}, \dot{\beta}_{2r}\right)$ in (50). The parameters of the proposed adaptive controller used in the simulation are listed in Table 2.

Table 2. Parameters of the proposed NINA-based adaptive control.

Parameters	Value	Parameters	Value
$\vartheta_{s1}, \vartheta_{s2}$	0.3	$\vartheta_{s3}, \vartheta_{s4}$	6
$K_1 \sim K_2$	10	$K_3 \sim K_4$	5
$A_1 \sim A_2$	0.3	$A_3 \sim A_4$	1
$B_1 \sim B_2$	1000	$B_3 \sim B_4$	1000
$\Lambda_1 \sim \Lambda_2$	5	$\Lambda_3 \sim \Lambda_4$	5

Note that subscripts 1 and 2 represent the motor-side control parameters of joints 1 and 2, respectively. Subscripts 3 and 4 represent the load-side control parameters of joints 1 and 2, respectively.

Remark 4. *For the flexible joint system mentioned above, the motor side is the actuated subsystem, and the link side is the unactuated subsystem. The elastic torque* (u_1, u_2) *helps coordinate the behavior of the motor and load sides. It can be verified that Assumptions 1 and 2 hold for* (u_1, u_2) *by examining the expression in (50).*

6.2. Trajectory Tracking Performance Validation

We compared the trajectory tracking performance of the three control methods: the proposed method, the RBFN-based method, and FT-SMC. The robot arm starts from the horizontal position and tracks sinusoidal trajectories as shown below:

$$\begin{cases} \alpha_r = \frac{\pi}{6}(1+\sin(2t)), \text{load side} \\ \beta_r = \frac{\pi}{6}(1+\cos(2t)), \text{motor side} \end{cases}.$$

To evaluate the tracking performance of the controllers, we introduce the following evaluation index, and the results are listed in Tables 3 and 4:

Table 3. Tracking performance of the three methods.

		SSP		CTP		
		Joint 1	**Joint 2**	**Joint 1**	**Joint 2**	**Average**
Link Side	NINA	−0.07°~0.08°	−0.03°~0.03°	1.75 s	1.78 s	1.77 s
	FT-SMC	−0.05°~0.09°	−0.07°~0.10°	1.88 s	1.24 s	1.56 s
	RBFN	−0.17°~0.16°	0.20°~0.20°	2.73 s	1.74 s	2.24 s
Motor Side	NINA	−0.11°~0.12°	−0.13°~0.13°	0.50 s	0.20 s	0.35 s
	FT-SMC	−0.21°~0.09°	−0.07°~0.10°	0.82 s	0.90 s	0.86 s
	RBFN	−0.29°~0.17°	−0.18°~0.19°	1.24 s	1.09 s	1.17 s

Table 4. Nonlinearity estimation performance of the three methods.

		SSE		CTE		
		Joint 1	**Joint 2**	**Joint 1**	**Joint 2**	**Average**
Link Side	NINA	−0.078~0.16 Nm	−0.12~0.11 Nm	0.7 s	0.4 s	0.55 s
	FT-SMC	−0.28~0.50 Nm	−0.19~019 Nm	1.9 s	1.7 s	1.8 s
	RBFN	−1.60~1.60 Nm	−0.8~0.8 Nm	2.76 s	1.9 s	2.33 s
Motor Side	NINA	−0.27~0.52 Nm	−0.064~0.006 Nm	0.24 s	0.19 s	0.22 s
	FT-SMC	−1.05~0.75 Nm	−0.14~0.14 Nm	0.9 s	1.3 s	1.10 s
	RBFN	−2.14~0.55 Nm	−0.66~064 Nm	1.2 s	1.4 s	1.30 s

(a) SSP (steady-state tracking error in position):

$$SSP = \left[\min e_j(t)\ \max e_j(t)\right],\ for\ t > t_{M1},\ j = \alpha\ or\ \beta$$

where $\min e_j(t)$ and $\max e_j(t)$ represent the lower and upper bounds of the position tracking error, respectively. t_{M1} represents the time since the tracking error varied periodically and steadily. In this simulation, it is set $t_{M1} = 4$ s.

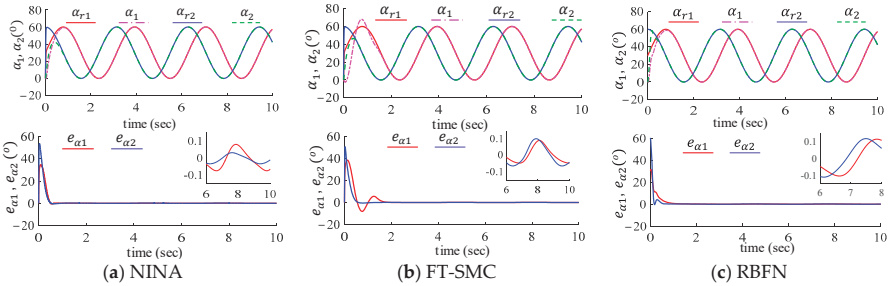
(b) CTP (convergence time of trajectory tracking): It is defined as the time when the tracking error is free from initial oscillation, shown in Figure 4c, and first comes into the range of steady-state, i.e., SSP.

(c) SSE (steady-state estimation error in system nonlinearity):

$$SSE = \left[\min\widetilde{F}_j(t)\ \max\widetilde{F}_j(t)\right], for\ t > t_{M2},\ j = \alpha\ or\ \beta$$

where $\min\widetilde{F}_j(t)$ and $\max\widetilde{F}_j(t)$ represent the low and up bounds of the nonlinearity estimation error, respectively. t_{M2} is defined similarly to t_{M1} and is set as $t_{M2} = 4$ s.

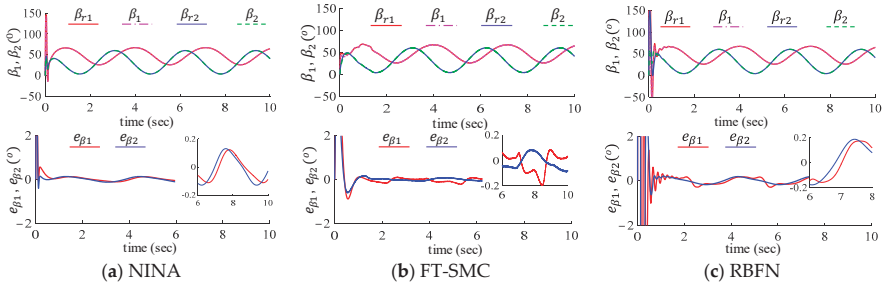
(d) CTE (convergence time of nonlinearity estimation): It is similar to the definition of CTP.



(**a**) NINA (**b**) FT-SMC (**c**) RBFN

Figure 4. Link-side tracking performance of (**a**) the proposed method, (**b**) FT-SMC, and (**c**) RBFN.

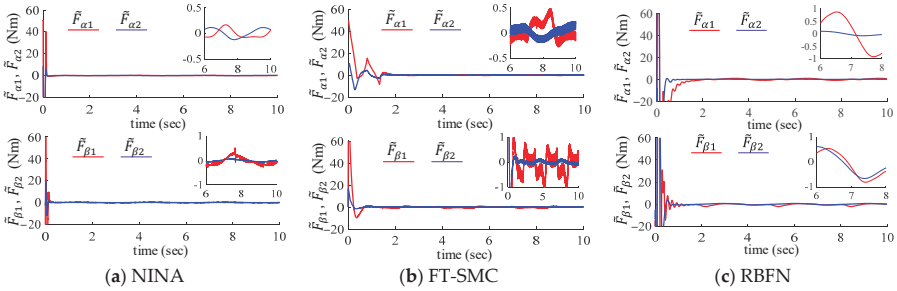
The tracking performance of the three controllers is illustrated in Figures 4 and 5. The steady-state tracking accuracy and convergence time are listed in Table 3. All three controllers can effectively track sinusoidal trajectories. Among them, the proposed algorithm exhibits smooth and fast convergence during the transient phase, while the other two methods show more pronounced oscillations. This is due to excessive control gain. As shown in Table 3, the tracking errors of the proposed control algorithm on the load side for joints 1 and 2 are, respectively $-0.07°$~$0.08°$ and $-0.03°$~$0.03°$; on the motor side, the tracking errors are $-0.11°$~$0.12°$ and $-0.13°$~$0.13°$, respectively. The overall steady-state tracking accuracy of the proposed algorithm is superior to the other two control algorithms.



(**a**) NINA (**b**) FT-SMC (**c**) RBFN

Figure 5. Motor-side tracking performance of (**a**) the proposed method, (**b**) FT-SMC, and (**c**) RBFN.

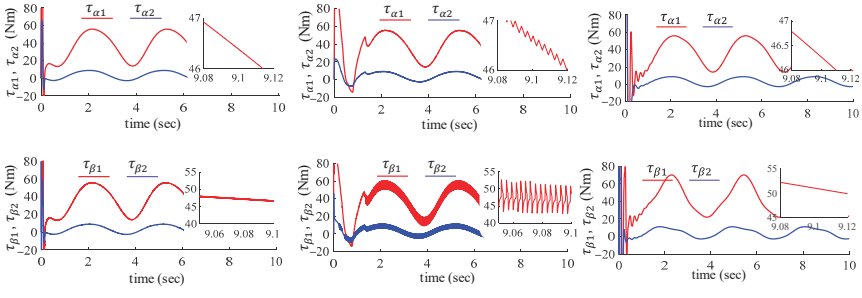
It can be seen from Figure 6 and Table 4 that the proposed algorithm achieves faster convergence of the nonlinear approximation error than the other two methods. This verifies its ability to track unknown disturbances with high dynamics. In addition, the proposed method also illustrates the high estimation accuracy of system nonlinearity.



(**a**) NINA (**b**) FT-SMC (**c**) RBFN

Figure 6. Approximation errors of the unknown system nonlinearity using (**a**) the proposed method, (**b**) the FT-SMC, and (**c**) the RBFN method.

Finally, the control signals of the three controllers are depicted in Figure 7. The control inputs of the proposed method and the RBFN-based method are milder, while the one of the FT-SMC shows significant chattering, especially on the motor side. It is a typical problem for sliding mode control. Compared with the traditional SMC method, the FT-SMC presented in [46] solved the peak phenomenon and suppressed the control chattering by asymptotical convergence, which is a considerable contribution. However, for nonlinear cascaded systems such as the flexible-joint manipulator, the control input of the outer loop (the load side) is usually mapped as the command of the inner loop (the motor side). This mapping process transmits the small chattering on the load side into the command layer of the motor side. The suppressed chattering is then amplified again by the motor-side control loop. As verified in Figure 7, the control input of the motor side contains obvious chattering, while the control input of the load side is milder.



(**a**) NINA (**b**) FT-SMC (**c**) RBFN

Figure 7. Control efforts of the proposed method (**top** row), the FT-SMC (**middle** column), and the RBFN method (**bottom** row).

In summary, the FT-SMC control shows good trajectory tracking accuracy and nonlinear estimation accuracy, but significant chattering occurs, which can lead to the failure of precision sensors and actuators in practical applications. Neural network-based control such as the RBFN-based method shows relatively lower convergence speed for nonlinearity approximation due to its comparably complex topology. In contrast, the proposed algorithm adopts a simple and effective estimation structure, which not only shows the ability for fast and accurate nonlinearity approximation but also maintains mild control input. This is also the major motivation for our research on this algorithm.

6.3. Robustness Validation

As shown in Figure 8, to further verify the stability and robustness of the proposed control, we examined the step response of the proposed method and its ability to recover from sudden disturbances. It can be observed that when encountering step changes, each joint can quickly track the new reference signal. The settling times for joints 1 and 2 are 0.618 s and 0.60 s, respectively. A 10 Nm impulse disturbance is introduced at 4 s and revoked at 6 s. It can be seen that the system can recover tracking of the original position within 2 s and has the ability to maintain a fixed point position with high precision (position tracking error $< 1 \times 10^{-5}$ degree).

Figure 9 compares the tracking performance of the proposed control method under different link lengths. Although the load environment has changed, the proposed adaptive control maintains high tracking performance. Figure 10 illustrates the dynamic behavior of the whole system under different joint stiffnesses. It is illustrated that all the synthetic tracking errors and nonlinearity estimation errors uniformly and asymptotically converge toward zero, regardless of the variation of the joint stiffness. These results verify the effectiveness and robustness of the proposed method.

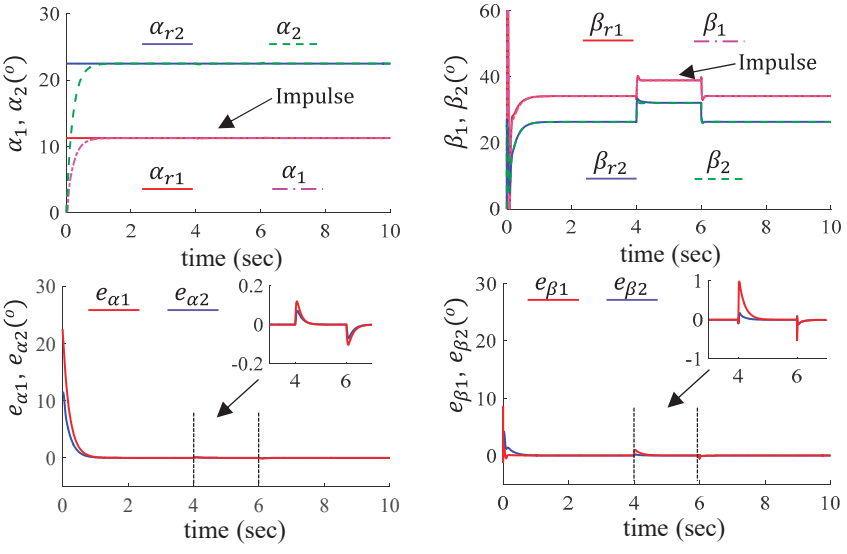


Figure 8. Tracking performance of the proposed method under step change and impulse disturbance. The first row shows the position of the load and motor sides, and the second row shows their tracking errors.

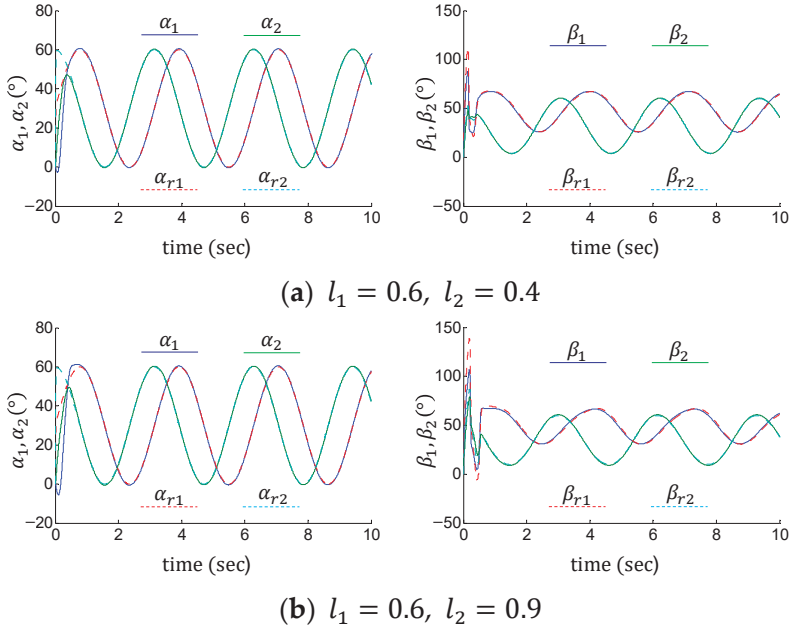


Figure 9. Results of tracking control under different link lengths using the proposed control method.

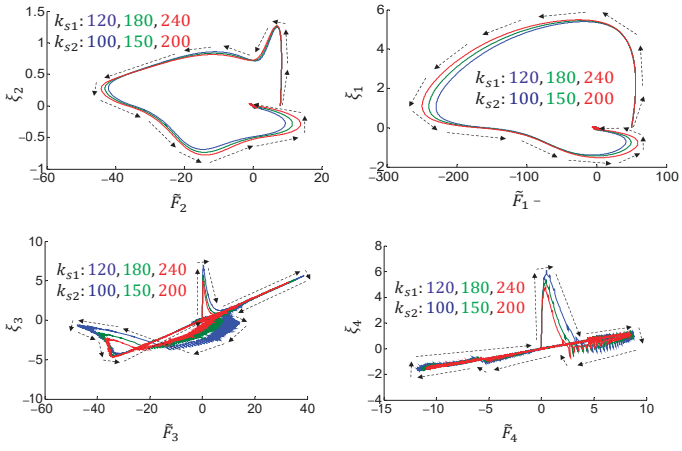


Figure 10. Dynamic behavior of the manipulator under different joint stiffness using the proposed control method.

7. Experiments

In this section, the proposed control method is further validated on a flexible-joint platform. The RBFN-based adaptive control method in [51] is introduced for comparison. Trajectory tracking experiments under different end loads are conducted.

7.1. Experiment Setup

Figure 11 shows a typical flexible-joint platform. From the left to right sides, there are a servo motor, a harmonic drive (with a 50:1 gear ratio), a flexible body, a torque sensor, and an output link with an end load. The flexible body here is a series of elastic actuators. The angular positions of load side α and motor side β are measured by optical encoders. The generated torque command τ_β is implemented through a servo driver. The torque sensor and signal detection-conversion card are employed to measure the output torque of load side τ_l and motor side τ_m, respectively. The nominal parameters of the platform are obtained via parameter identification and measurement, which are listed in Table 5.

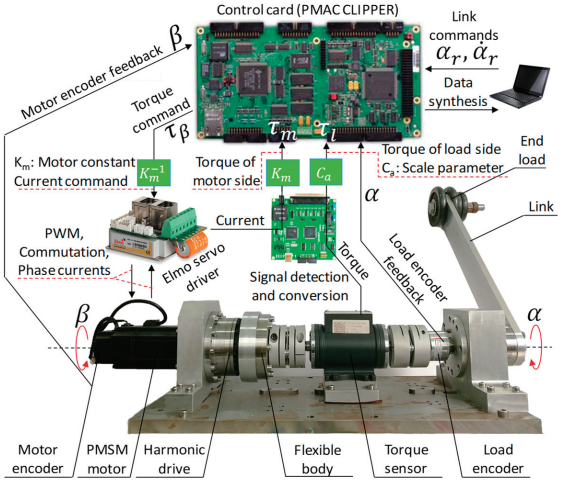


Figure 11. Architecture of the experimental system.

Table 5. Parameters of the flexible joint system.

Measured Values of Mechanical Parameters			Control Parameters			
J_α	1.090, under 2 kg end load	kg.m^2	ϑ_{s1}	1.125000	ϑ_{s2}	0.030000
	1.840, under 4 kg end load	kg.m^2	K_1	0.135000	K_2	0.010000
g_α	15.12, under 2 kg end load	N.m	Λ_1	26.50000	Λ_2	200.0000
	24.08, under 4 kg end load	N.m	A_1	0.000180	A_2	0.000001
J_β	4.65×10^{-4}	kg.m^2	λ_1	125000.0	λ_2	100000.0
k_s	927.0	N.m/rad				
k_d	1.54	N.m/rad/sec				
η	50.0	—				

Note that subscripts α and β represent the load and motor sides of the flexible-joint platform, respectively.

7.2. Experimental Results

Figures 12 and 13 show the tracking performance, nonlinearity approximations, and control inputs of the flexible joint using the proposed control method under 2 kg and 4 kg load conditions, respectively. The link action is set as follows: The initial posture of the link is vertically downward. It first rotates at a constant speed of 18°/s toward the horizontal level, then swings around the horizontal position. The swing amplitude and frequency are 21.6° and 0.8 Hz, respectively. The black dotted lines in the left column of Figure 12 indicate the horizontal position. The experimental results verify the tracking performance of the control system under both ramp and harmonic trajectories. The entire process is divided into three phases, i.e., the ramping phase, the switching phase, and the waving phase. The system exhibits transient responses in the switching phase (8 s to 10 s), due to the discontinuity of the velocity command $\dot{\alpha}_r$.

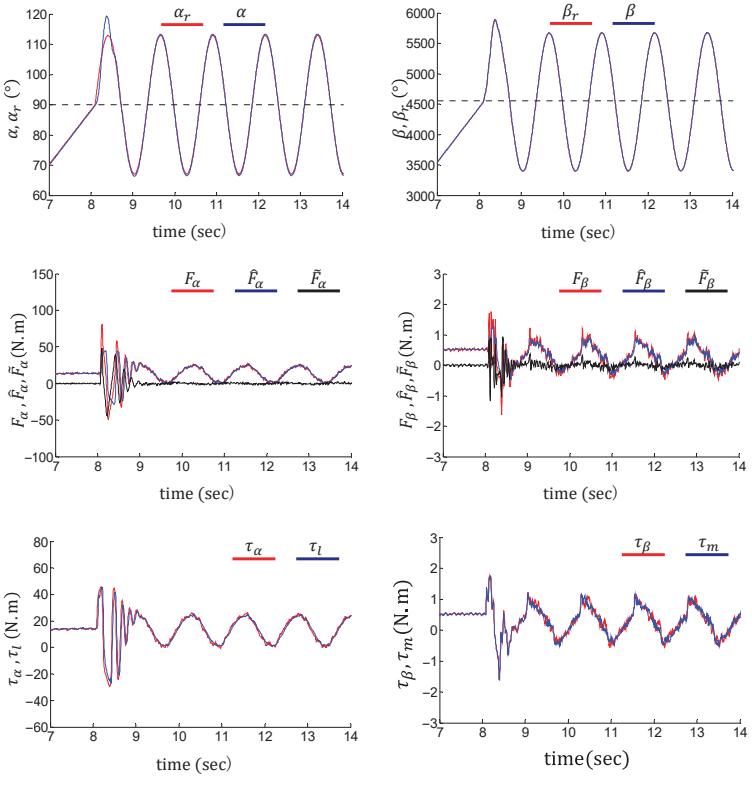


Figure 12. Performance of NINA-based adaptive control under a 2 kg end load.

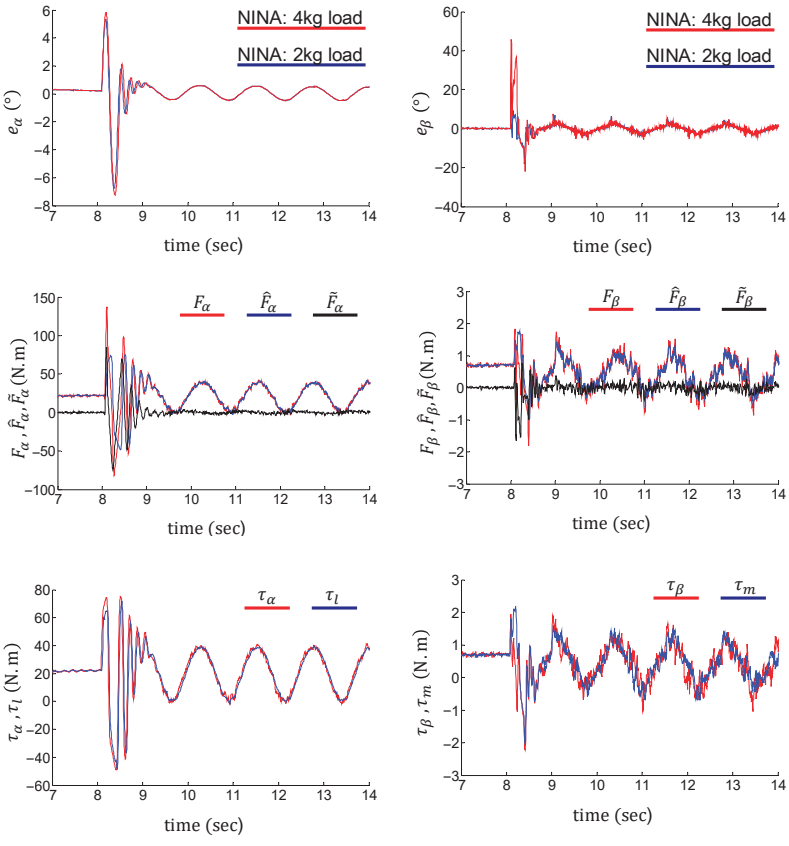


Figure 13. Performance of NINA-based adaptive control under a 4 kg end load.

As shown in the first rows of Figures 12 and 13, the control system stabilizes within 1 s during the switching phase. Tracking errors are limited within 0.5° and 10° on the load and motor sides (with a 50:1 gear ratio) during the ramping and waving phases, respectively. Although the flexible joint waves vertically under an end load, the tracking errors do not contain obvious biases.

The second rows of Figures 12 and 13 show accurate nonlinearity approximations, which validate the effectiveness of the NINA technique. The third rows of Figures 12 and 13 indicate that the above control performances are achieved under relatively clean control inputs. It is noteworthy that the tracking errors under different end loads are nearly identical. This verifies the robustness of the proposed control method.

A classical RBFN-based adaptive control presented in [51] is compared with the proposed control method. On the load side, $\Lambda_1 e_\alpha + \dot{\alpha}_r$, $\Lambda_1 \dot{e}_\alpha + \ddot{\alpha}_r$, α, and $\dot{\alpha}$ are supplied to the input layer of RBFN. On the motor side, $\Lambda_2 e_\beta + \dot{\beta}_r$, $\Lambda_2 \dot{e}_\beta + \ddot{\beta}_r$, β, and $\dot{\beta}$ are supplied to the input layer of RBFN. Five neurons are set in hidden layers on the load side and the motor side. The control torques are obtained from the output layer of the RBFN. For more details, please refer to [51].

The performance of RBFN control is shown in Figure 14. In the switching phase, the tracking error of the proposed method converges faster than the RBFN-based method. In the swing phase, the load- and motor-side tracking errors of the RBFN control are bounded by $|e_\alpha| < 1.75°$ and $|e_\beta| < 3°$, respectively. The tracking errors of the proposed NINA-based control method are bounded by $|e_\alpha| < 0.25°$ and $|e_\beta| < 3°$. In addition, the nonlinearity approximation of the RBFN control ($\hat{F}_\alpha$, $\hat{F}_\beta$) shows obvious lags behind their nominal

values (F_α, F_β), thereby resulting in relatively large estimation errors ($\widetilde{F}_\alpha$, $\widetilde{F}_\beta$). The above comparison indicates that the proposed NINA-based adaptive control can realize better control performance than the RBFN-based adaptive control on the flexible joint system.

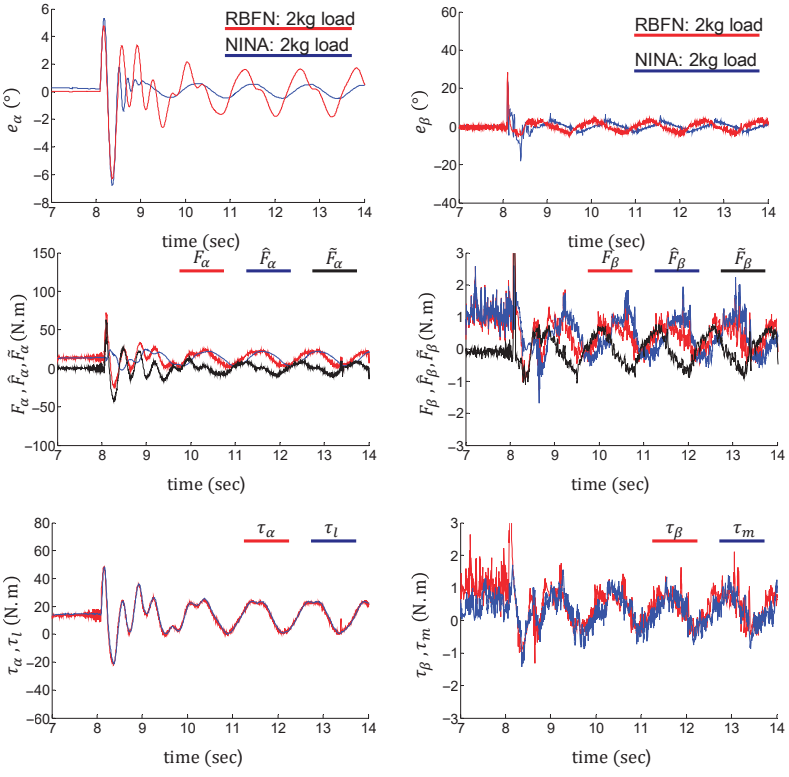


Figure 14. Performance of the RBFN adaptive controller under a 2 kg end load.

8. Conclusions

This study proposed a simplified adaptive control based on NINA for a class of nonlinear cascaded systems. The uniformity and ultimate stability of the proposed control were proven. The nonlinearities of each subsystem were approximated using the synthetic form of a steady component and an alternating component based only on local tracking errors. The proposed control method was validated through applications on the flexible joint system involving numerical simulations and experiments. The simulation results illustrated that the proposed method can achieve similar control accuracy as FT-SMC but uses milder control inputs. It was also indicated that the proposed method is insensitive to external loads and parametric perturbations. The proposed method was compared with an RBFN-based method. The experimental results demonstrated that the proposed method could achieve better control performance than an RBFN-based method.

Future work could be extended to flexible manipulators with variable stiffness. Future interests lie in two main areas: The first is optimizing the mapping process from the control input of the unactuated subsystem to the command layer of the actuated subsystem, which could improve the stability and noise level of the control system. The second is augmenting the adaptive law with a priori information on the system, to accelerate the convergence of the nonlinearity approximation.

Author Contributions: Conceptualization, Z.L.; formal analysis, Z.L.; methodology, Z.L. and H.J.; project administration, J.Z.; supervision, J.Z.; validation, Z.L.; writing—original draft, Z.L.; writing—review and editing, Z.L. and H.J. All authors have read and agreed to the published version of the manuscript.

Funding: The authors gratefully acknowledge the financial support of the National Natural Science Foundation of China under Grants 62373121, STI 2030-Major Projects 2021ZD0201400, and the National Natural Science Foundation of China under Grants 92048301.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

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

References

1. Han, S.I.; Lee, J.M. Fuzzy Echo State Neural Networks and Funnel Dynamic Surface Control for Prescribed Performance of a Nonlinear Dynamic System. *IEEE Trans. Ind. Electron.* **2014**, *61*, 1099–1112. [CrossRef]
2. Jin, H.; Lee, J. An RMRAC Current Regulator for Permanent-Magnet Synchronous Motor Based on Statistical Model Interpretation. *IEEE Trans. Ind. Electron.* **2009**, *56*, 169–177. [CrossRef]
3. Le-Tien, L.; Albu-Schäffer, A. Robust Adaptive Tracking Control Based on State Feedback Controller with Integrator Terms for Elastic Joint Robots with Uncertain Parameters. *IEEE Trans. Control Syst. Technol.* **2018**, *26*, 2259–2267. [CrossRef]
4. Xu, J.X.; Guo, Z.Q.; Lee, T.H. Design and Implementation of Integral Sliding-Mode Control on an Underactuated Two-Wheeled Mobile Robot. *IEEE Trans. Ind. Electron.* **2014**, *61*, 3671–3681. [CrossRef]
5. Han, S.I.; Lee, J.M. Balancing and Velocity Control of a Unicycle Robot Based on the Dynamic Model. *IEEE Trans. Ind. Electron.* **2015**, *62*, 405–413. [CrossRef]
6. Jin, H.; Hwang, J.; Lee, J. A Balancing Control Strategy for a One-Wheel Pendulum Robot Based on Dynamic Model Decomposition: Simulations and Experiments. *IEEE-ASME Trans. Mechatron.* **2011**, *16*, 763–768. [CrossRef]
7. 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. Mechatron.* **2010**, *15*, 17–26. [CrossRef]
8. Zhu, Y.H.; Gao, Y.S.; Xu, C.H.; Zhao, J.; Jin, H.Z.; Lee, J. Adaptive Control of a Gyroscopically Stabilized Pendulum and Its Application to a Single-Wheel Pendulum Robot. *IEEE-ASME Trans. Mechatron.* **2015**, *20*, 2095–2106. [CrossRef]
9. Zhu, Y.H.; Li, G.; Zhao, J.; Jin, H.Z. Attitude-Guided Robust Adaptive Path Following Control for Ducted Fan UAV. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 31–36.
10. Miranda-Colorado, R.; Aguilar, L.T. Robust PID control of quadrotors with power reduction analysis. *ISA Trans.* **2020**, *98*, 47–62. [CrossRef]
11. Li, J.; Zhang, G. Dynamic surface control for path following of the USV-UAV with time-varying disturbances. In Proceedings of the 2022 34th Chinese Control and Decision Conference (CCDC), Hefei, China, 15–17 August 2022; pp. 3115–3120.
12. Ccari, L.F.C.; Yanyachi, P.R. A Novel Neural Network-Based Robust Adaptive Formation Control for Cooperative Transport of a Payload Using Two Underactuated Quadcopters. *IEEE Access* **2023**, *11*, 36015–36028. [CrossRef]
13. Fridman, E. Near-optimal H-infinity control of linear singularly perturbed systems. *IEEE Trans. Autom. Control* **1996**, *41*, 236–240. [CrossRef]
14. Lu, G.P.; Yeung, L.E. H_∞-control problem for linear systems with multiple time-delays via dynamic output feedback. *Math. Comput. Simul.* **2002**, *60*, 335–345. [CrossRef]
15. Sun, L.Y.; Tong, S.C.; Liu, Y. Adaptive Backstepping Sliding Mode H_∞ Control of Static Var Compensator. *IEEE Trans. Control Syst. Technol.* **2011**, *19*, 1178–1185. [CrossRef]
16. Dai, Y.; Wang, D.; Shen, F.Y.; Iqbal, J. A robust optimal control by grey wolf optimizer for underwater vehicle-manipulator system. *PLoS ONE* **2023**, *18*, e0287405. [CrossRef] [PubMed]
17. Garcia, G.A.; Keshmiri, S.; Shukla, D. Nonlinear Control based on H-Infinity Theory for Autonomous Aerial Vehicle. In Proceedings of the 2017 International Conference on Unmanned Aircraft Systems (ICUAS'17), Miami, FL, USA, 13–16 June 2017; pp. 336–345.
18. Rigatos, G.; Abbaszadeh, M. Nonlinear optimal control for multi-DOF robotic manipulators with flexible joints. *Optim. Control Appl. Methods* **2021**, *42*, 1708–1733. [CrossRef]
19. Polyak, B.T.; Nazin, A.V.; Khlebnikov, M.V.; Nazin, S.A. Rejection of bounded disturbances via invariant ellipsoids technique. In Proceedings of the 45th IEEE Conference on Decision and Control, San Diego, CA, USA, 13–15 December 2006; pp. 1429–1434.
20. Khlebnikov, M.V.; Polyak, B.T.; Kuntsevich, V.M. Optimization of linear systems subject to bounded exogenous disturbances: The invariant ellipsoid technique. *Autom. Remote Control* **2011**, *72*, 2227–2275. [CrossRef]
21. Polyakov, A.; Poznyak, A. Invariant ellipsoid method for minimization of unmatched disturbances effects in sliding mode control. *Automatica* **2011**, *47*, 1450–1454. [CrossRef]

22. Gonzalez-Garcia, S.; Polyakov, A.E.; Poznyak, A.S. Using the method of invariant ellipsoids for linear robust output stabilization of spacecraft. *Autom. Remote Control* **2011**, *72*, 540–555. [CrossRef]
23. Gritli, H.; Belghith, S. LMI-based synthesis of a robust saturated controller for an underactuated mechanical system subject to motion constraints. *Eur. J. Control* **2021**, *57*, 179–193. [CrossRef]
24. Baleanu, D.; Hasanabadi, M.; Vaziri, A.M.; Jajarmi, A. A new intervention strategy for an HIV/AIDS transmission by a general fractional modeling and an optimal control approach. *Chaos Solitons Fractals* **2023**, *167*, 113078. [CrossRef]
25. Sereshki, Z.T.; Talebi, H.; Abdollahi, F. A nonlinear adaptive H_∞ optimal control method without solving HJIE: An analytical approach. *IEEE Trans. Autom. Control* **2024**. [CrossRef]
26. Ballesteros, M.; Fuentes-Aguilar, R.Q.; Chairez, I. Exponential Continuous Non-Parametric Neural Identifier with Predefined Convergence Velocity. *IEEE-CAA J. Autom. Sin.* **2022**, *9*, 1049–1060. [CrossRef]
27. Su, H.G.; Zhang, H.G.; Gao, D.W.Z.; Luo, Y.H. Adaptive Dynamics Programming for H_∞ Control of Continuous-Time Unknown Nonlinear Systems via Generalized Fuzzy Hyperbolic Models. *IEEE Trans. Syst. Man Cybern.-Syst.* **2020**, *50*, 3996–4008. [CrossRef]
28. Rigatos, G.; Busawon, K.; Abbaszadeh, M. A nonlinear optimal control approach for the truck and N-trailer robotic system. *IFAC J. Syst. Control* **2022**, *20*, 100191. [CrossRef]
29. Wang, Y.S.; Piao, M.N.; Chen, Z.Q.; Sun, M.W.; Sun, Q.L. Capability Exploration of Extended-State Observer-Based Control Under the Uncertain Case of Disturbance and Actuator Saturation. *IEEE Trans. Ind. Electron.* **2023**, *70*, 1841–1851. [CrossRef]
30. Altan, A.; Aslan, O.; Hacioglu, R. Real-Time Control based on NARX Neural Network of Hexarotor UAV with Load Transporting System for Path Tracking. In Proceedings of the 2018 6th International Conference on Control Engineering & Information Technology (CEIT), Istanbul, Turkey, 25–27 October 2018.
31. Li, Z.J.; Su, C.Y.; Li, G.L.; Su, H. Fuzzy Approximation-Based Adaptive Backstepping Control of an Exoskeleton for Human Upper Limbs. *IEEE Trans. Fuzzy Syst.* **2015**, *23*, 555–566. [CrossRef]
32. Yu, J.P.; Shi, P.; Dong, W.J.; Chen, B.; Lin, C. Neural Network-Based Adaptive Dynamic Surface Control for Permanent Magnet Synchronous Motors. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 640–645. [CrossRef]
33. Tong, S.C.; Sui, S.; Li, Y.M. Fuzzy Adaptive Output Feedback Control of MIMO Nonlinear Systems with Partial Tracking Errors Constrained. *IEEE Trans. Fuzzy Syst.* **2015**, *23*, 729–742. [CrossRef]
34. Li, Z.J.; Su, C.Y.; Wang, L.Y.; Chen, Z.T.; Chai, T.Y. Nonlinear Disturbance Observer-Based Control Design for a Robotic Exoskeleton Incorporating Fuzzy Approximation. *IEEE Trans. Ind. Electron.* **2015**, *62*, 5763–5775. [CrossRef]
35. Zhou, Q.; Shi, P.; Liu, H.H.; Xu, S.Y. Neural-Network-Based Decentralized Adaptive Output-Feedback Control for Large-Scale Stochastic Nonlinear Systems. *IEEE Trans. Syst. Man Cybern. Part B-Cybern.* **2012**, *42*, 1608–1619. [CrossRef]
36. Krstic, M.; Kokotovic, P.V.; Kanellakopoulos, I. *Nonlinear and Adaptive Control Design*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 1995.
37. Ma, J.J.; Zheng, Z.Q.; Li, P. Adaptive Dynamic Surface Control of a Class of Nonlinear Systems with Unknown Direction Control Gains and Input Saturation. *IEEE Trans. Cybern.* **2015**, *45*, 728–741. [CrossRef] [PubMed]
38. Chen, C.; Xie, L.H.; Xie, K.; Lewis, F.L.; Xie, S.L. Adaptive optimal output tracking of continuous-time systems via output-feedback-based reinforcement learning. *Automatica* **2022**, *146*, 110581. [CrossRef]
39. Won, D.; Kim, W.; Shin, D.; Chung, C.C. High-gain disturbance observer-based backstepping control with output tracking error constraint for electro-hydraulic systems. *IEEE Trans. Control Syst. Technol.* **2014**, *23*, 787–795. [CrossRef]
40. Khalil, H.K. High-gain observers in nonlinear feedback control. In *New Directions in Nonlinear Observer Design*; Springer: London, UK, 2007; pp. 249–268.
41. Adil, A.; N'Doye, I.; Laleg-Kirati, T.M. High-Gain Observer Design for Nonlinear Systems with Delayed Output Measurements using Time-Varying Gains. In Proceedings of the 2022 IEEE 61st Conference on Decision and Control (CDC), Cancún, Mexico, 6–9 December 2022; pp. 235–240.
42. Jaber, L.; Ichalal, D.; Oufroukh, N.A.; Nouvelière, L.; Mammar, S. A New High Gain Observer Based on Artificial Delays. In Proceedings of the 2023 IEEE International Conference on Networking, Sensing and Control (ICNSC), Marseille, France, 25–27 October 2023; pp. 1–6.
43. Vaidyanathan, S.; Lien, C.-H. *Applications of Sliding Mode Control in Science and Engineering*; Springer: Cham, Switzerland, 2017; Volume 709.
44. Cheng, X.; Liu, H.S.; Lu, W.K. Chattering-Suppressed Sliding Mode Control for Flexible-Joint Robot Manipulators. *Actuators* **2021**, *10*, 288. [CrossRef]
45. Fateh, A.; Momeni, H.; Masouleh, M.T. Taylor-based adaptive sliding mode control method for robot manipulators. *IET Control. Theory Appl.* **2023**, *17*, 1105–1115. [CrossRef]
46. Miranda-Colorado, R. Finite-time sliding mode controller for perturbed second-order systems. *ISA Trans.* **2019**, *95*, 82–92. [CrossRef] [PubMed]
47. Gonzalezvelasco, E.A. Connections in Mathematical Analysis—The Case of Fourier-Series. *Am. Math. Mon.* **1992**, *99*, 427–441. [CrossRef]
48. Polycarpou, M.M.; Ioannou, P.A. A robust adaptive nonlinear control design. *Automatica* **1996**, *32*, 423–427. [CrossRef]
49. Zhao, Z.J.; Liu, Z.J.; He, W.; Hong, K.S.; Li, H.X. Boundary adaptive fault-tolerant control for a flexible Timoshenko arm with backlash-like hysteresis. *Automatica* **2021**, *130*, 109690. [CrossRef]

50. Ma, Y.H.; He, X.Y.; Zhang, S.; Sun, Y.B.; Fu, Q. Adaptive Compensation for Infinite Number of Actuator Faults and Time-Varying Delay of a Flexible Manipulator System. *IEEE Trans. Ind. Electron.* **2022**, *69*, 13141–13150. [CrossRef]
51. Lee, M.J.; Choi, Y.K. An adaptive neurocontroller using RBFN for robot manipulators. *IEEE Trans. Ind. Electron.* **2004**, *51*, 711–717. [CrossRef]
52. Spong, M.W. Modeling and Control of Elastic Joint Robots. *J. Dyn. Syst. Meas. Control.-Trans. ASME* **1987**, *109*, 310–319. [CrossRef]

sensors

MDPI

Article

Adhesion Coefficient Identification of Wheeled Mobile Robot under Unstructured Pavement

Hongchao Zhang [1,2], Bao Song [1], Junming Xu [1], Hu Li [3,4] and Shuhui Li [2,*]

[1] School of Mechanical Science & Engineering, Huazhong University of Science and Technology, Wuhan 430074, China; d202187005@hust.edu.cn (H.Z.); songbao@hust.edu.cn (B.S.); xjm0627@163.com (J.X.)
[2] North China Vehicle Research Institute, Beijing 100072, China
[3] Guangdong Intelligent Robotics Institute, Dongguan 523830, China; drhuli@foxmail.com
[4] Hubei Key Laboratory of Intelligent Robot, Wuhan Institute of Technology, Wuhan 430205, China
* Correspondence: dreamy_on2023@163.com

Abstract: Because of its uneven and large slope, unstructured pavement presents a great challenge to obtaining the adhesion coefficient of pavement. An estimation method of the peak adhesion coefficient of unstructured pavement on the basis of the extended Kalman filter is proposed in this paper. The identification accuracy of road adhesion coefficients under unstructured pavement is improved by introducing the equivalent suspension model to optimize the calculation of vertical wheel load and modifying vehicle acceleration combined with vehicle posture data. Finally, the multi-condition simulation experiments with Carsim are conducted, the estimation accuracy of the adhesion coefficient is at least improved by 3.6%, and then the precision and effectiveness of the designed algorithm in the article are verified.

Keywords: unstructured pavement; state estimation; extended Kalman filter; equivalent suspension model; pavement adhesion coefficient

Citation: Zhang, H.; Song, B.; Xu, J.; Li, H.; Li, S. Adhesion Coefficient Identification of Wheeled Mobile Robot under Unstructured Pavement. *Sensors* **2024**, *24*, 1316. https://doi.org/10.3390/s24041316

Academic Editor: Hui Kong

Received: 26 December 2023
Revised: 9 February 2024
Accepted: 16 February 2024
Published: 18 February 2024

1. Introduction

In the field of intelligent vehicles, the unstructured pavement is characterized by complexity and randomness, which more easily causes safety problems and has higher performance requirements for vehicle safety control [1–3]. Safety control of vehicles often needs to adjust the force between road surfaces and tires, and the interaction force is also a key and vital factor that affects the stability of the vehicle chassis [4,5]. Therefore, the adhesion coefficient of road surfaces is a vital parameter to obtain accurate motion control of vehicles [6,7], and it also provides a significant input for the decision and planning of intelligent vehicles. It is vital and essential to precisely identify the adhesion coefficients of the unstructured pavements for the safe driving of vehicles.

Recently, a variety of methods have been proposed by domestic and foreign scholars to estimate the road adhesion coefficient. At present, road adhesion coefficient identification mainly consists of cause-based methods as well as effect-based methods [8–11]. With the development of vehicle intelligence, some special devices (optical sensor or ultrasonic sensor, etc.) are required to measure the factors associated with tires or roads (for example, the deformation and noise of tires, the road texture, etc.), and then identify the road adhesion coefficient in the cause-based methods [12,13]. For example, given the critical and difficult problems about the estimation of adhesion coefficients, Bo Leng et al. proposed fusing vehicle dynamics with machine vision during the estimation [14]. A local binarization algorithm was designed by Du et al. to extract the spatial and texture features of roads gathered by the high-definition cameras [15]. Then, the feature information was introduced into the modified VGGNet to classify the road adhesion level. Herrmann T. et al. adopted an on-board camera and lidar to first estimate the road adhesion coefficient and then collect it based on dynamic information [16]. In a word, the estimation accuracy of cause-based

methods is high, but the special measuring equipment in the experiment is expensive and susceptible to fog, rain, and snow.

Effect-based estimation methods use sensors to study tire and road-related factors (for example, the deformation and noise of tires, the road texture, etc.). Then the road adhesion coefficient is calculated by these factors. The estimation methods based on the curve of the adhesion coefficient and the slip rate [17], together with the relationship between the right torque and the side deflection angle of tires [18], are common in the vehicle field, which only needs the common on-board sensors, such as wheel speed sensors and attitude sensors. For example, aligning the torque and the side-slip angle or the relationship description between the tire lateral force is performed through linearization [19], Brush [20,21], TMsimple [22], Pacejka [23], and Burckhardt [24] to identify adhesion coefficients under the condition of lateral movements of the vehicle. However, such a method based on the response recognition of tires is affected by many external uncertainties due to the complexity of the generation mechanism of the tire noise, and sometimes it is insurmountable to accurately identify the adhesion coefficients. Recently, some identification methods on the basis of visual information have been proposed in the current study [25–28]. For example, given the nondeterminacy of kinematic models and deep-learning models, an image-based fusion estimation method by virtue of the virtual sensing theory was put forward to exactly realize the identification of the road surface condition in reference [25]. However, these visual information-based methods are susceptible to light. In addition, scholars make the best use of different vehicle dynamics models [29–34] and various kinds of filters [35–38] to estimate the adhesion coefficient. For example, the vehicle dynamic model, the tire model, and the wheel model were introduced into the unscented Kalman filter to accurately identify the adhesion coefficient of roads in reference [36]. According to the similarity principle and the adaptive square root cubature Kalman filter, a longitudinal-lateral cooperative estimation algorithm was designed to identify the state of vehicles and the adhesion coefficients of four-wheel independent drive electric vehicles in reference [38]. In short, most of the above algorithms build the vehicle dynamics model with the seven-degree-of-freedom (VDMSDF) and simulate it with the Carsim 2019.0. The complexity and freedom of vehicle models provided by Carsim are much higher than those of the commonly used VDMSDF. Real data should be directly used to replace some parameters that are difficult to fit with the models, which are closer to the actual vehicle model. In addition, the filters, by virtue of two types of models, are adopted to accurately estimate the adhesion coefficients of the pavement under the coupled conditions of lateral and longitudinal forces, which is simple to solve and has fast convergence speed. However, this type of filter-based method does not fully consider the surface roughness and the large slope characteristics of the unstructured pavement, and still faces some challenges:

1. The existing algorithm regards the body and the wheel as a rigid connection, which leads to a large error in the calculated vertical load of the tire under the conditions of the uneven road.
2. The existing algorithms often ignore the influence of gravity on the body acceleration, and straightforwardly input the body acceleration into the observer as an observation quantity, which makes the algorithm unable to accurately estimate the road surface with a certain slope.

Therefore, considering the effects of vertical static and dynamic loads on vehicle acceleration, an estimation method of adhesion coefficients of the pavement on the basis of the extended Kalman filter is proposed for the above issue. The paper primarily investigates the identification of adhesion coefficients under uneven and slope pavement, and it is assumed that all tires together with all wheels are the same. So, the main contributions and highlights of the proposed method are highlighted in the following sentences:

3. The impact of vehicle acceleration on the vertical load is fully considered in the vertical load model of vehicles based on the equivalent suspension model, which can depict road conditions by rule and line.

4. The dynamics model of vehicles established in this paper no longer regards the body and the wheel as rigid body connections, which can more accurately depict the dynamic characteristics of vehicles.
5. By virtue of the extended Kalman filter and the improved dynamic model, the estimation algorithm of adhesion coefficient identification on the unstructured pavement designed in this paper can not only adapt to the non-structural and high-slope pavement but also improve the identification accuracy of adhesion coefficients.

The rest of the paper is approximately structured as follows: In Section 2, the improved dynamics model is discussed first, which is foundation of the system dynamics model in the later content. In Section 3, the identification of adhesion coefficients on the basis of the extended Kalman filter is designed to estimate the tire-road peak adhesion coefficient. In Section 4, the designed identification algorithm in this paper is proven through simulation tests. Finally, the conclusion as well as the application scenario of the proposed method is given.

2. Improved Dynamics Model

The adhesion coefficient is related to the force of the tire. Therefore, this paper gradually analyzes the forces from the whole vehicle to the tires and then decomposes the forces from tires to the vertical load. The relationship between the adhesion coefficient and the force of the tire is established through the force analysis in the above steps. Therefore, this chapter will expand the description from the dynamics model of vehicles, tire models and vertical load in turn.

2.1. Improved Vehicle Dynamics Model

The vehicle dynamics model can simulate and analyze the motion states of vehicles under different driving scenarios and represent the mathematical relationship among different control inputs, environmental inputs, and vehicle responses during driving, which is the basis of the research for adhesion coefficient identification. At present, the VDMSDF is commonly applied to research on adhesion coefficient identification. It combines the vehicle dynamics model with three degrees of freedom and the wheel model. In addition, the rotational degree of freedom for four wheels is considered, except for the freedom of longitudinal, transverse, and yaw.

Given Figure 1, the following equation can be calculated according to the vehicle dynamics model:

$$\dot{v}_x - v_y\omega_z = a_x \tag{1}$$

$$\dot{v}_y + v_x\omega_z = a_y \tag{2}$$

The longitudinal dynamic formula is approximately described as the following equation:

$$m\left(\dot{v}_x - v_y\omega_z\right) = F_{x,fl}\cos\delta + F_{x,fr}\cos\delta + F_{x,rl} + F_{x,rr} - F_{y,fl}\sin\delta - F_{y,fr}\sin\delta \tag{3}$$

Similarly, the lateral dynamics equation can be obtained as follows:

$$m\left(\dot{v}_y + v_x\omega_z\right) = F_{x,fl}\sin\delta + F_{x,fr}\sin\delta + F_{y,rl} + F_{y,rr} + F_{y,fl}\cos\delta + F_{y,fr}\cos\delta \tag{4}$$

The yaw motion dynamics equation can be calculated in the same way:

$$\begin{aligned} I_z\dot{\omega}_z = {} & \tfrac{B_f}{2}\left(F_{x,fr}\cos\delta - F_{x,fl}\cos\delta - F_{y,fr}\sin\delta + F_{y,fl}\sin\delta\right) + \tfrac{B_r}{2}\left(F_{x,rr} - F_{x,rl}\right) \\ & + L_f\left(F_{x,fl}\sin\delta + F_{y,fl}\cos\delta + F_{y,fr}\sin\delta + F_{y,fl}\sin\delta\right) - L_r\left(F_{y,rl} + F_{y,rr}\right) \end{aligned} \tag{5}$$

where v_x and v_y are the longitudinal speed and the lateral speed, respectively. ω_z is the angular velocity of the yaw. δ is the angle of the front wheel. m is the total vehicle

mass, $F_{x,ij}(ij = fl, fr, rl, rr)$ is the longitudinal force of the tire on the wheels. $F_{y,ij}(ij = fl, fr, rl, rr)$ is the transverse force for four-wheel tires. The moment of inertia is set to I_z when vehicles rotate about the z axis. The front and rear tracks are assumed as B_f and B_r, respectively. The distances from the center of mass to the front and rear axles are respectively set to L_f and L_r.

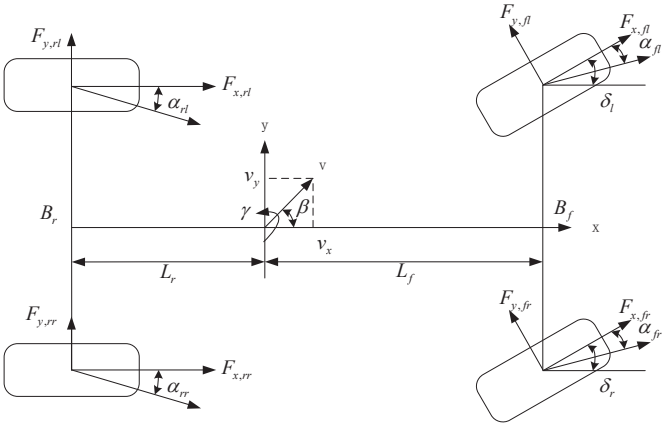


Figure 1. Vehicle dynamics model.

The above model is only suitable for the horizontal road. When the vehicle is driving on a slope road, gravity affects the acceleration of the vehicle. In this case, there is also acceleration caused by gravity, except for the acceleration caused by tire stress. Therefore, it is indispensable to take the influence of gravity into consideration and modify the above dynamics expressions.

It is vital and necessary to take the effects of gravity into account on a slop road. The acceleration component caused by the tire force is calculated by combining the attitude and acceleration of the vehicles.

$$\begin{cases} a_{Fx} = a_x - g\sin\theta \\ a_{Fy} = a_y - g\sin\phi \end{cases} \tag{6}$$

The longitudinal dynamics equation and the transverse dynamics equation are calculated as follows:

$$ma_{Fx} = F_{x,fl}\cos\delta + F_{x,fr}\cos\delta + F_{x,rl} + F_{x,rr} - F_{y,fl}\sin\delta - F_{y,fr}\sin\delta \tag{7}$$

$$ma_{Fy} = F_{x,fl}\sin\delta + F_{x,fr}\sin\delta + F_{y,rl} + F_{y,rr} + F_{y,fl}\cos\delta + F_{y,fr}\cos\delta \tag{8}$$

where a_{Fx} is the longitudinal acceleration component caused by the tire stress, a_{Fy} is the transverse acceleration component caused by the tire stress, ϕ is the roll angle of vehicles, θ is the pitch angle of vehicles.

Given the established dynamics model for vehicles, it is necessary to further calculate the longitudinal and lateral forces of tires by the relationship of the wheel force.

2.2. Dugoff Tire Model

The tire model describes the relationship between the tire force and the motion parameters of wheels through mathematical relations. In other words, it is the relationship between the inputs of tires and the outputs of tires under different road conditions. The longitudinal force, the transverse force, and the righting moment in the tire model are usually calculated by the input parameters such as the slip rate, the side deflection angle, and the vertical load.

The Dugoff tire model can describe the relationship between the road adhesion coefficient and the lateral and longitudinal forces. Meanwhile, this model can directly calculate the longitudinal force and the lateral force according to the slip rate and the side deflection angle of the vehicle. The Dugoff tire model is expressed as follows [39–42]:

$$F_x = C_x \frac{\lambda}{1+\lambda} \times f(L) \tag{9}$$

$$F_y = C_y \frac{\tan \alpha}{1+\lambda} \times f(L) \tag{10}$$

where L and $f(L)$ can be respectively expressed as follows:

$$L = \frac{\mu F_z (1+\lambda)}{2\sqrt{{C_x}^2 \lambda^2 + {C_y}^2 \tan^2 \alpha}} \tag{11}$$

$$f(L) = \begin{cases} L(2-L), L < 1 \\ 1 \quad\quad , L \geq 1 \end{cases} \tag{12}$$

C_x and C_y are the longitudinal and lateral stiffness of tires, respectively. λ is the slip rate. α is the lateral drift angle of tires. L is the nonlinear characteristic parameter used to express the slip of tires.

In order to effectively identify the adhesion coefficients, it is necessary and vital to reorganize the Dugoff tire model so that the dominant relational expression between the road adhesion coefficient and the Dugoff tire model can be directly obtained. Let $C_x^0 = C_x/\mu F_z$ and $C_y^0 = C_y/\mu F_z$. Considering the calculation formulas for the lateral and longitudinal forces in the Dugoff tire model, a nonlinear characteristic parameter of tires under slipping L can be expressed:

$$L = \frac{1+\lambda}{2\sqrt{\left(C_x^0\right)^2 \lambda^2 + \left(C_y^0\right)^2 \tan^2 \alpha}} \tag{13}$$

Next, the lateral force and longitudinal force of tires can be calculated by the following equations:

$$F_x = \mu F_z C_x^0 \frac{\lambda}{1+\lambda} \times f(L) \tag{14}$$

$$F_y = \mu F_z \cdot C_y^0 \frac{\tan \alpha}{1+\lambda} \times f(L) \tag{15}$$

It can be seen from the calculation formula of L that this deformation does not change the value of L, nor does it change the calculation of lateral and longitudinal forces. Moreover, the product of parameters in the formula except for adhesion coefficients after deformation is set to the normalized force, namely F_x^0 and F_y^0. Then the following formula can be obtained:

$$F_x = \mu {F_x}^0 \tag{16}$$

$$F_y = \mu {F_y}^0 \tag{17}$$

Therefore, the actual tire force can be expressed as the dynamic equation through the normalized force and the adhesion coefficient of roads, given the corresponding relationship between the force of tires and the adhesion coefficient in the Dugoff tire model.

It is obvious that the model in Equations (17) and (18) can be applied to the algorithm after the mathematical deformation, which is conducive to the subsequent research on adhesion coefficient identification. Therefore, the deformable Dugoff tire model is adopted in identification algorithms.

Therefore, considering the side deflection angles, the slip rate, and the vertical load, the Dugoff normalized force can be calculated as follows:

$$\begin{cases} {F_x}^0 = F_z \cdot C_x \frac{\lambda}{1+\lambda} \times f(L) \\ {F_y}^0 = F_z \cdot C_y \frac{\tan\alpha}{1+\lambda} \times f(L) \end{cases} \tag{18}$$

Since the force of tires changes with the speed of vehicles during the driving process, add a speed correction $1 - \varepsilon v_x \sqrt{{C_x}^2\lambda^2 + {C_y}^2 \tan^2\alpha}$ to the nonlinear characteristic parameters of tires under slipping L, namely:

$$L = \frac{\mu F_z(1+\lambda)}{2\sqrt{{C_x}^2\lambda^2 + {C_y}^2 \tan^2\alpha}} \left(1 - \varepsilon v_x \sqrt{{C_x}^2\lambda^2 + {C_y}^2 \tan^2\alpha}\right) \tag{19}$$

where ε is the influence factor of the velocity. It is only related to the material and structure of tires and can be utilized to describe the effect of the slip speed on the force of tires. In addition, the Dugoff tire model requires that the vertical load, the lateral drift angle of tires, and the slip rate be known, so that the slip rate and the lateral drift angle of tires can be calculated from the state parameters of vehicles.

Firstly, the vehicle speed signal, the front wheel angle signal, and the yaw angle speed signal are obtained by the sensor installed on the vehicles. On this basis, the lateral drift angle of wheels α_{ij} can be calculated using the following equations:

$$\alpha_{fl} = -\left(\delta - \arctan\frac{v_y + L_f\omega_z}{v_x - \frac{B_f\omega_z}{2}}\right) \tag{20}$$

$$\alpha_{fr} = -\left(\delta - \arctan\frac{v_y + L_f\omega_z}{v_x + \frac{B_f\omega_z}{2}}\right) \tag{21}$$

$$\alpha_{rl} = \arctan\frac{v_y - L_r\omega_z}{v_x - \frac{B_r\omega_z}{2}} \tag{22}$$

$$\alpha_{rr} = \arctan\frac{v_y - L_r\omega_z}{v_x + \frac{B_r\omega_z}{2}} \tag{23}$$

In Equations (20)–(23), v_x and v_y are the longitudinal speed and the transverse speed of vehicles, respectively. ω_z is the yaw angle speed of vehicles. δ is the angle of the front wheel. B_f and B_r are the front wheel base and the rear wheel base of vehicles, respectively. L_f and L_r are the distances from the center of mass to the front axles and the rear axles, respectively. The speed in the longitudinal axis direction of the core wheel under the wheel coordinate system $v_{wx,ij}$ is calculated as follows:

$$v_{wx,fl} = \left(v_x - \frac{B_f\omega_z}{2}\right)\cos\delta + \left(v_y + L_f\omega_z\right)\sin\delta \tag{24}$$

$$v_{wx,fr} = \left(v_x + \frac{B_f\omega_z}{2}\right)\cos\delta + \left(v_y + L_f\omega_z\right)\sin\delta \tag{25}$$

$$v_{wx,rl} = v_x - \frac{B_r\omega_z}{2} \tag{26}$$

$$v_{wx,rr} = v_x + \frac{B_r\omega_z}{2} \tag{27}$$

The wheel speed is obtained through the wheel speed sensor. The slip rate of wheels λ_{ij} is obtained as the following equation:

$$\lambda_{ij} = \frac{\omega_{w,ij} R_W - v_{wx,ij}}{\max\left(\omega_{w,ij} R_W, v_{wx,ij}\right)} \tag{28}$$

where $\omega_{w,ij}$ represents the speed of wheels, R_W is the radius of wheels, $ij \in \{fl, fr, rl, rr\}$. In addition to the acquisition of the lateral drift angle and the slip rate, it is also necessary to analyze the vertical load during the movement. The calculation of the vertical load can be discussed in the following chapter.

2.3. Improved Vertical Load Model

It is necessary for the identification of the adhesion coefficients to obtain the side deflection angle of tires, the vertical load, together with the slip rate. Under a non-structural road, the vertical load is often affected by the vehicle's vertical acceleration, so it is necessary to build an estimation model of the vertical load of tires with higher accuracy. A vertical load model based on the equivalent suspension is established in this paper.

Firstly, the vertical static load of wheels F_{w_ij} can be calculated by the self-propelled parameters:

$$F_{w\_fl} = \frac{1}{2} \times \frac{m_b g L_r}{L_f + L_r} + m_w g \tag{29}$$

$$F_{w\_rl} = \frac{1}{2} \times \frac{m_b g L_r}{L_f + L_r} + m_w g \tag{30}$$

$$F_{w\_rl} = \frac{1}{2} \times \frac{m_b g L_f}{L_f + L_r} + m_w g \tag{31}$$

$$F_{w\_rr} = \frac{1}{2} \times \frac{m_b g L_f}{L_f + L_r} + m_w g \tag{32}$$

In Equations (29)–(32), m_b is the spring mass of the vehicle. m_W is the wheel mass; The acceleration of the gravity is set to g.

In Figure 2, based on the models of the roll motion and the pitch motion of vehicles, the displacement, together with the speed of suspensions, is analyzed as follows:

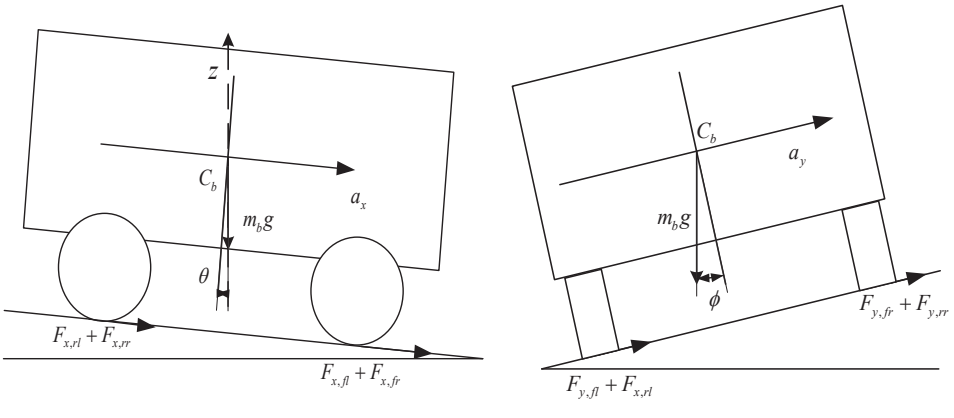


Figure 2. Roll and pitch motion models of vehicles.

The displacements of the suspension for each wheel are calculated by the roll and pitch angle of the body together with the centroid displacement:

$$z_{s\_fl} = z_b + \frac{B_f}{2}\sin\phi - L_f \sin\theta \tag{33}$$

$$z_{s\_fr} = z_b - \frac{B_f}{2}\sin\phi - L_f \sin\theta \tag{34}$$

$$z_{s\_rl} = z_b + \frac{B_r}{2}\sin\phi + L_r \sin\theta \tag{35}$$

$$z_{s\_rr} = z_b - \frac{B_r}{2}\sin\phi + L_r \sin\theta \tag{36}$$

In Equations (33)–(36), z_b is the vertical displacement of the vehicle centroid. The roll angle of the body is set as ϕ. Set θ as the pitch angle of the body. The suspension velocity is obtained by differentiating the suspension displacement:

$$\dot{z}_{s\_fl} = \dot{z}_b + \frac{B_f}{2}\cos\phi \times \dot{\phi} - L_f \cos\theta \times \dot{\theta} \tag{37}$$

$$\dot{z}_{s\_fr} = \dot{z}_b - \frac{B_f}{2}\cos\phi \times \dot{\phi} - L_f \cos\theta \times \dot{\theta} \tag{38}$$

$$\dot{z}_{s\_rl} = \dot{z}_b + \frac{B_r}{2}\cos\phi \times \dot{\phi} + L_r \cos\theta \times \dot{\theta} \tag{39}$$

$$\dot{z}_{s\_rr} = \dot{z}_b - \frac{B_r}{2}\cos\phi \times \dot{\phi} + L_r \cos\theta \times \dot{\theta} \tag{40}$$

The vertical acceleration and the vertical displacement of the wheel are obtained by the accelerometer and the angle sensor, respectively, and then the vertical speed signals are processed. The dynamic suspension force is calculated as follows:

$$F_{dzs\_fl} = k_f\left(z_{w\_fl} - z_{s\_fl}\right) + c_f\left(\dot{z}_{w\_fl} - \dot{z}_{s\_fl}\right) \tag{41}$$

$$F_{dzs\_fr} = k_f\left(z_{w\_fr} - z_{s\_fr}\right) + c_f\left(\dot{z}_{w\_fr} - \dot{z}_{s\_fr}\right) \tag{42}$$

$$F_{dzs\_rl} = k_r(z_{w\_rl} - z_{s\_rl}) + c_r\left(\dot{z}_{w\_rl} - \dot{z}_{s\_rl}\right) \tag{43}$$

$$F_{dzs\_rr} = k_r(z_{w\_rr} - z_{s\_rr}) + c_r\left(\dot{z}_{w\_rr} - \dot{z}_{s\_rr}\right) \tag{44}$$

In Equations (41)–(44), z_{w_ij} denotes the vertical displacement of each wheel; k_f and k_r are the equivalent stiffness coefficients of the suspension, respectively. c_f and c_r are the equivalent damping coefficients of the suspension, respectively. The force analysis of wheels is shown in Figure 3:

The dynamic vertical load of tires can be calculated by the force of dynamic suspension and wheel acceleration:

$$F_{dw\_fl} = m_w \ddot{z}_{w\_fl} - F_{ds\_fl} \tag{45}$$

$$F_{dw\_fr} = m_w \ddot{z}_{w\_fr} - F_{ds\_fr} \tag{46}$$

$$F_{dw\_rl} = m_w \ddot{z}_{w\_rl} - F_{ds\_rl} \tag{47}$$

$$F_{dw\_rr} = m_w \ddot{z}_{w\_rr} - F_{ds\_rr} \tag{48}$$

The current vertical load of wheels is obtained by adding the calculated vertical static load and the dynamic load.

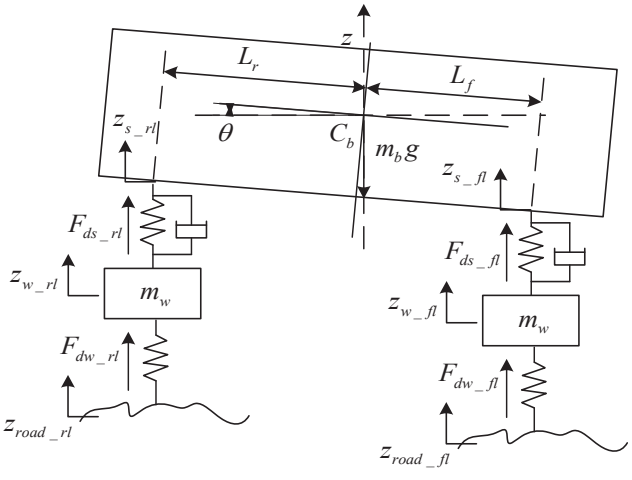


Figure 3. Force analysis of wheels.

3. Adhesion Coefficient Identification by Virtue of Extended Kalman Filter

3.1. Description of System Equations Based on the Improved Dynamics Model

Considering that the influencing factors of the adhesion are complex and the reproducibility of the adhesion is poor in different scenarios, based on the principle of grasping the main contradiction, the modeling errors caused by the sensor noise, limited/limited sampling time, unmodeled dynamics, and other factors in the paper are equivalent to Gaussian noise in the dynamic model about the road adhesion coefficient. In addition, due to the nonlinearity of the established dynamic equation about the adhesion coefficient, it is necessary to select a suitable filter for estimation. The effectiveness of the extended Kalman filter has been recognized by many scholars and engineers because it can take into account the nonlinear modeling error of dynamic models about the adhesion coefficient. Meanwhile, considering the real-time performance of the extended Kalman filter, this paper selects the extended Kalman filter to identify the adhesion coefficient. Thus, the state equation for the designed vehicle dynamics is given as follows:

$$x(t) = f(x(t), u(t), w(t)) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \mu_{fl} \\ \mu_{fr} \\ \mu_{rl} \\ \mu_{rr} \end{bmatrix} + w(t) \tag{49}$$

where $w(t)$ is the process noise, the mean square error is Q, μ_{ij} is the adhesion coefficient of the four-lane highway.

The actual tire force in the three dynamics equations of vehicles can be written in the form of the normalized force. Then the nonlinear equation is linearized. The measurement equation is expressed as follows:

$$\begin{aligned} y(t) &= h(x(t), v(t)) \\ &= \begin{bmatrix} h(1,1) & h(1,2) & \frac{F^0{}_{xrl}}{m} & \frac{F^0{}_{xrr}}{m} \\ h(2,1) & h(2,2) & \frac{F^0{}_{yrl}}{m} & \frac{F^0{}_{yrr}}{m} \\ h(3,1) & h(3,2) & h(3,3) & h(3,4) \end{bmatrix} \times \begin{bmatrix} \mu_{fl} \\ \mu_{fr} \\ \mu_{rl} \\ \mu_{rr} \end{bmatrix} + v(t) \end{aligned} \tag{50}$$

where

$$h(1,1) = \frac{F^0{}_{xfl} \times \cos\delta_l - F^0{}_{yfl} \times \sin\delta_l}{m}$$

$$h(1,2) = \frac{F^0{}_{xfr} \times \cos\delta_r - F^0{}_{yfr} \times \sin\delta_r}{m}$$

$$h(2,1) = \frac{F^0{}_{xfl} \times \sin\delta_l + F^0{}_{yfl} \times \cos\delta_l}{m}$$

$$h(2,2) = \frac{F^0{}_{xfr} \times \sin\delta_r + F^0{}_{yfr} \times \cos\delta_r}{m}$$

$$h(3,1) = \left(l_a\left(F^0{}_{xfl} \times \sin\delta_l + F^0{}_{yfl} \times \cos\delta_l\right) + \frac{T_f}{2}\left(F^0{}_{xfl} \times \cos\delta_l - F^0{}_{yfl} \times \sin\delta_l\right)\right)/I_z$$

$$h(3,2) = \left(l_a\left(F^0{}_{xfr} \times \sin\delta_r + F^0{}_{yfr} \times \cos\delta_r\right) + \frac{T_f}{2}\left(F^0{}_{xfr} \times \cos\delta_r - F^0{}_{yfr} \times \sin\delta_r\right)\right)/I_z$$

$$h(3,3) = \frac{-\frac{T_r}{2} \times F^0{}_{xrl} - l_b \times F^0{}_{yrl}}{I_z}$$

$$h(3,4) = \frac{\frac{T_r}{2} \times F^0{}_{xrr} - l_b \times F^0{}_{yrr}}{I_z}$$

where R is the mean square error, $v(t)$ is assumed as the process noise, F^0_{xij} and F^0_{yij} are the longitudinal and lateral normalized forces of tires, respectively, δ_l and δ_r are the angles of left front and right front wheels, respectively.

Assume that all tires are identical and their adhesion coefficients are uniform in this paper. In order to improve the calculation speed, the above state and measurement equations are modified as follows:

$$x(t) = f(x(t), u(t), w(t)) = 1 \times \mu + w(t) \tag{51}$$

$$y(t) = h(x(t), v(t)) = \begin{bmatrix} h_1 \\ h_2 \\ h(3,1) + h(3,2) + h(3,3) + h(3,4) \end{bmatrix} \mu + v(t) \tag{52}$$

$$h_1 = \frac{F^0{}_{xfl} \times \cos\delta_l - F^0{}_{yfl} \times \sin\delta_l}{m} + \frac{F^0{}_{xfr} \times \cos\delta_r - F^0{}_{yfr} \times \sin\delta_r}{m} + \frac{F^0{}_{xrl}}{m} + \frac{F^0{}_{xrr}}{m} \tag{53}$$

$$h_2 = \frac{F^0{}_{xfl} \times \sin\delta_l + F^0{}_{yfl} \times \cos\delta_l}{m} + \frac{F^0{}_{xfr} \times \sin\delta_r + F^0{}_{yfr} \times \cos\delta_r}{m} + \frac{F^0{}_{yrl}}{m} + \frac{F^0{}_{yrr}}{m} \quad (54)$$

where μ is the road adhesion coefficient, and other symbols are defined in the same way as the above formulas.

3.2. Extended Kalman Filter

The detailed iterative process of the extended Kalman filter is as follows: Let $\hat{x}_{k-1}$ be the state estimation at time $k-1$. Set P_{k-1} as the covariance at time $k-1$. Linearize the equation of state by Taylor expansion at $\hat{x}_{k-1}$ and obtain [43–46]:

$$x_k = f(\hat{x}_{k-1}, u_{k-1}, w_{k-1}) + F_{k-1}(x_{k-1} - \hat{x}_{k-1}) + w_{k-1} \quad (55)$$

where $F_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1}, u_{k-1}}$. Let $\widetilde{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0)$. Similarly, linearize the measurement equation and obtain:

$$z_k = h(\widetilde{x}_k, v_k) + H_k(x_k - \widetilde{x}_k) + v_k \quad (56)$$

where $H_k = \left.\frac{\partial h}{\partial x}\right|_{\widetilde{x}_k}$. The predicted prior state is

$$\hat{x}_k^- = E[f(\hat{x}_{k-1}, u_{k-1}, w_{k-1}) + F_{k-1}(x_{k-1} - \hat{x}_{k-1}) + w_{k-1}] \quad (57)$$

Since w_{k-1} is the process noise satisfying Gaussian distribution with zero mean value and the estimated state is assumed to be the true value, the following equation can be obtained

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}, 0) \quad (58)$$

The prior covariance of the state estimation is

$$\begin{aligned} P_k^- &= E\left[\left(x_k - \hat{x}_k^-\right)\left(x_k - \hat{x}_k^-\right)^T\right] \\ &= E\left\{[F_{k-1}(x_{k-1} - \hat{x}_{k-1}) + w_{k-1}][(F_{k-1}(x_{k-1} - \hat{x}_{k-1}) + w_{k-1})]^T\right\} \\ &= F_{k-1}P_{k-1}{F_{k-1}}^T + Q \end{aligned} \quad (59)$$

Similarly, the predicted values and the covariance matrix for measurements are respectively as following:

$$\begin{cases} \hat{z}_k = h(\widetilde{x}_k, 0) \\ P_{zz,k} = H_k P_k^- H_k^T + R \end{cases} \quad (60)$$

The cross-covariance matrix between the state and the measurement is

$$\begin{aligned} P_{xz,k} &= E\left[\left(x_k - \hat{x}_k^-\right)\left(z_k - \hat{z}_k\right)^T\right] \\ &= E\left\{\left(x_k - \hat{x}_k^-\right)\left[\left(H_k\left(x_k - \hat{x}_k^-\right) + v_k\right)\right]^T\right\} \\ &= P_k^- H_k^T \end{aligned} \quad (61)$$

The gain matrix of the state is

$$K_k = \frac{P_{xz,k}}{P_{zz,k}} = \frac{P_k^- H_k^T}{H_k P_k^- H_k^T + R} \quad (62)$$

Then the estimation of the state at time k is

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - \hat{z}_k) \quad (63)$$

The covariance matrix of the state estimation is

$$\begin{aligned} P_k^- &= E\left[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T\right] \\ &= E\left\{\left[x_k - \hat{x}_k^- - K_k(z_k - \hat{z}_k)\right]\left[x_k - \hat{x}_k^- - K_k(z_k - \hat{z}_k)\right]^T\right\} \\ &= (I - K_k H_k)P_k^-(I - K_k H_k)^T + K_k R K_k^T \\ &= (I - K_k H_k)P_k^- \end{aligned} \tag{64}$$

The flow chart of the extended Kalman algorithm is shown in Figure 4. It mainly includes two modules, of which one is the update module and the other is the prediction module. The filter integrates the received sensor signals and first updates the time to achieve the prior estimation of state. After that, update the measurements to gain the posterior estimate according to the measurements. The equation modification is done by updating the equations of the state and measurement, which completes one iteration. With continuous iteration over time, the parameters are constantly modified to complete the precise estimation of adhesion coefficients.

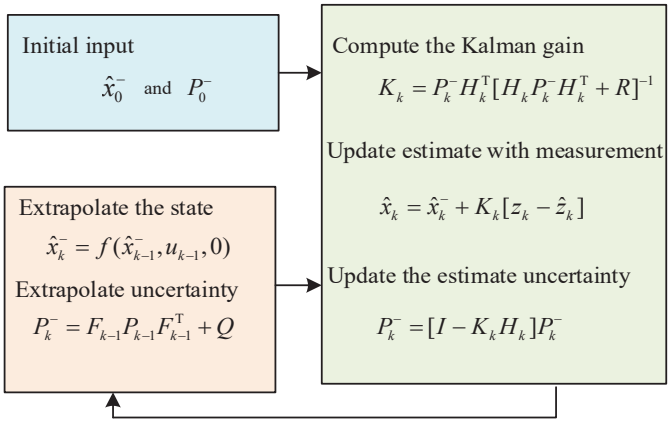


Figure 4. Algorithm flow chart of extended Kalman filter.

3.3. Identification Principle of Road Adhesion Coefficients

According to the study about the dynamics model of vehicles, the mathematical expression between the response signal of vehicles and the adhesion coefficients is established. According to Figure 5, the necessary data for the identification of adhesion coefficients are collected by vehicle sensors. For example, the wheel angle sensor is used to capture the angles of four wheels. The tachometer sensor is used to gather the rotational speed of four wheels. Then the collected data is sent through the vehicle communication network. The controller makes full use of the collected data to identify the adhesion coefficient and then conducts the corresponding algorithm to drive the vehicles.

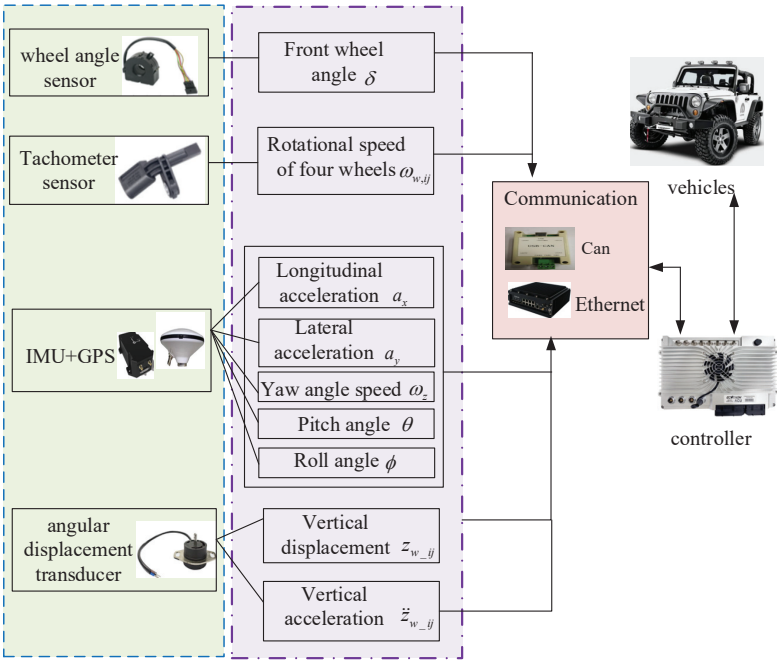


Figure 5. Flow chart of data acquisition.

The overall process of the adhesion coefficient estimation methods is given, just like in Figure 6. The identification algorithm for the road adhesion coefficient is designed by the mathematical relationship in Figure 6. Firstly, the signal gathered by the vehicle sensors is processed. Secondly, the slip rate and the side deflection angle are calculated by the wheel speed, the longitudinal and transverse speed, the front wheel angle, and other parameters. The signals of the wheel vertical displacement, the vertical acceleration, the pitch of vehicles, and the roll angle are transmitted to the vertical load calculation module for processing. Thirdly, the vertical loads, the output slip rate, and the side deflection angles in each parameter calculation module are used as input parameters to calculate the normalized force through the Dugoff tire model. Finally, the normalized force, the vehicle acceleration signal, and the wheel angle are transferred to the extended Kalman filter. The prior state estimation is obtained through the prediction part of the filter. And the measurement is updated by the respective measured values for correction, and the identification result of the adhesion coefficient of roads is updated and corrected through iterations.

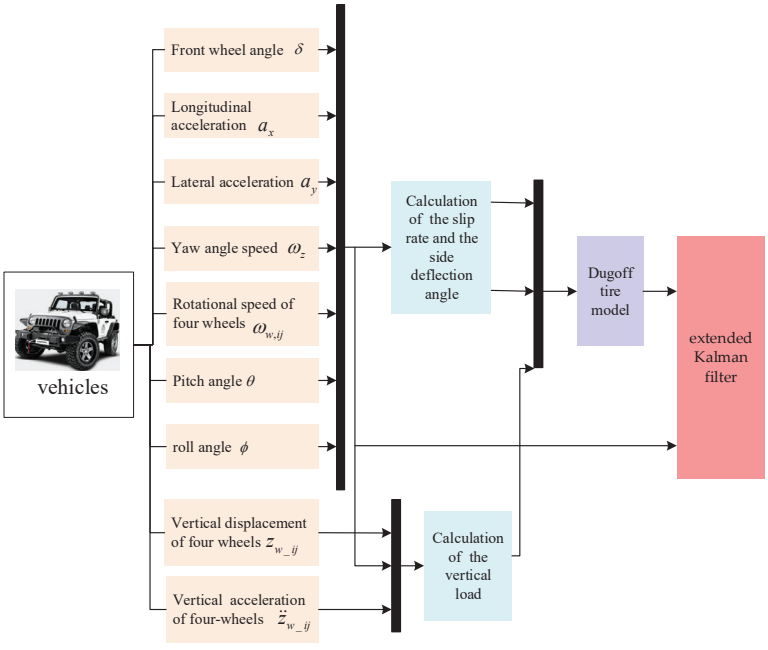


Figure 6. Flow chart of adhesion coefficient identification algorithms.

4. Simulation Results and Analysis

CarSim is a special simulation software in the vehicle field that can simulate the vehicle's response to driver, road surface, and aerodynamic input. And Carsim is widely utilized in modern automobile control systems. Based on the recognition and wide application of Carsim in the industry, this paper uses Carsim to build a simulation environment to simulate the vehicle response under different road surfaces and scenarios with different control inputs and enters vehicle response signals into the algorithm module built in Matlab/Simulink as input signals. Through the joint simulation, the identification effects of the traditional seven-DOF vehicle dynamics model and the improved algorithm proposed in this paper are compared under different road surfaces and driving scenarios.

4.1. Simulation Scenario and Parameter Setting

When designing the algorithm, the applicability of the algorithm under slopes and uneven road surfaces is optimized. In order to ensure its effectiveness, the vehicle model and wheel vertical load model are first verified. Finally, the experiments of pavement adhesion coefficient estimation under different working conditions are carried out. The sensor acquisition principle of this project is shown in Figure 5. Due to the limitations of laboratory equipment, this project is only verified by simulation experiments in CarSim. For the above three scenarios, the simulation parameters are described in the following two sections.

4.1.1. Simulation Parameters in the Vertical Load Model and the Dynamics Model

CarSim can directly output the actual tire force. In the co-simulation environment, the angle of front wheels, the longitudinal force of tires, and the transverse force of tires are input into the dynamic equation of vehicles in the directions of the longitudinal, transverse, and yaw, respectively, to solve their accelerations. In order to verify the applicability of the established vehicle dynamics model and the vertical load model under the slope road, a simulation environment is built in CarSim, and the slope is set at 0.2 m/1 m. Experimental parameters are assumed, just like in Table 1. In addition, the road surface is supposed

to have certain unevenness (see Figure 7) and slope to simulate the characteristics of a non-structural road surface.

Table 1. Experimental parameter setting.

Target Speed/(km/h)	Brake Control/ MPa	Open Loop Control/Deg	Pavement Adhesion Coefficient μ	Slope
100	10	0	0.3	0.1

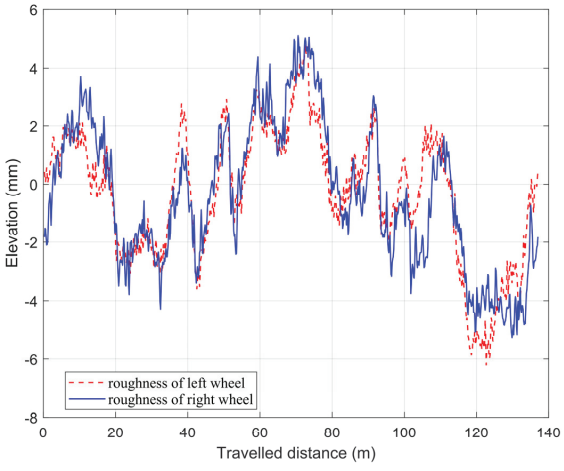


Figure 7. Road roughness.

4.1.2. Simulation Parameters in the Experimental Verification of the Adhesion Coefficient Identification

For the sake of verifying the effectiveness of the developed algorithm, the corresponding models are constructed in the MATLAB/Simulink environment. The vehicle motion parameters are set by CarSim, the actual vehicle driving scenario is simulated, and the required parameters are estimated by the extended Kalman filter. The process of experimentation for the identification of specific pavement adhesion coefficients is presented in Figure 8. The vehicle model, the driving environment, and the control input are set by CarSim to satisfy the simulation requirements of different environments and different working conditions, and then the vehicle response signal is input into MATLAB to identify and compare with our method and traditional methods (VDMSDF). It should be noted that the simulation environment of CarSim can be tested with real cars if conditions permit. In order to verify the effectiveness of the adhesion coefficient identification algorithm, the scenarios with and without steering and braking are analyzed and verified, respectively.

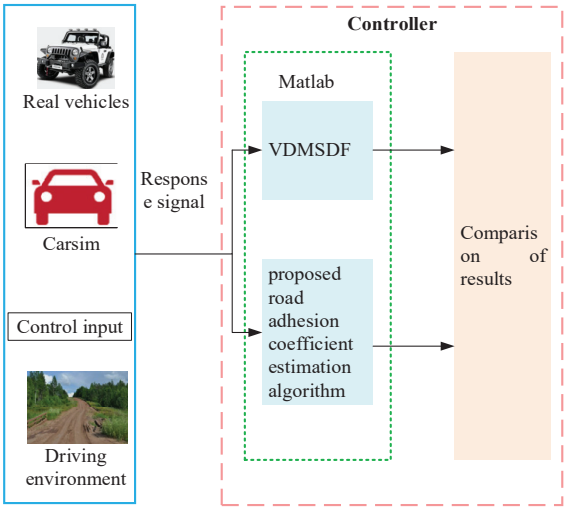


Figure 8. Experimental flow chart of adhesion coefficient identification for pavements.

Case 1: No steering and braking scenario

According to various working conditions, CarSim is used to set different adhesion coefficients, select the same vehicle model for simulation, and output the corresponding vehicle response signals. The algorithm model is built in the MATLAB/Simulink environment to receive vehicle responses and compare the true road adhesion coefficient with the identification value. The project designs the following three sets of simulation experiments to test the identification results of high, medium, and low adhesion coefficients, respectively.

The method proposed in this paper is primarily applied to the autonomous driving of vehicles in complex off-road, high-speed scenarios. Therefore, the initial speeds of three simulation experiments are respectively set at 100 km/h to meet the requirements of high-speed driving, and the road surface is set to have certain unevenness (see Figure 7) and slope to simulate the characteristics of non-structural road surfaces, and the specific parameter is displayed in Table 2.

Table 2. Experimental parameter setting.

Target Speed/(km/h)	Brake Control/MPa	Open Loop Control/Deg	Pavement Adhesion Coefficient μ	Slope
100	10	0	0.7	0.1
100	10	0	0.5	0.1
100	10	0	0.3	0.1

Case 2: Steering and braking scenario

For the sake of testing the estimation effect of the method in different scenarios, a vehicle steering braking scenario is built in CarSim for experiments. Also, three sets of simulation experiments in Table 3 are designed to test the estimation results of different adhesion coefficients in the turning scene, respectively. The road surface is set to have certain unevenness (see Figure 7). So the slope is set at 0.2 m/1 m to aim at proving the validity of the algorithm under different slope conditions.

Table 3. Parameter setting table of steering and braking.

Target Speed	Brake Control	Open Loop Control	Adhesion Coefficient μ
100 km/h	10 MPa	45 deg	0.7
100 km/h	10 MPa	45 deg	0.5
100 km/h	10 MPa	45 deg	0.3

Case 3: Variable adhesion coefficient scenario

The road surface drive on by vehicles usually covers a variety of types. Due to different ground materials and other factors, the adhesion force received by the vehicle during the driving process can also change accordingly. Therefore, in addition to the two above scenes, this paper also considers a scene with a varying adhesion coefficient to verify the effectiveness of the algorithm. The true adhesion coefficient is assumed to be 0.3 for 0–1.7 s and 0.6 for 1.7 s to 4 s. Other parameter settings are the same as those in Case 1 and Case 2.

4.2. Experimental Results and Analysis about the Dynamics Model of Vehicles

The longitudinal, transverse, and yaw accelerations of CarSim are outputs for comparison. The correctness of the proposed vehicle dynamics model can be tested by comparison. The experimental curves are shown in Figures 9–11.

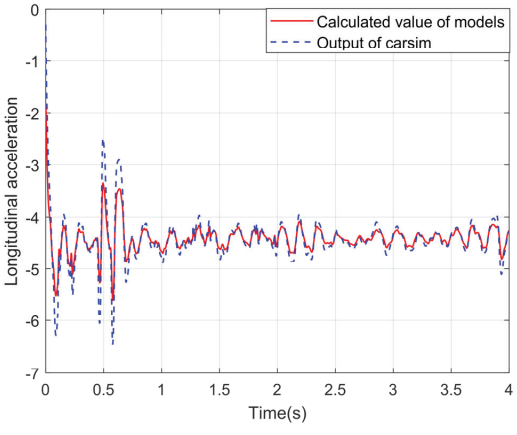


Figure 9. Longitudinal acceleration under the slope road.

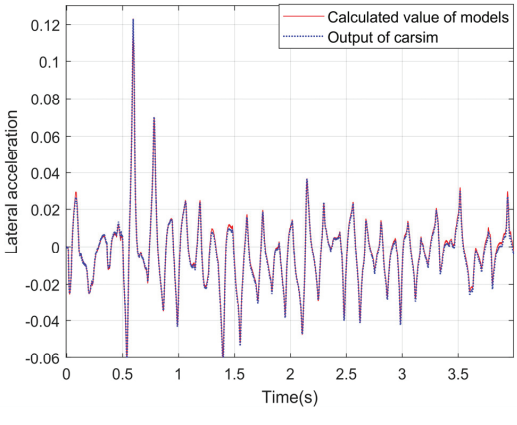


Figure 10. Lateral acceleration under the slope road.

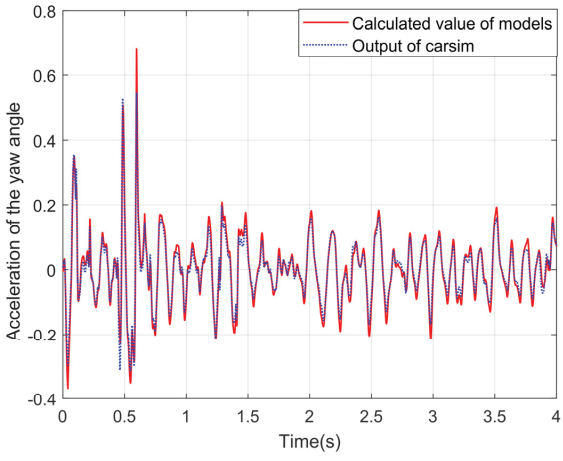


Figure 11. Acceleration of yaw angles under the slope roads.

It can be seen from the above experimental curves that the calculated values of the lateral and longitudinal acceleration, together with the acceleration of the yaw angle, can maintain a good consistency with the real values in the changing trend, which indicates that the established vehicle model can better reflect the motion status of vehicles under the simulation condition of the slope road. Since the equivalent suspension model is introduced in this paper to optimize the calculation of vertical wheel load and to correct vehicle acceleration combined with vehicle pose data, the proposed algorithm in this paper improves the accuracy of the vehicle dynamics model under uneven and sloping road surfaces.

4.3. Experimental Results and Analysis about the Vertical Load Model

A simulation environment is built in CarSim, and the vehicle response is input into the vertical load calculation module. The correctness of the model is verified by comparing the output results of the vertical load model with the output values of the vertical load in CarSim. Under the conditions of slope and uneven road, the simulation experiment is carried out, and the experimental curves are shown in Figures 12–15.

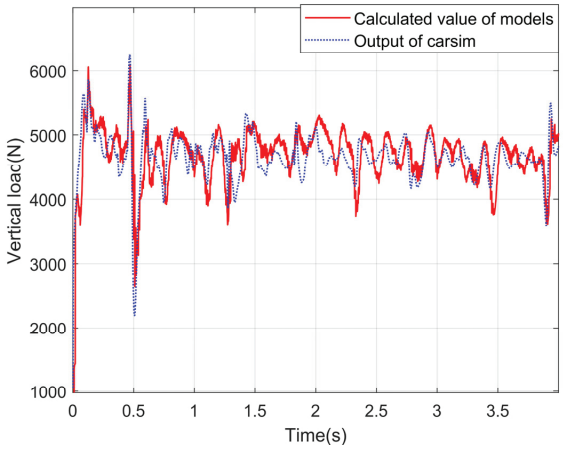


Figure 12. Vertical loads for left front wheels under the slope road.

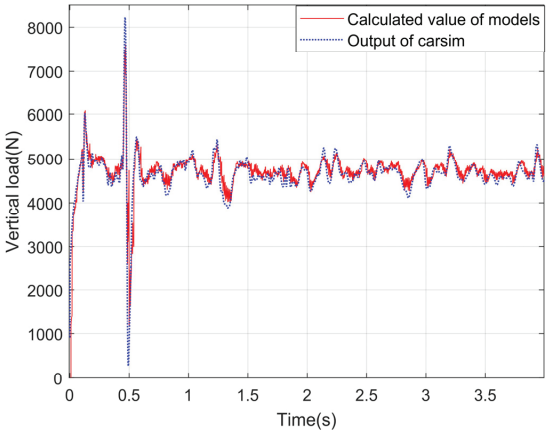


Figure 13. Vertical load of the right front wheel under the slope road.

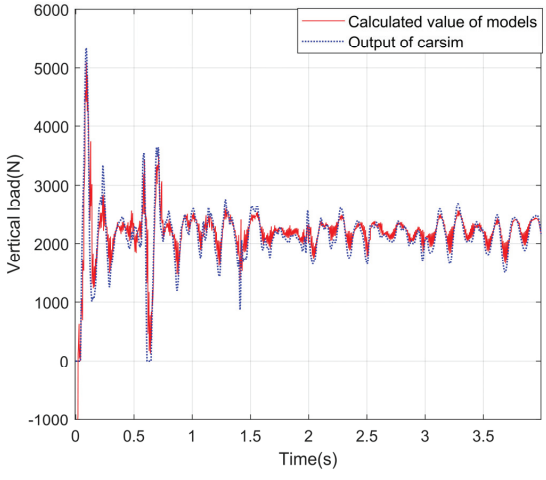


Figure 14. Vertical load of the left back wheel under the slope road.

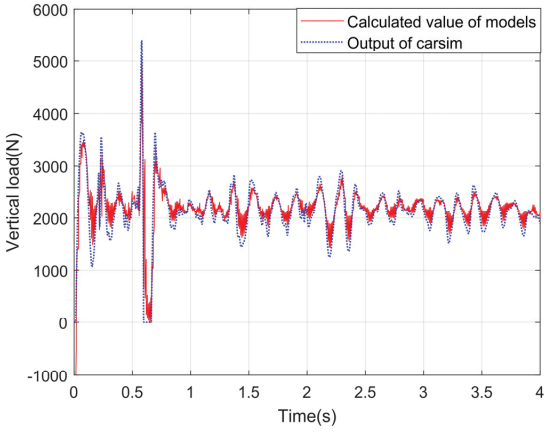


Figure 15. Vertical load of the right back wheel under the slope road.

It can be seen from the observation of the experimental curves that although some errors exist between the calculation results for the vertical load of four wheels and the corresponding CarSim outputs at the peak of the fluctuation, the changing trends are basically the same. It indicates that the model can calculate the vertical loads of the wheels more accurately on a slope and uneven road. In a word, the reason why the algorithm proposed in this paper has high precision is that the equivalent suspension model is introduced in this paper, and the load is calculated by the vertical response signals such as velocity and acceleration.

4.4. Experimental Results and Analysis about the Adhesion Coefficient Identification

For the two simulation scenarios described in Section 4.1.2, this section will analyze the corresponding simulation results.

4.4.1. No Steering and Braking Scenario

When the adhesion coefficient of roads is set to 0.7, the simulation and road adhesion coefficient estimation are conducted in this part. The estimation results of the adhesion coefficients on the basis of the equivalent suspension model are compared with the conventional adhesion coefficient estimation results based on the VDMSDF. The experimental result is shown in Figure 16.

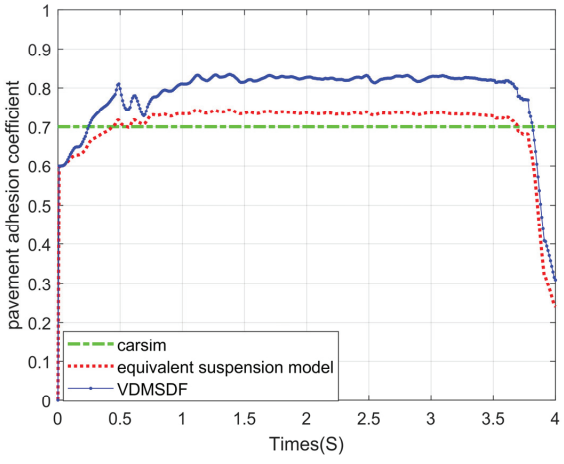


Figure 16. Estimated results of pavement adhesion coefficient 0.7.

The estimated curve of the pavement adhesion coefficient based on the equivalent suspension is shown in Figure 16. It can be seen from Figure 16 that the curve converges to about 0.74 at 1 s. And then a small fluctuation is maintained until the vehicle's braking is completed after 3 s. Based on the VDMSDF, the convergence speed of the curve in Figure 16 is slow and finally stabilizes at about 0.83. Considering that there is a large deviation between both methods, the proposed method in this paper has better estimation accuracy than the latter.

Set the adhesion coefficient of roads at 0.5. The estimated curve is shown in Figure 17.

As can be seen from the experimental curve under the slope road surface in Figure 17, the adhesion coefficient estimation based on the traditional seven-degree-of-freedom vehicle dynamics model fluctuates around 0.6 and cannot converge to 0.5. The adhesion coefficient identification on the basis of the equivalent suspension converges to about 0.5 in 1 s and then has less fluctuation.

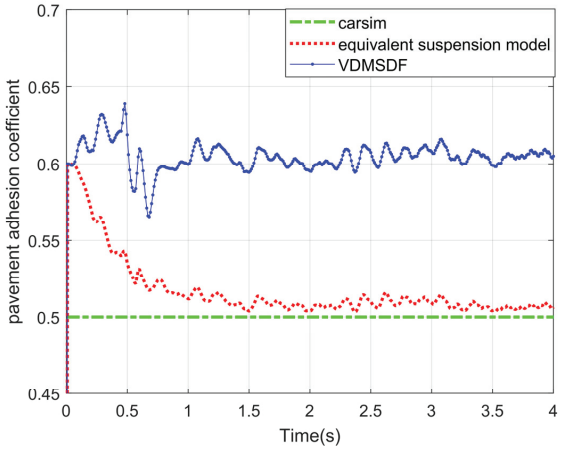


Figure 17. Estimated results of pavement adhesion coefficient 0.5.

Set the adhesion coefficient of roads to 0.3, and the corresponding estimated result is shown in Figure 18.

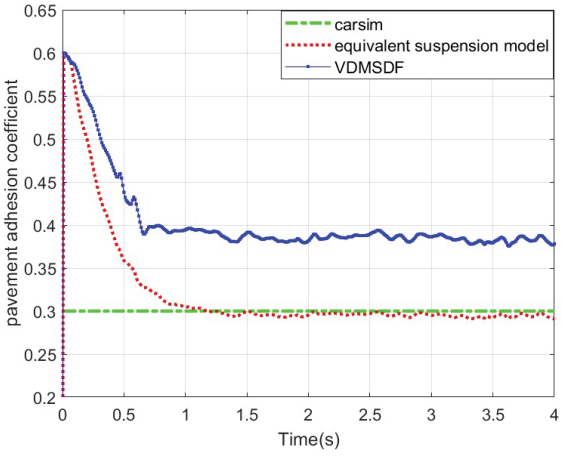


Figure 18. Estimated results of pavement adhesion coefficient 0.3.

The experimental curve in Figure 18 suggests that the corresponding estimated result based on the equivalent suspension rapidly converges at 0~1 s and then remains stable at around 0.29. Based on the VDMSDF, the estimated result curve is stable around 0.39.

Generally speaking, although a certain error exists between the estimation results based on the equivalent suspension and the adhesion coefficients set by CarSim, the error is small, and the accuracy is higher than that based on the VDMSDF. The root-mean-square errors (RMSEs) of the estimation results of two algorithms are displayed in Table 4.

Table 4. RMSEs of experimental estimation results.

Pavement Adhesion Coefficient μ	RMSEs of Equivalent Suspension Model	RMSEs of VDMSDF
0.7	0.05608594	0.11667209
0.5	0.03688530	0.10776193
0.3	0.069288424	0.115742819

According to the RMSEs of the estimation results in Table 4, the equivalent suspension model-based estimation result has a smaller error under various road adhesion coefficients, and the estimation accuracy is improved by at least 3.6%. Compared with the estimation based on the VDMSDF, the estimation accuracy is higher on the non-structural road surface with its slope and uneven road surface characteristics.

4.4.2. Steering and Braking Scenario

The results of the road adhesion coefficient estimation based on the equivalent suspension model are also compared with those based on the seven-DOF vehicle dynamics model in Figure 19.

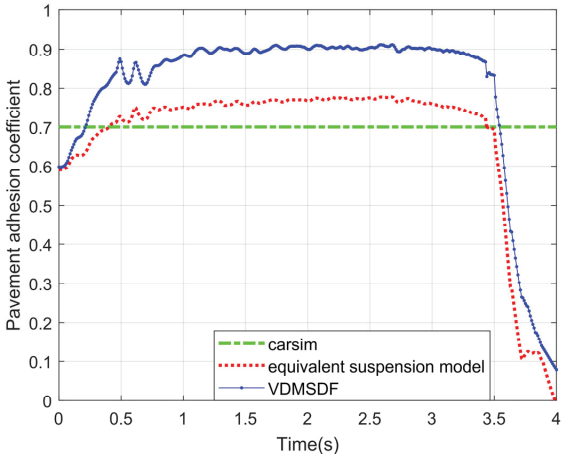


Figure 19. Estimated results of pavement adhesion coefficient 0.7.

The experimental curve in Figure 19 indicates that the estimated result curve based on the equivalent suspension model converges to about 0.76 at 1 s and then maintains a small fluctuation until the vehicle braking is completed after 3 s. Based on the VDMSDF, the curve convergence speed is slow, and finally stabilizes at about 0.9, with a large deviation.

The adhesion coefficient of roads is set to 0.5, and estimating results under steering braking conditions are shown in Figure 20.

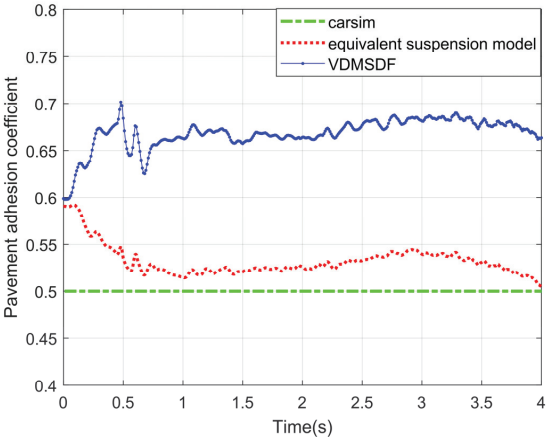


Figure 20. Estimated results of pavement adhesion coefficient 0.5.

The experimental curve in Figure 20 shows that the estimated result curve based on the equivalent suspension model remains stable at around 0.51. Based on VDMSDF, the estimated result curve is stable around 0.68.

The adhesion coefficient of roads is set to 0.3, and the estimated result curve under the steering braking condition is shown in Figure 21.

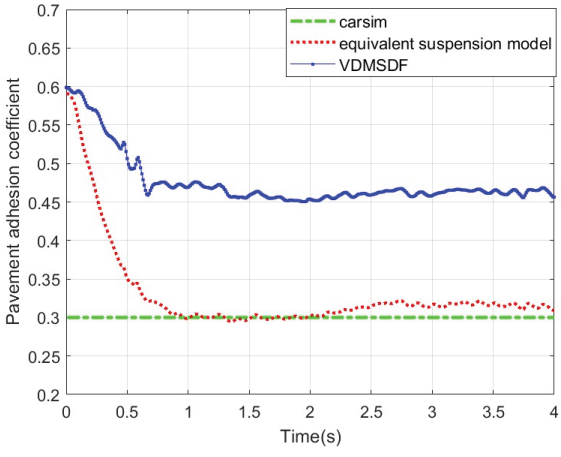


Figure 21. Estimated results of pavement adhesion coefficient 0.3.

As shown in Figure 21, the experimental results of the equivalent suspension model and the seven-DOF vehicle dynamics model remain stable at around 0.3 and 0.45, respectively.

The root-mean-square errors (RMSEs) of the two estimation methods are calculated, and the results are shown in Table 5.

Table 5. RMSEs of experimental estimation results.

Pavement Adhesion Coefficient μ	RMSEs of Equivalent Suspension Model	RMSEs of VDMSDF
0.7	0.06281493	0.17942874
0.5	0.03788459	0.16536712
0.3	0.07688491	0.18405923

According to the RMSEs of the estimation results in Table 5, the estimation results on the basis of the equivalent suspension model have a smaller error under various road adhesion coefficients, and the estimation accuracy is improved by at least 3.7%. In a word, compared with the estimation based on the VDMSDF, the estimation accuracy is higher on the non-structural road surface under the steering braking conditions.

4.4.3. Variable Adhesion Coefficient Scenario

In this scenario, the designed identification algorithm on the basis of the traditional seven-DOF vehicle dynamics model together with the proposed method in the paper is applied to identify and compare the adhesion coefficients. The experimental curves are displayed in the figure.

The experimental curve in Figure 22 suggests that the proposed algorithm can identify the sudden change of adhesion coefficients with high accuracy, and the sudden change in pavement adhesion coefficients can be identified within 1.5 s.

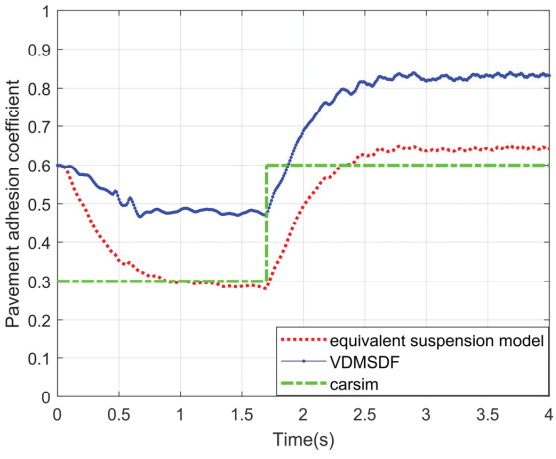


Figure 22. Estimated results of variable adhesion coefficient scenario.

In a word, the identification method by virtue of the equivalent suspension model has higher accuracy on the road surface with slope and road roughness characteristics, and the method has strong applicability and can be used under the straight line, the steering conditions, and the different slope conditions. The reason why the identification error of the contrast method is large is that the traditional dynamics models with seven-degrees of freedom ignore the vertical response of the vehicle; the vertical load of the wheels in the model is very different from the actual value while driving on uneven road surfaces. By introducing the equivalent suspension model, the algorithm proposed in this paper reduces the deviation of the wheel vertical load and improves identification accuracy.

5. Conclusions

In the study, an estimation method of adhesion coefficients on unstructured pavement by virtue of the extended Kalman filter is put forward in this paper, which can better identify the adhesion coefficient under the road with the uneven and large slope. The identification accuracy of road adhesion coefficients under unstructured pavement is improved by introducing the equivalent suspension model to optimize the calculation of vertical wheel load and modifying vehicle acceleration combined with vehicle posture data. And the multi-condition simulation experiments with CarSim prove that the proposed identification algorithm for adhesion coefficients has a higher estimation accuracy that has improved by at least 3.6%. In a word, the designed method in the paper is efficient. The identification of adhesion coefficients in the designed method is mainly applied to automatic driving scenarios. For example, the ground adhesion coefficient can be obtained and shared through the excitation response data from the driving vehicles in front, which is convenient for the path planning and stability control of subsequent vehicles.

Author Contributions: For research articles with five authors. Conceptualization, H.Z., J.X. and B.S.; software, H.Z. and J.X.; investigation, H.Z. and J.X.; writing, J.X., H.L. and S.L.; supervision, B.S., S.L. and H.L.; funding acquisition, H.L. and B.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key R&D Program of China (2023YFB4700292), Guangdong Basic and Applied Basic Research Foundation (2022A1515110880, 2023A1515011650) and Open project of Hubei Key Laboratory of Intelligent Robot (Wuhan Institute of Technology) under Grant HBIR 202309.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: All authors in this paper declares no conflicts of interest, and decides to publish the results must be declared in this section.

References

1. Qi, G.; Fan, X.; Li, H. A comparative study of the recursive least squares and fuzzy logic estimation methods for the measurement of road adhesion coefficient. *Aust. J. Mech. Eng.* **2023**, *21*, 1230–1246. [CrossRef]
2. Wang, S.F.; Liang, Q.W.; Liu, Z.; Zhang, J.Y. Research on collision avoidance control of intelligent vehicles based on mstukf road adhesion coefficient identification. *Adv. Transp. Stud.* **2022**, *58*, 245–260.
3. Novikov, I.; Lazarev, D. Experimental installation for calculation of road adhesion coefficient of locked car wheel. *Transp. Res. Procedia* **2017**, *20*, 463–467. [CrossRef]
4. Xue, W.; Zheng, L. Active collision avoidance system design based on model predictive control with varying sampling time. *Automot. Innov.* **2020**, *3*, 62–72. [CrossRef]
5. Wang, C.; Wang, Z.; Zhang, Z.; Liu, J.; Li, W.; Wu, Y.; Li, X.; Yu, H.; Cao, D. Integrated post-impact planning and active safety control for autonomous vehicles. *IEEE Trans. Intell. Veh.* **2023**, *8*, 2062–2076. [CrossRef]
6. Leng, B.; Jin, D.; Hou, X.; Tian, C.; Xiong, L.; Yu, Z. Tire-road peak adhesion coefficient estimation method based on fusion of vehicle dynamics and Machine Vision. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 21740–21752. [CrossRef]
7. Huang, K.; Jiang, C.; Qiu, M.-M.; Wu, D.; Zhang, B.-Z. Research on intelligent vehicle lane changing and obstacle avoidance control based on road adhesion coefficient. *J. Vib. Control* **2021**, *28*, 3269–3285. [CrossRef]
8. Leng, B.; Tian, C.; Hou, X.; Xiong, L.; Zhao, W.; Yu, Z. Tire-road peak adhesion coefficient estimation based on multisource information assessment. *IEEE Trans. Intell. Veh.* **2023**, *8*, 3854–3870. [CrossRef]
9. Qi, G.; Fan, X.; Li, H. A comparative study of the unscented Kalman filter and particle filter estimation methods for the measurement of the road adhesion coefficient. *Mech. Sci.* **2022**, *13*, 735–749. [CrossRef]
10. Liu, R.; Wei, M.; Sang, N.; Wei, J. Research on curved path tracking control for four-wheel steering vehicle considering road adhesion coefficient. *Math. Probl. Eng.* **2020**, *2020*, 3108589. [CrossRef]
11. Chen, Y.; Li, X.; Qu, T. Research on road adhesion coefficient estimation for electric vehicle under tire cornering condition. *Appl. Mech. Mater.* **2013**, *427*, 275–279. [CrossRef]
12. Li, C.; Liu, P.; Xie, Z.; Li, Z.; Huan, H. Road adhesion coefficient estimation based on vehicle-road coordination and deep Learning. *J. Adv. Transp.* **2023**, *2023*, 3633058. [CrossRef]
13. Wang, Q.; Xu, J.; Sun, T.; Lv, Z.; Zong, G. Research on road adhesion condition identification based on an improved ALexNet model. *J. Adv. Transp.* **2021**, *2021*, 1–14. [CrossRef]
14. Leng, B.; Jin, D.; Xiong, L.; Yang, X.; Yu, Z. Estimation of tire road peak adhesion coefficient for intelligent electric vehicles based on camera and tire dynamics information fusion. *Mech. Syst. Sig. Process.* **2021**, *150*, 107275. [CrossRef]
15. Guo, H.; Zhao, X.; Liu, J.; Dai, Q.; Liu, H.; Chen, H. A fusion estimation of the peak tire–road friction coefficient based on road images and dynamic information. *Mech. Syst. Signal Process.* **2023**, *189*, 110029. [CrossRef]
16. Herrmann, T.; Wischnewski, A.; Hermansdorfer, L.; Betz, J.; Lienkamp, M. Real-time adaptive velocity optimization for autonomous electric cars at the limits of handling. *IEEE Trans. Intell. Veh.* **2021**, *6*, 665–677. [CrossRef]
17. Guo, C.; Wang, X.; Su, L.; Wang, Y. Safety distance model for longitudinal collision avoidance of logistics vehicles considering slope and road adhesion coefficient. *Proc. Inst. Mech. Eng. Part D J. Automob. Eng.* **2021**, *235*, 498–512. [CrossRef]
18. Ma, B.; Lv, C.; Liu, Y.; Zheng, M.; Yang, Y.; Ji, X. Estimation of road adhesion coefficient based on tire aligning torque distribution. *J. Dyn. Syst. Meas. Contr.* **2018**, *140*, 051010. [CrossRef]
19. Chen, L.; Luo, Y.; Bian, M.; Qin, Z.; Luo, J.; Li, K. Estimation of tire-road friction coefficient based on frequency domain data fusion. *Mech. Syst. Sig. Process.* **2017**, *85*, 177–192. [CrossRef]
20. Mooryong, C.; Jiwon, J.O.; Seibum, B.C. Linearized recursive least squares methods for real-time identification of tire-road friction coefficient. *IEEE Trans. Veh. Technol.* **2013**, *62*, 2906–2918. [CrossRef]
21. Kikuuwe, R. A brush-type tire model with non-smooth representation. *Math. Probl. Eng.* **2019**, *2019*, 9747605. [CrossRef]
22. Shao, L.; Jin, C.; Lex, C.; Eichberger, A. Robust road friction estimation during vehicle steering. *Veh. Syst. Dyn.* **2019**, *57*, 493–519. [CrossRef]
23. Singh, K.; Sivaramakrishnan, S. Extended pacejka tire model for enhanced vehicle stability control. *arXiv* **2023**, *2305*, 18422.
24. Vieira, D.; Orjuela, R.; Spisser, M.; Basset, M. An adapted Burckhardt tire model for off-road vehicle applications. *J. Terramech.* **2022**, *104*, 15–24. [CrossRef]
25. Tian, C.; Leng, B.; Hou, X.; Xiong, L.; Huang, C. Multi-sensor fusion based estimation of tire-road peak adhesion coefficient considering model uncertainty. *Remote Sens.* **2022**, *14*, 5583. [CrossRef]
26. Šabanovič, E.; Žuraulis, V.; Prentkovskis, O.; Skrickij, V. Identification of road-surface type using deep neural networks for friction coefficient estimation. *Sensors* **2020**, *20*, 612–629. [CrossRef]
27. Feng, J.; Zhao, F.; Ye, M.; Sun, W. The auxiliary system of cleaning vehicle based on road recognition technology. *SAE Tech. Pap.* **2021**, *1*, 0245.

28. Lian, Y.; Feng, W.; Liu, S.; Nie, Z. A road adhesion coefficient-tire cornering stiffness normalization method combining a fractional-order multi-variable gray model with a LSTM network and vehicle direct yaw-moment robust control. *Front. Neurorobot.* **2023**, *17*, 1229808. [CrossRef]
29. Wu, Y.; Li, G.; Fan, D. Joint estimation of driving state and road adhesion coefficient for distributed drive electric vehicle. *IEEE Access* **2021**, *9*, 75460–75469. [CrossRef]
30. Wang, J.; Jang, Y.; Yu, J.; Wu, M.; Li, N. Road adhesion coefficient estimation based on second-order linear extended state observer. *Math. Probl. Eng.* **2023**, *2023*, 5953102. [CrossRef]
31. Zhang, C.; Sun, J.; He, J.; Liu, L. Online estimation of the adhesion coefficient and its derivative based on the cascading SMC Observer. *J. Sens.* **2017**, *2017*, 8419295. [CrossRef]
32. Lu, X.; Shi, Q.; Li, Y.; Xu, K.; Tan, G. Road adhesion coefficient identification method based on vehicle dynamics model and multi-algorithm fusion. *SAE Tech. Pap.* **2022**, *1*, 0908. [CrossRef]
33. Krakhmalev, O.; Krakhmalev, N.; Gataullin, S.; Makarenko, I.; Nikitin, P.; Serdechnyy, D.; Liang, K.; Korchagin, S. Mathematics Model for 6-DOF Joints Manipulation Robots. *Mathematics* **2021**, *9*, 2828. [CrossRef]
34. Pugi, L.; Favilli, T.; Berzi, L.; Locorotondo, E.; Pierini, M. Brake blending and torque vectoring of road electric vehicles: A flexible approach based on smart torque allocation. *Int. J. Electr. Hybrid Veh.* **2020**, *12*, 87–115. [CrossRef]
35. Sun, Y.A.; Xiong, L.; Yu, Z.P.; Feng, Y.; Ren, L.J. Road adhesion coefficient estimation. *Appl. Mech. Mater.* **2011**, *52*, 1503–1508. [CrossRef]
36. Chen, Z.; Duan, Y.; Wu, J.; Zhang, Y. On-board estimation of road adhesion coefficient based on ANFIS and UKF. *SAE Tech. Pap.* **2022**, *2022*, 0297.
37. Li, G.; Fan, D.S.; Wang, Y.; Xie, R.C. Study on vehicle driving state and parameters estimation based on triple cubature Kalman filter. *Int. J. Heavy Veh. Syst.* **2020**, *27*, 126. [CrossRef]
38. Chen, X.; Li, S.; Li, L.; Zhao, W.; Cheng, S. Longitudinal lateral cooperative estimation algorithm for vehicle dynamics states based on adaptive-square-root-cubature-Kalman-filter and similarity-principle. *Mech. Syst. Sig. Process.* **2022**, *176*, 176–197. [CrossRef]
39. Ding, N.; Taheri, S. A modified Dugoff tire model for combined-slip forces. *Tire Sci. Technol.* **2010**, *38*, 228–244. [CrossRef]
40. Villano, E.; Lenzo, B.; Sakhnevych, A. Cross combined UKF for vehicle sideslip angle estimation with a modified Dugoff tire model: Design and experimental results. *Meccanica* **2021**, *56*, 2653–2668. [CrossRef]
41. Sun, X.; Zhang, H.; Cai, Y.; Wang, S.; Chen, L. Hybrid modeling and predictive control of intelligent vehicle longitudinal velocity considering nonlinear tire dynamics. *Nonlinear Dyn.* **2019**, *97*, 1–6. [CrossRef]
42. Jin, X.; Yin, G. Estimation of lateral tire–road forces and sideslip angle for electric vehicles using interacting multiple model filter approach. *J. Frankl. Inst.* **2015**, *352*, 686–707. [CrossRef]
43. Anggraeni, D.; Sudiarto, B.; Kurniawan, F.; Priambodo, P.S. Lithiumion battery modelling and adaptive extended Kalman filter implementation for BMS application software development. *Int. J. Renew. Energy Res.* **2023**, *13*, 418–427.
44. Salton, A.T.; Pimentel, G.A.; Melo, J.V.; Castro, R.S.; Benfica, J. Data-driven covariance tuning of the extended Kalman filter for visual-based pose estimation of the Stewart platform. *J. Control Autom. Electr. Syst.* **2023**, *34*, 720–730. [CrossRef]
45. Yang, B.; Li, G.; Tang, W.; Li, H. Research on optimized SOC estimation algorithm based on extended Kalman filter. *Front. Energy Res.* **2022**, *10*, 1027439. [CrossRef]
46. Nair, S.H.; Devika, M.K.; Raj, A.; Bernard, B.; Vinod, V.; Sunil Kumar, P.R. Improving distance relay performance for heavily distorted voltage and current by accurate signal reconstruction using extended Kalman filter. *Meas. Sci. Technol.* **2023**, *34*, 045014. [CrossRef]

sensors

MDPI

Article

NeoSLAM: Long-Term SLAM Using Computational Models of the Brain

Carlos Alexandre Pontes Pizzino [1,*], Ramon Romankevicius Costa [1], Daniel Mitchell [2] and Patrícia Amâncio Vargas [2]

[1] PEE/COPPE—Department of Electrical Engineering, Federal University of Rio de Janeiro, Cidade Universitária, Centro de Tecnologia, Bloco H, Rio de Janeiro 21941-972, RJ, Brazil; ramon@coep.ufrj.br
[2] Edinburgh Centre for Robotics, Heriot-Watt University, Edinburgh EH14 4AS, UK; d.mitchell.5@research.gla.ac.uk (D.M.); p.a.vargas@hw.ac.uk (P.A.V.)
* Correspondence: carlos.pizzino@coppe.ufrj.br

Abstract: Simultaneous Localization and Mapping (SLAM) is a fundamental problem in the field of robotics, enabling autonomous robots to navigate and create maps of unknown environments. Nevertheless, the SLAM methods that use cameras face problems in maintaining accurate localization over extended periods across various challenging conditions and scenarios. Following advances in neuroscience, we propose NeoSLAM, a novel long-term visual SLAM, which uses computational models of the brain to deal with this problem. Inspired by the human neocortex, NeoSLAM is based on a hierarchical temporal memory model that has the potential to identify temporal sequences of spatial patterns using sparse distributed representations. Being known to have a high representational capacity and high tolerance to noise, sparse distributed representations have several properties, enabling the development of a novel neuroscience-based loop-closure detector that allows for real-time performance, especially in resource-constrained robotic systems. The proposed method has been thoroughly evaluated in terms of environmental complexity by using a wheeled robot deployed in the field and demonstrated that the accuracy of loop-closure detection was improved compared with the traditional RatSLAM system.

Keywords: long-term visual SLAM; biologically inspired robots; neurorobotics; sparse distributed representation; hierarchical temporal memory

Citation: Pizzino, C.A.P.; Costa, R.R.; Mitchell, D.; Vargas, P.A. NeoSLAM: Long-Term SLAM Using Computational Models of the Brain. *Sensors* **2024**, *24*, 1143. https://doi.org/10.3390/s24041143

Academic Editors: Yuanlong Xie, Shiqi Zheng, Zhaozheng Hu and Shuting Wang

Received: 2 January 2024
Revised: 30 January 2024
Accepted: 2 February 2024
Published: 9 February 2024

1. Introduction

A Simultaneous Localization and Mapping (SLAM) method allows a robot to continuously create a map of the environment and at the same time estimate its location based on this map [1]. Since robots can be applied sector-wide, it is important that robots can safely and accurately localize themselves with the environment to ensure efficient operation in challenging sectors, such as the Offshore Renewable Energy (ORE) sector [2–4], Nuclear sector [5,6], and Medical sector [7].

One of the most important aspects of a SLAM system is place recognition or loop-closure detection (LCD). This aims to reduce the robot pose uncertainty due to the effect of the errors introduced by the odometry. Moreover, the incorrect data associations in the LCD can result in a critical failure for SLAM algorithms [8].

As the usage of vision sensors has increased rapidly, the visual place recognition (VPR) task has been widely studied in recent years. Cameras are cheaper than laser scanners and provide increasing amounts of information. By definition, a VPR system must be able to recognize a previously visited place via visual information [9]. In other words, an autonomous robot that operates in an environment should be able to recognize different places when it revisits them after some time as in long-term robot operation (Figure 1).

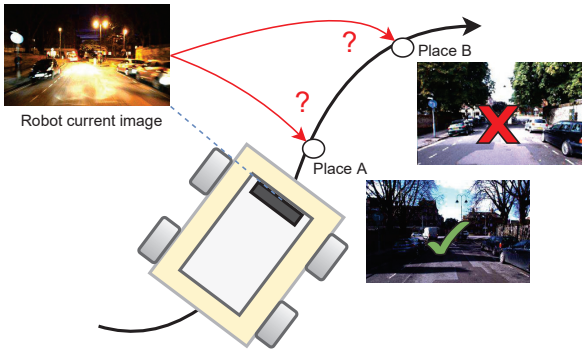


Figure 1. In a SLAM system, the goal of VPR might be to find the matching image between the current image and database images for loop-closure detection. The figure serves as a compelling illustration of the need to address the problem.

The implementation of feature-extraction techniques is a vital phase in VPR systems as it involves recognizing and describing distinct patterns or indicators in an image and then using them for matching and recognition. Most of the state-of-the-art methods can be broadly categorized into two types based on the nature of image-wise descriptors: local and global descriptors. Local image descriptors focus on extracting distinctive features from key points in the image (e.g., Scale-Invariant Feature Transform (SIFT) [10], Speeded-Up Robust Features (SURF) [11], Oriented FAST and Rotated BRIEF (ORB) [12], and Binary Robust Invariant Scalable Keypoints (BRISK) [13]. On the other hand, global image descriptors aim to capture a holistic representation of the entire scene (e.g., Bag-of-Visual-Words (BoVW) [14], Convolutional Neural Network (CNN) [15], Fisher Vectors [16], and Vector of Locally Aggregated Descriptors (VLAD) [17].

Although many studies have been conducted recently, there are still open questions regarding the long-term operation of robots [18], as can be seen in Figure 2.

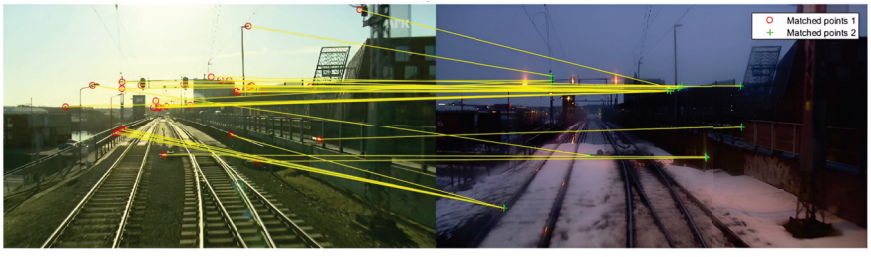


Figure 2. Speeded-Up Robust Features (SURF) [11] applied to Nordland dataset [19] (licensed under Creative Commons) for daytime versus evening. While effective in certain scenarios, SURF may struggle when confronted with significant changes in illumination, as it relies on local features that are not inherently invariant to lighting variations.

Long-term SLAM models need to be robust to changes in the robot's environment caused by different conditions (day–night cycles, changing weather, and so on), occlusion, and viewpoints [20] and deal with the stability–plasticity dilemma, which is a concept in neuroscience and machine learning. This dilemma refers to the challenge of finding the right balance between stability and plasticity in learning systems. Whereas stability refers to the ability of a system to maintain existing knowledge, plasticity refers to adapting and learning from new experiences [18].

Inspired by neuroscience discoveries, various brain-inspired methodologies have been proposed and demonstrated that the analysis of animal behaviors and the biological process of navigation and mapping can produce interesting insights for robotic applications in the world under extremely different environmental conditions. In [21–23], Milforld et al. proposed the RatSLAM system, which includes a bioinspired SLAM system based on computational models of the rodent hippocampus. The system is based on a Continuous Attractor Neural Network (CANN) and can construct a cognitive map through low-resolution monocular image data and odometry information. In [24], Silveira et al. proposed the DolphinSLAM inspired by the RatSLAM. This SLAM system is a bioinspired algorithm for underwater robots using probabilistic local features based on the data-association method of the Fast-Appearance-Based Mapping technique (FAB-MAP). In [25], Yuan et al. proposed an entorhinal–hippocampal model with high biological fidelity, which is able to build cognitive maps simultaneously by integrating activities of place cells and grid cells with visual inputs. In [26], Lu et al. proposed a full visual SLAM system combining biologically inspired visual odometry and RatSLAM. In [27], Kazmi et al. introduced Gist+RatSLAM, a framework that integrates the Gist descriptor into the RatSLAM pipeline. In [28], Zhou et al. adopted the RatSLAM algorithm's vision processing based on the Oriented FAST and Rotated BRIEF (ORB) feature-extraction approach. In [29], Zeng et al. proposed a model based on conjunctive head-direction-by-velocity and conjunctive grid-by-velocity cells to integrate movement and sensory information. In [30], Yu et al. proposed a neuroinspired SLAM system, namely NeuroSLAM, which integrates 3D grid cell models and multilayered head-direction cell models based on RatSLAM. In [31], the authors presented an unsupervised learning framework for multisensor representation that yields low-dimensional latent state descriptors that can be used for RatSLAM. In [32], Kasebi et al. used the Scale-Invariant Feature Transform (SIFT) algorithm to improve the visual matching of the RatSLAM system.

Although these works use biologically inspired SLAM methods in order to perform inference based on the data produced by the front end, they do not bring benefits from neuroscience-based methods to visual place recognition and loop-closure tasks. For instance, the visual template feature is organized as a one-dimensional vector whose values only depend on pixel intensity in the RatSLAM algorithm, and this feature is susceptible to changes in illumination intensity.

In recent years, there has been an increasing amount of literature on the integration of biological and neuroscientific principles into the development of visual place recognition algorithms. In [33], Fen et al. investigated multiscale grid cells observed in the mammalian brain for performing visual place recognition. In [34], Neubert et al. explored the relationships between place recognition and HTM theory and presented the Simplified Higher Order Sequence Memory (SHOSM) algorithm. This neurally inspired model is a simplified version of the HTM framework for place recognition. A significant analysis and discussion on the subject were presented by the authors. This theoretical research was then applied to real-world data in combination with CNN-based image descriptors in [35]. Pizzino et al. demonstrated that the usage of the framework originally proposed by [36] can extend the run time during long-term operations when compared to [35]. The outcomes indicate that the suggested architecture is capable of encoding an internal representation of the world by employing a fixed number of cells, thereby enhancing system scalability. In [37], J. Li et al. proposed a loop-closure detection by using a neural hashing algorithm inspired by the fly olfactory circuit, which is applied to a hippocampal–entorhinal-based SLAM system to build cognitive maps with improved robustness and effectiveness.

In this paper, we present a novel visual SLAM method that integrates computational models inspired by the neocortex and hippocampal–entorhinal regions. This study set out to assess the feasibility of implementation in robots that operate in dynamic environments characterized by continuously changing appearances. The usage of Spatial Distributed Representations (SDRs), which are known to be representations in the brain, enables real-time performance and allows for efficient memory storage. A CNN-based encoder is used

to promote the robustness of matching for correspondences in the images. Based on the neocortex model, the network gradually and continually adapts with each new input. The system was validated and verified via the deployment of the SLAM system onboard real-world ground-based robots. The environments where the evaluation of the system took place consisted of a university campus with outdoor and indoor areas and a farm yard that included several animals within a barn.

A summary of the contributions of this work includes the following:

- We present NeoSLAM, a novel long-term visual SLAM that integrates computational models of the neocortex and hippocampal–entorhinal system in order to enhance the efficiency and robustness to changes in the robot's environment caused by different conditions.
- A new loop-closure detector based on spatial-view cells is presented. The method uses binary sparse distributed representations, offering a compact and powerful way to encode complex patterns in data. Unlike traditional dense representations, which require substantial computational resources, our method significantly reduces the computational load, making it particularly suitable for real-time applications.
- We provide a thorough experimental evaluation that involves deploying the system in practical scenarios, enabling a thorough examination of its performance and capabilities within the Robot Operating System (ROS) framework.

This paper proceeds as follows: Section 2 presents the architecture and the detailed model of our NeoSLAM method. Section 3 describes the design of the experiments for investigating and evaluating the performance of NeoSLAM. Sections 4 and 5 provide analyses of the results and suggestions for future work, respectively.

2. NeoSLAM

Over recent years, the interaction between the neocortex and the hippocampus has been investigated and serves as inspiration for our method. Both brain parts are critical regions in the primate brain and work together to cope with various cognitive functions, particularly those related to learning and spatial memory [38].

The neocortex is the outer layer of the brain and is responsible for higher-order cognitive processes, such as perception, language, and conscious thought. It is involved in processing sensory information and integrating it with other information to form complex representations of the world. On the other hand, the hippocampus is a seahorse-shaped structure located deep within the brain underneath the neocortex and is primarily associated with memory formation and spatial navigation. In primates, the hippocampus receives major input via the entorhinal cortex. These inputs come from the ends of many processing streams of the cerebral association cortex, including the visual cortical areas [39].

Similar to rodents, primate place cells, also referred to as spatial-view cells, exhibit selective firing patterns associated with specific locations in the environment. However, primate spatial-view cells often have larger receptive fields, meaning they are active in broader areas of space compared to the more precise place fields of rodents. The firing patterns of primate spatial-view cells are not limited to specific locations alone. They can also show specificity to other spatial features, such as landmarks or boundaries. For example, some spatial-view cells may fire preferentially when the animal is near a particular visual landmark while others may be sensitive to the boundaries of the environment [40].

Based on theories and models of neuroscience, NeoSLAM allows the robot to perform SLAM in real time by recognizing places it has previously visited under variations in appearance and illumination. The system consists of five major modules, namely the encoder, neocortex, spatial-view cells and LCD, pose cells, and experience map, as can be seen in Figure 3. First, we present the neocortex model, and after that, the other parts.

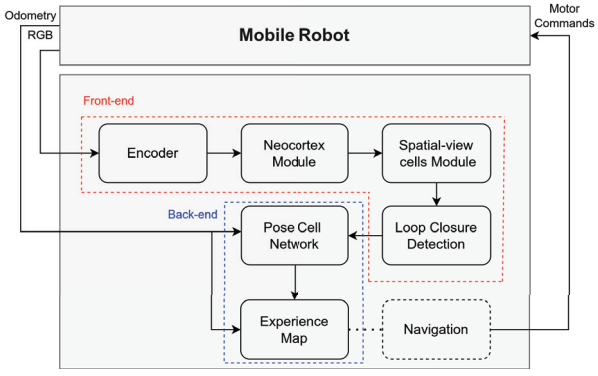


Figure 3. An overview of the major modules of the NeoSLAM system. Most SLAM approaches are commonly divided into two main components: the front-end and the back-end blocks. The former is responsible for real-time sensor data processing, encompassing tasks such as feature extraction and establishing correspondences between consecutive frames. This component plays a crucial role in tracking the system's pose and understanding the environment. On the other hand, the latter focuses on optimization and map refinement. It integrates the accumulated sensor measurements over time, corrects for errors, and optimizes the estimated trajectory and map. In our system, both components draw inspiration from neuroscience models.

2.1. Neocortex Model

In this work, we use one framework that models a number of structural and algorithmic properties of the neocortex, namely hierarchical temporal memory (HTM) proposed by Hawkins in [41]. In this framework, the brain processes information by using sparse distributed representations (SDRs).

2.1.1. Sparse Distributed Representations

The neocortex does not directly receive photons or vibrations from the external environment. Before entering the brain for processing, sensory signals need to be converted into a shared representation space. This widely employed representation space in the mammalian brain is referred to as sparse distributed representations (SDRs).

Sparse neuronal activity refers to the phenomenon observed in mammalian brains where only a small fraction of neurons are active at any given time, while the majority remain silent or exhibit low firing rates. This pattern of sparsity is a fundamental characteristic of neural processing and has been extensively studied to understand how information is represented and processed in the brain.

One key advantage of sparse neuronal activity is its efficiency in representing information. By activating only a subset of neurons, the brain can achieve high-capacity representations while minimizing energy consumption and neural resources. This sparsity also enables the selective and precise encoding of relevant features or stimuli, allowing the brain to efficiently extract and process salient information from the environment.

Given a population of n neurons or binary units, their instantaneous activity is represented as an SDR, i.e., an n-dimensional vector of binary components:

$$\mathbf{x} = [b_0, \cdots, b_{n-1}], \tag{1}$$

where $\mathbf{x} \in \mathbb{S} : \{1, 0\}^n$ and a small percentage of the components are one. The number of components in x that are one is defined as $\omega_x = \|\mathbf{x}\|_1$.

The similarity between two SDR vectors is determined by an overlap score, i.e., the number of bits that are one in both vectors. Let $\Phi : \mathbb{S} \times \mathbb{S} \to \mathbb{N}$ be a function defined by

$$\Phi = \mathbf{x} \cdot \mathbf{y}, \tag{2}$$

where $\mathbf{x}, \mathbf{y} \in \mathbb{S}^n$. The overlap score is simply the number of bits that are one in the same locations.

A match between two SDR vectors occurs if their overlap exceeds the threshold β, i.e.,

$$\Psi = \Phi(\mathbf{x}, \mathbf{y}) \geq \beta. \tag{3}$$

Typically β is set such that $\beta \leq \omega_x$ and $\beta \leq \omega_y$.

2.1.2. Notation

Let n represent the number of minicolumns in the layer, m the number of cells per column, and nm the total number of cells in the layer. A single-level network structure consists of one region that is arranged in minicolumns C_j, $(j = 1, 2, 3, \ldots, n)$, that have multiple cells $c_{i,j}$, $(i = 1, 2, 3, \ldots, m)$. In this work, the following terminology for describing the algorithms is used:

- Cell state: each cell $c_{i,j}$ can be in an active state, in a predictive (depolarized) state, or in a nonactive state.
- Active state: Matrix of active cells, $A^t = \{a^t_{i,j}\}$, where $a^t_{i,j}$ is the active state of the i'th cell in the j'th column at any time step t. A value of 1 indicates an active state and a value of 0 indicates an inactive state.
- Predictive state: Matrix of predictive cells, $\Pi^t = \{\pi^t_{i,j}\}$, where $\pi_{i,j}$ is the predictive state of the i'th cell in the j'th column at any time step t. A value of 1 indicates a predicted state and a value of 0 indicates an unpredicted state.
- Winner cells: Matrix of winner cells, $W^t = \{w^t_{i,j}\}$, where $w^t_{i,j}$ is the winner i'th cell in the j'th column at any time step t. A winner cell is an active cell that was predicted or selected from the bursting minicolumn.
- Minicolumn state: Each minicolumn C_j has a binary state variable A^t_j, where a value of 1 indicates an active state and a value of 0 indicates an inactive state.
- Dendrite segments: Each cell $c_{i,j}$ has one proximal dendrite segment and one or more distal dendrite segments. The proximal dendrite segment is a single shared dendrite segment per each minicolumn C_j of cells and receives feed-forward connections F_j from dimensions of the input I. The distal dendrite segments receive lateral input from nearby cells through the synapses on each segment.
- Synapse: Connection between an axon of one neuron and a dendrite of the other. The dendrite segments contain a number of potential synapses that have an associated permanence value. The permanence value of a synapse is a scalar value ranging from 0.0 to 1.0. If the permanence value of the potential synapse is greater than a threshold ϵ, it becomes a functional synapse, as can be seen in Figure 4. In HTM theory, synapses have binary weights.

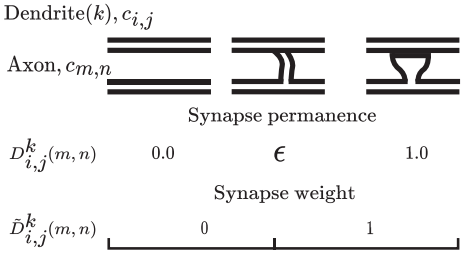


Figure 4. A permanence value is assigned to each potential synapse and represents the growth of the synapse. It is a key parameter that regulates the strength of connections between neurons and allows the model to adapt and encode relevant patterns in the input space over time. Synapses are subject to both potentiation and depression. If a synapse is active and contributes to the cell's activation, its permanence value may be increased. Conversely, its permanence value may be decreased.

Let $D_{i,j}^{k} = \{d_{i,j}^{k}\}$ be a set of distal segments that represents the k'th segment of the cell $c_{i,j}$ used to store the synaptic permanence value. The matrix of dendrite branch connectivity, $\tilde{D}_{i,j}^{k} = \{\tilde{d}_{i,j}^{k}\}$, is defined as

$$\tilde{d}_{i,j}^{k} = \begin{cases} 1, & \text{if } d_{i,j}^{k} \geq \epsilon \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The matrix of positive terms of dendrite branch synapses, $\dot{D}_{i,j}^{k} = \{\dot{d}_{i,j}^{k}\}$, is defined by

$$\dot{d}_{i,j}^{k} = \begin{cases} 1, & \text{if } d_{i,j}^{k} \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

- Learning: the process of learning involves incrementing or decrementing the permanence values ϵ of potential synapses on a dendrite segment.

2.1.3. Hierarchical Temporal Memory

Hierarchical temporal memory (HTM) [42] is a neural network model designed to simulate the structure and function of the human neocortex. It is particularly designed to perform pattern recognition and anomaly detection and exhibits three key properties: sequence learning, continual learning, and sparse distributed representations.

The HTM algorithm is based on the idea that the neocortex learns and recognizes patterns through a hierarchical structure of neurons that process information in a sequence of temporal patterns. This structure allows the neocortex to detect and learn complex patterns over time and make predictions based on past experiences.

The HTM algorithm mimics this structure by creating a hierarchical structure of nodes, where each node represents a pattern or sequence of patterns. The nodes are connected to each other in a way that allows them to learn and recognize patterns over time. The algorithm also includes mechanisms for learning and adapting to new patterns in the input data.

One of the key advantages of the HTM algorithm is its ability to handle noisy and incomplete data. The hierarchical structure allows the algorithm to detect and learn from patterns even when the input data are incomplete or contain errors. The algorithm can also adapt to changes in the input data and learn new patterns without requiring a lot of retraining.

The HTM cells are stacked into columns, where groupings form an HTM region. The region replicates the structure and operations of the cortical column in the neocortex. Cells within a region gather information from three distinct sources: feedforward, contextual, and feedback information is conveyed through three separate connections.

The cortical columns within the neocortex are commonly denoted as minicolumns or columns. The fundamental unit in HTM is the cell. Minicolumns comprise multiple cells, and the network space of HTM is constituted by a substantial number of minicolumns.

The cells can be in inactive, active, or predicted states where segments (links to cells) comprise a collection of synapses. The synapses allow for information to be shared with the cells where, in HTM, this information is displayed as binary values. The information sent to the cell will determine the state. Processing is conducted by a Spatial Pooler algorithm for feedforward input, and temporal memory is used to provide contextual input. It is suggested that feedback is an optional component and is not currently addressed as per HTM theory [43].

Figure 5 shows the HTM model and includes two modules: Spatial Pooler (SP) and temporal memory (TM).

The main role of the Spatial Pooler (SP) in HTM theory is finding spatial patterns in the input data and transforming them into SDRs in a continuous online fashion. It may be decomposed into three stages: overlap calculation, inhibition, and learning [44]. In the case

of visual localization, SP acts as a feature detector and extracts distinctive properties of a place that can be used to recognize this place [34].

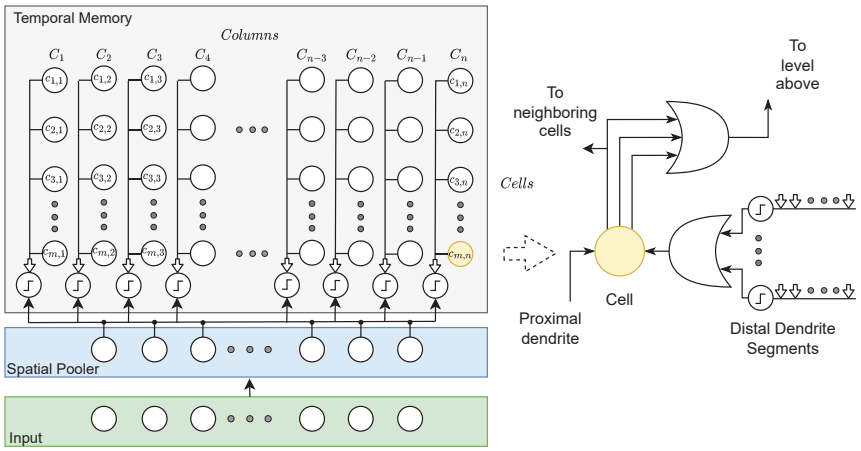


Figure 5. The cells within a column possess a collective proximal dendrite corresponding to the input space through a series of synapses illustrated as arrows. A cell is portrayed along with its distal dendrites on the right side where each dendrite segment establishes multiple synaptic connections with other cells.

Each minicolumn C_j has a binary state variable A_j and can have potential synapses (connections) to the part of the input space [44] or a sparse set of feed-forward connections from the dimensions of the input [35].

In [44], the Hebbian rule ensures that synaptic permanences are adjusted accordingly. The synapse is only connected if the permanence is greater than the threshold required to connect. To ensure that most of the inputs are active within the local inhibition rule, a local mechanism enables a small faction of minicolumns.

In [35], the authors proposed a simplified method to calculate the activation of the minicolumns as follows:

$$A_j^t = 1 \Leftrightarrow \sum_{m \in F_j} I_m^t \geq \theta \tag{6}$$

The SP output represents the activation of minicolumns in response to feedforward inputs.

Meanwhile, the temporal memory learns a sequence and forms a representation in the context of previous inputs. Basically, the algorithm determines the active cells of the columns and learns distal synaptic permanence.

A set of possible synapses is generated at random from a division of cells in the layer within the distal portion. In addition, the permanence of these are also chosen at random.

Equation (7) presents the active state, where we assume that a group C^t of columns for the inhibitory process has already been chosen that is most appropriate to compliment the current feed-forward inputs.

It is assumed that an inhibitory process has already selected a set C^t of columns that best match the current feed-forward input pattern. The calculation of the active state is given by the following equation:

$$a_{i,j}^t = 1 \Leftrightarrow A_j^t \wedge \left(\pi_{i,j}^{t-1} \vee \left(\forall m : \neg \pi_{m,j}^{t-1}\right)\right) \tag{7}$$

A cell is activated in a winning column if this cell was predicted ($\pi_{i,j}^{t-1} = 1$) or if no cells in this minicolumn are predicted. In the latter case, the minicolumn will experience

"bursting", and thus all cells will be activated. The algorithm will select one of the cells in the learning process.

The predictive state for the current time step is calculated as the following equation:

$$\pi_{i,j}^{t} = 1 \Leftrightarrow \exists_k \| \tilde{D}_{i,j}^{k} \circ A^{t} \|_1 > \theta \tag{8}$$

where θ represents the spiking threshold and $\circ$ is an element-wise multiplication (Schur product). A cell will be depolarized if at least one segment k is active, which occurs if there are more than θ connected synapses with active presynaptic cells.

The learning process uses a Hebbian-like rule. It reinforces the dendritic segment that was responsible for activating and causing the depolarization. The choice of those segments $D_{i,j}^{k}$ is such that

$$\forall A_j^t (\pi_{i,j}^{t-1} > 0) \wedge \| \tilde{D}_{i,j}^{k} \circ A^{t-1} \|_1 > \theta \tag{9}$$

It means that the first term selects winning columns that contain correct predictions and the second one selects those segments specifically responsible for the prediction [36].

At this moment, the algorithm will select one of the cells as a learning cell, i.e., the winner cells $w_{i,j}^{t}$, as follows:

$$w_{i,j}^{t} = 1 \Leftrightarrow A_j^t \wedge (\pi_{i,j}^{t-1} > 0) \wedge \| \tilde{D}_{i,j}^{k} \circ A^{t-1} \|_1 > \theta \tag{10}$$

If "bursting" takes place, one of the activated cells is selected on this minicolumn. This cell will represent the context in the future if the current sequence transition repeats:

$$\begin{aligned} w_{i,j}^{t} = 1 \Leftrightarrow & A_j^t \wedge \left(\forall m : \neg \pi_{m,j}^{t-1} \right) \wedge \| \dot{D}_{i,j}^{k} \circ A^{t-1} \|_1 = \\ & max_i \left(\| \dot{D}_{i,j}^{k} \circ A^{t-1} \|_1 \right) \end{aligned} \tag{11}$$

where the function max_i will select the cell with the segment that was closest to being active, even though it was below the threshold.

The synaptic permanence value is adjusted to reward synapses with active presynaptic cells and punish synapses with inactive cells as follows:

$$\Delta D_{i,j}^{k} = r \left(\dot{D}_{i,j}^{k} \circ A^{t-1} \right) - \gamma \dot{D}_{i,j}^{k} \tag{12}$$

where r and γ values will increase and decrease all the permanence values corresponding to presynaptic cells.

In [34], the authors eliminated the dendrite segments and replaced the Hebbian-like learning with one-shot learning. This means that $k = 0$, permanence values are always equal to a maximum value, and $\theta = 0$. They set the variable $\pi_{i,j}^{t}$ if there is an active cell $a_{m,n}$ with a lateral predictive connection to this cell $c_{i,j}$:

$$\pi_{i,j}^{t} = 1 \Leftrightarrow \| \tilde{D}_{i,j} \circ A^{t} \|_1 > 0 \tag{13}$$

In this case, the set of predictive connections is updated based on the previous and current winner cells:

$$\mathbb{P} = \mathbb{P} \cup \left\{ \left(c_{m,n}, c_{i,j} \right) : w_{m,n}^{t-1} \wedge w_{i,j}^{t} \wedge \left(\nexists l : \left(c_{l,n}, c_{i,j} \right) \in \mathbb{P} \right) \right\} \tag{14}$$

$$\mathbb{P} \subset \left\{ \left(c_{m,n}, c_{i,j} \right) : i, j, m, n \in \mathbb{N} \right\}$$

$$w_{i,j}^{t} = 1 \Leftrightarrow \left(a_{i,j}^{t} \wedge p_{i,j}^{t} \right) \vee b_{i,j} \tag{15}$$

The output of HTM temporal memory represents the activation of individual cells across all minicolumns.

2.2. Encoder

Initiating the application of an HTM system involves the initial phase of transforming a data source into a Sparse Distributed Representation (SDR) through a process known as encoding [45]. The encoding process mirrors the functions performed by sensory organs in humans and other animals.

According to Sünderhauf et al. [15], the features generated by Convolutional Neural Networks (CNNs) outperform other methods for the task of visual place recognition in robotics. Despite being trained for a highly specific target task, these models can be effectively applied to deal with different problems. The authors demonstrated that deep features from different layers of CNNs consistently perform better than the traditional system as SIFT or SURF. Additionally, they established important results from which the features are extracted; higher layers of the CNN hierarchy encode semantic information, middle layers exhibit robustness against appearance changes, and top layers are more robust with respect to viewpoint changes.

Indeed, CNNs have become fundamental tools in computer vision, and their architecture is inspired by the visual-processing systems in biological organisms [46]. The success of CNNs has contributed to our understanding of how the brain processes visual information and has enabled the exploration of the connections between artificial and biological intelligence.

Based on [35], the method uses the *conv3* layer of the pretrained CNN AlexNet [47]. This network consists of eight layers: five convolutional layers, two fully connected hidden layers, and one fully connected output layer. The length of the AlexNet-based descriptor is l_{cnn} = 64,896 and can be excessive for HTM Spatial Pooler performing the SDR transformation. Because of that, we use dimensionality reduction techniques based on the random projection [15] and binarization by using the method proposed in [35], namely binary locality-sensitive hashing (LSH).

2.3. Spatial-View-Cells Module

This module models spatial-view cells that fire whenever the robot views a certain part of the environment, as primates do. It is responsible for resetting the accumulative errors of the odometry. Each cell represents what the robot is perceiving, similar to the RatSLAM system. However, when a novel visual scene is seen, a new view cell is not necessarily created and associated with the HTM descriptor, as can be seen in Figure 6.

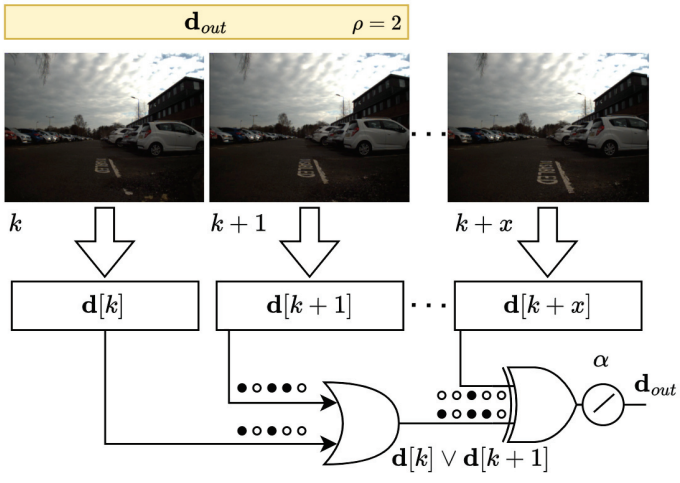


Figure 6. Spatial-view-cell model based on SDR properties and the neurons in primates' hippocampus that respond when a certain part of the environment is in the animal's field of view.

We took advantage of the sparsely distributed representation properties. In [48], Ahmad and Hawkins state that empirical evidence demonstrates that every region of the neocortex represents information using sparse activity patterns at any point in time, and the sparsity might vary from less than one percent to several percent of the total neurons. In other words, SDRs are vectors with thousands of bits and at any point in time a small percentage of the bits are 1's and the rest are 0's. The meaning of the features is represented in the set of active bits. As a consequence, if two different descriptors have an active bit in the same location, they share the same attribute.

Therefore, we can take advantage of an important property of SDRs that is related to the union. It is possible to take a set of SDRs and form a new SDR and maintain their attributes of them. For this, we simply make an *OR* operation, and the result is compared to determine if it is a member of the set of SDRs used to form the union.

Given a set $\mathbb{D}$ that represents a sequence of SDR descriptors, defined as

$$\mathbb{D} = [\mathbf{d}[k], \mathbf{d}[k + N_0]], \tag{16}$$

where $\mathbf{d} \in \mathbb{S}^n$ is defined in Section 2.1.1 and represents the images taken at a specific place and at particular instants in discrete-time $k \in \mathbb{N}$, and N_0 is an interval defined by

$$N_0 = [0, x] = \{x \in \mathbb{N} : \Phi(k + x, k + x + 1) \geq \alpha\} \tag{17}$$

The output of this model is the descriptor

$$\mathbf{d}_{out} = \mathbf{d}[k] \vee \mathbf{d}[k + N_0], \tag{18}$$

The parameter α can be understood as the maximum overlap in order to control the sparsity of the results.

Additionally, we define the parameter ρ as the maximum size of the interval of the spatial-view cells that enclose representations with the similarity between SDRs, i.e.,

$$N_0 = [0, \min(x, \rho)] \tag{19}$$

This method helps to avoid the corridor problem, i.e., the absence of a significant structure along both outdoor and indoor environments.

2.4. Pose Cell Network

The Pose Cell Network is designed to simulate the functioning of neurons in the rodent brain, specifically those found in the hippocampus and entorhinal cortex. These neurons, known as pose cells, are thought to play a fundamental role in spatial representation and navigation. In the RatSLAM system, the Pose Cell Network emulates the rat's ability to create a cognitive map of its environment by encoding information about the animal's position and orientation [49].

A three-dimensional continuous attractor network models the pose cells, where the dimensions represent the pose (i.e., (x, y, θ)) of a ground-based robot with activity matrix P.

As the animal explores its environment, these pose cells fire in a manner that creates a unique neural signature for different locations and orientations. The integration of these pose cell activations over time forms a neural representation of the spatial layout, allowing the RatSLAM system to build and update a map of the environment in a self-supervised manner [23].

2.5. Experience Map

Pose cells represent a finite area; therefore, a single cell is represented by multiple physical places via the wrapping behavior of the cube's edges.

An estimate of the robot's global pose is provided by the experience map and is created by the combination of spatial-view and pose cells.

The map results in a graph where each node represents a different state in the spatial-view and pose cells. When the state of these components changes and is therefore not matching with any pre-existing node in the graph, a new experience node is created within the graph. With these transitions between experiences, links are formed between the current and previous nodes [23].

3. Experimental Validation

In this section, we present the experimental procedure for evaluating the NeoSLAM. To evaluate the system performance, we implemented it in two different robots by using the Robot Operating System (ROS Melodic) in three different environments.

The NeoSLAM project deals with complex software tools and dependencies, making it challenging to ensure consistent results across different computing environments (available in https://github.com/cappizzino/neoslam_ws). To address this issue, a Singularity version 3.9.5 container was integrated.

Singularity allows one to create containerized environments that encapsulate all the dependencies and configurations needed for ROS applications. This makes it easy to share ROS-based projects across different systems and ensures consistent behavior regardless of the host environment.

Besides, it is possible to define the exact software versions, libraries, and configurations required for ROS project within the container. This ensures that the code behaves consistently across different development and deployment stages.

Singularity containers can be shared with other researchers and developers, facilitating collaboration in the robotics community. This is particularly useful when working on open-source ROS projects or when collaborating on research initiatives.

Next, we describe the experimental setup and the evaluation metrics used.

3.1. Experimental Setup

The first robot used in order to evaluate our model was the Clearpath Husky A200. Husky is an Unmanned Ground Vehicle (UGV), developed by Clearpath Robotics. It is a differential-drive-wheeled robot that has a top speed of 1.0 m/s. The robot is equipped with a bumblebee stereo camera mounted on the pan tilt unit where only the left camera was utilized for collecting images.

The second UGV is a hay-cleaning robot equipped with a simple Microsoft LifeCam VX-700, Rion AH200C IMU, and C16 LS LiDAR, which realizes 360° three-dimensional scanning with 16 laser beams. The robot has differential wheel drive and an onboard computer running Ubuntu. With this robot, we decided to build a map by using the LiDAR-Inertial Odometry Smoothing and Mapping (LIO-SAM) ROS package, a state-of-the-art LiDAR SLAM method, in order to create quite accurate ground truth data.

3.2. Environments

In the evaluation of the neocortex model, three different environments were selected to verify and validate the effectiveness of the model created, namely a Robotarium, Heriot-Watt University campus, and a cow barn.

3.2.1. Robotarium

Figure 7 shows the Robotarium environment, which is a multipurpose robotics laboratory at the Edinburgh Centre for Robotics (HWU). The area is very cluttered, consisting of several workstations with desks and chairs, robots positioned around the room, etc.

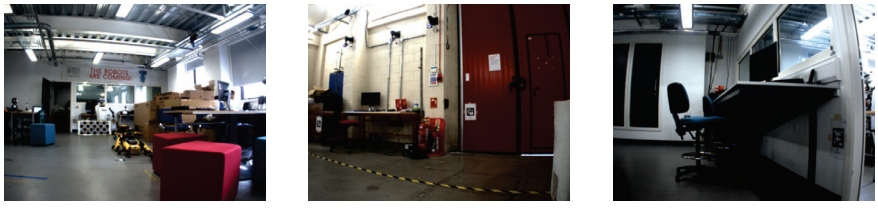


Figure 7. Images collected by the Dual UR5 Clearpath Husky robot at the Heriot-Watt University indoor Robotarium.

3.2.2. Heriot-Watt University

We evaluated our method at Heriot-Watt University, which includes both outdoor and indoor areas. The indoor corridors consisted of a clear area with posters positioned along the walls of the corridor (Figure 8). There were a number of fire doors that were opened ahead of the robot. This area represents a challenge for the neocortex model as many of the images captured are very similar. Meanwhile, the outdoor environment is made up of roads, sidewalk, and car-parking areas.

Figure 8. A selection of images of the environment within the Heriot-Watt University campus where the furthest right image displays the route that the robot was teleoperated throughout.

3.2.3. Cow Barn on the Farmland

The cow barn is an open-sided structure where cattle eat, as can be seen in Figure 9. The main area of the barn is filled with bales of hay stacked and includes individual stalls for cows and a dedicated feeding area.

Figure 9. Images and environment from the cow barn on the farmland.

3.3. Evaluation Metrics

Establishing metrics for visual place recognition typically involves defining evaluation criteria that assess the performance and precision of the recognition system. Some common metrics for VPR include

- Precision (P);
- Recall (R); and
- Area under the precision–recall curve (AUC).

Precision (P) measures the proportion of correctly identified positive instances (true positives—TP) out of the total instances identified as positive, including the false positives (FP). It quantifies the precision of positive predictions made by a VPR system. In other

words, precision answers the question: "Of all the items the system identified as positive, how many were actually positive?":

$$P = \frac{\#TP}{\#TP + \#FP} \tag{20}$$

Recall (R), also known as the sensitivity or the true positive rate, measures the proportion of correctly identified positive instances out of all the actual positive instances, including false negatives (FN). It quantifies the system's ability to find all the relevant positive instances. In other words, recall answers the question: "Of all the actual positive items, how many did the system identify correctly?":

$$R = \frac{\#TP}{\#TP + \#FN} \tag{21}$$

In order to illustrate the trade-off between precision and recall for a classification model, a precision–recall curve can be generated by plotting these pairs of precision and recall values for various thresholds. In a precision–recall curve, different decision thresholds are used to classify data points, and for each threshold, precision and recall values are calculated.

The area under the precision–recall curve (AUC) is a numerical value that quantifies the overall performance of a classification model as represented by its precision–recall curve. It summarizes the classifier's performance in a single scalar value, with a higher AUC indicating better performance.

3.4. General Procedures

The table below shows the general configuration of the experiments (Table 1).

Table 1. Configuration of the experiments.

Experiment	Robot	Environment	Ground Truth
1	Clearpath Husky	Robotarium	Annotated manually
2	Clearpath Husky	Robotarium	Annotated manually
3	Hay-cleaning robot	Cow barn	LiDAR-based SLAM

In Experiment 1 and 2, the images were collected by the Husky robot during the teleoperation. We ensure real-time running, i.e., images were collected while the full data-processing pipeline for loop-closure detection was running. In these cases, odometry data were recorded with 10 Hz and images with approximately 1.0 Hz. In the last experiment, ground truth data were created by using a LiDAR-based SLAM system. Places with a ground truth distance < 3 m are considered as the same place.

3.5. Parameter Configurations

CNN features were computed by a pretrained AlexNet by using Pytorch. The neocortex model was implemented in Python 2.7.17 by using the Numenta Platform for Intelligent Computing (NuPIC) developed by Numenta, which implements the HTM learning algorithms. We implemented both Spatial Pooler and Temporal Memory algorithms to create temporal and spatial representations of the images. A description of the HTM's parameters is given in [50] and their values can be found in the project repository.The main parameters' values can be seen in the table below (Table 2).

Table 2. Main parameters for Spatial Pooler, Temporal Memory and Spatial-View Cells module

Parameter	Module	Robotarium	HWU	Cow Barn
input-, columnDimensions	Spatial Pooler	2048	2048	2048
numActiveColumnsPerInhArea	Spatial Pooler	40	40	40
columnDimensions	Temporal memory	2048	2048	2048
cellsPerColumn	Temporal memory	32	32	32
activationThreshold	Temporal memory	4	4	4
maximum size of the interval ρ	Spatial-view cells	1	3	2
maximum overlap α	Spatial-view cells	-	384	384

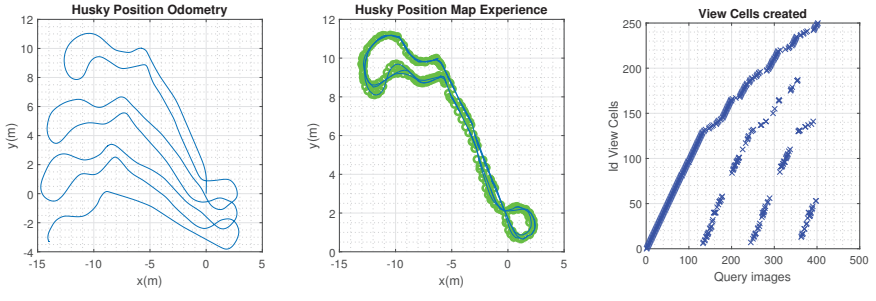
4. Results

In this section, we discuss the results obtained by using the proposed NeoSLAM method in the delineated experimental scenarios. We validate the following real-world experiments by using evaluation metrics compared to the original RatSLAM.

4.1. Robotarium

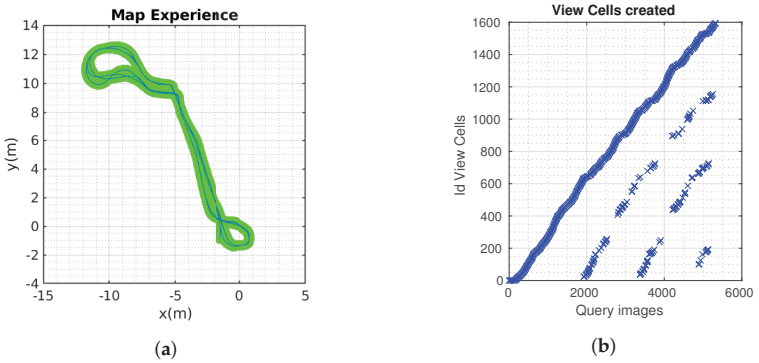
In this scenario, NeoSLAM and RatSLAM generated a consistent topological map of the environment. The algorithms demonstrated real-time performance, processing sensor data and updating the map and trajectory estimates on the fly (Figures 10–13).

The SLAM algorithms proved to be robust against sensor noise, occlusions, and dynamic objects. It successfully handled scenarios with moving objects, such as people walking in the environment, without significant degradation in performance.



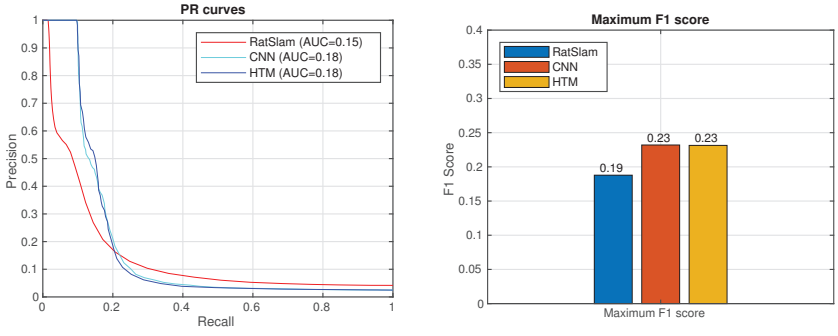
(**a**) (**b**) (**c**)

Figure 10. NeoSLAM results. (**a**) Odometry information from Husky robot. (**b**) Experience map, showing the topological map. (**c**) Visual templates over the duration of the experiment.



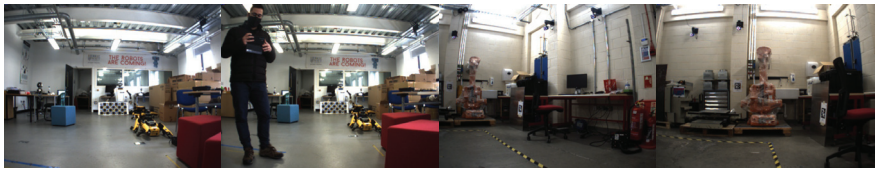
(**a**) (**b**)

Figure 11. RatSlam results. (**a**) Experience map, showing the topological map. (**b**) Visual templates over the duration of the experiment.



(**a**) (**b**)

Figure 12. Spatial-view cell NeoSLAM (HTM), CNN, and view-cell RatSLAM descriptors. (**a**) Precision–recall curves. (**b**) Maximum F1 Scores.



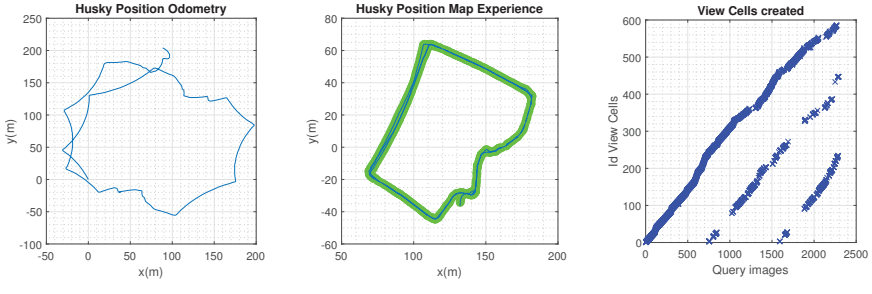
(**a**) (**b**)

Figure 13. NeoSlam results: examples of true and false positive places. (**a**) True positive images. (**b**) False positive images.

4.2. Heriot-Watt University

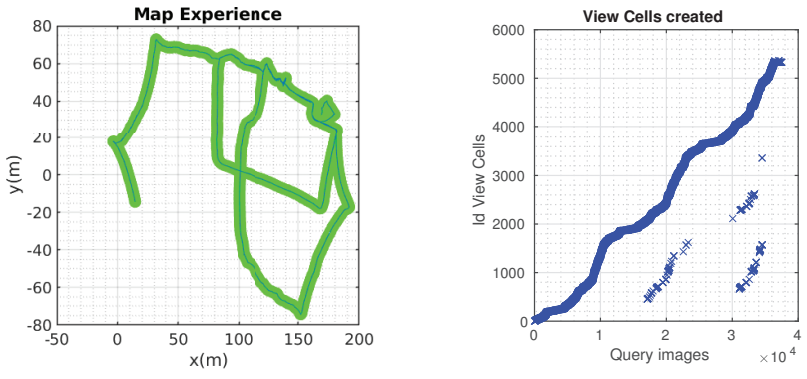
Husky completed three laps, approximately 1.1 km. The route was completed at different times during the day to ensure that the model was tested in different daylight hours. This would assess the resilience of the neocortex model in different lighting: afternoon and evening (Figures 14–16).

The transitions between outdoor and indoor environments multiple times, simulating a scenario where a robotic system needs to adapt to changing conditions, aims to assess the adaptability and robustness of our approach across diverse and dynamic scenarios. In total, 2504 images were analyzed. Even so, the system was capable of running in real time without the loss of information due to the possibility of comparing by using the hamming distance.



(**a**) (**b**) (**c**)

Figure 14. NeoSLAM results. (**a**) Odometry information from Husky robot. (**b**) Experience map, showing the topological map. (**c**) Visual templates over the duration of the experiment.



(**a**) (**b**)

Figure 15. RatSlam results. (**a**) Experience map, showing the topological map. (**b**) Visual templates over the duration of the experiment.

(**a**) (**b**)

Figure 16. NeoSlam results: examples of true and false positive places. (**a**) True positive images. (**b**) False positive images.

4.3. Cow Barn

In this scenario, cows are moving and eating in different places. The illuminations exhibit significant variation at the middle and extremity of the barn.

Figure 17 (left) shows the trajectory built by LIO-SAM [51]. The ground truth is calculated and is shown in Figure 17 (right). The robot completed three laps, approximately 400 m. The comparison between the methods is shown below (Figure 18).

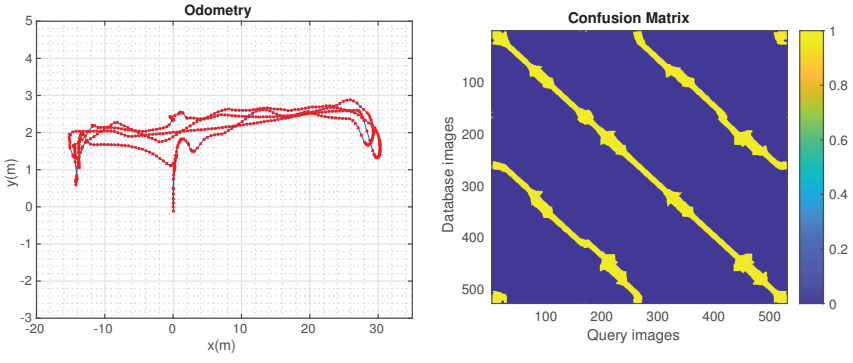
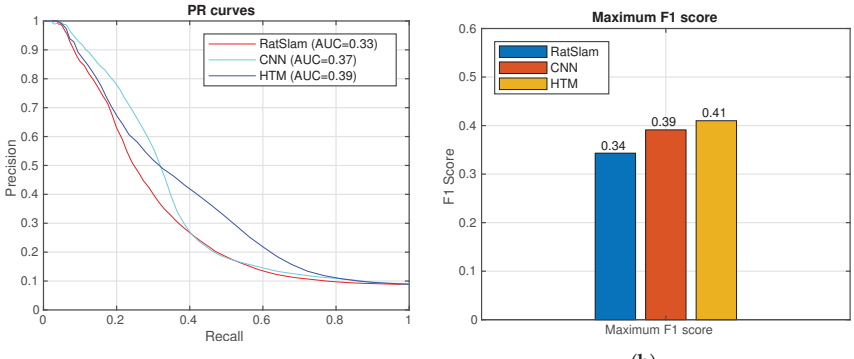


Figure 17. LIO-SAM odometry and confusion matrix.



(**a**) (**b**)

Figure 18. Spatial-view-cell NeoSLAM (HTM), CNN, and view-cell RatSLAM descriptors. (**a**) Precision–recall curves. (**b**) Maximum F1 Scores.

5. Discussion and Conclusions

The aim of this work was to propose a new visual-based SLAM method using a computational model that integrates neocortex and hippocampus models to deal with environments where appearance continually changes. Our approach focuses on real-world robot implementations, ensuring the relevance and applicability of the findings to practical robotics applications.

Through three distinct experiments, the performance of traditional RatSLAM and NeoSLAM are compared. RatSLAM relies on feature extraction and matching in visual data, which can be affected by lighting conditions, occlusions, and environmental changes, and NeoSLAM is proposed to deal with this problem. The proposed method, with the inclusion of the neocortex model and spatial-view cells for visual place recognition and loop-closure detection, outperforms RatSLAM.

Visual place recognition can benefit immensely from incorporating features learned by using CNNs. Nevertheless, these approaches are computationally intensive, so it is necessary to examine the capability of real-time operation. Using a pretrained CNN is an important alternative from the aspects of reducing the computational cost.

Representations in HTM are binary sparse distributed, operations on these structures are very efficient related to time, and the features obtained from CNNs can be incorporated. Besides, the sequential information that this model incorporates might be a key element of successful approaches for place recognition in changing environments.

In the Robotarium scenario, we did not observe a significant improvement in spite of the fact that the area under the precision–recall curve of the HTM model was slightly higher than RatSLAM. It is important to note the number of view cells created in RatSLAM is approximately ten times higher than the spatial-view cells of NeoSLAM, which completes the analysis of the images every one second.

In the second experiment undertaken at Heriot-Watt University, the results of NeoSLAM overcome the performance of RatSLAM. The complexity of the scenario, the strong similarity of places, and the appearance changes caused a lot of false positives, which hamper the construction of a topological map. These results align with the hypothesis that the incorporation of CNNs into HTM generates distinctive features, contributing to enhanced long-term visual place recognition.

In the final scenario, a ground truth was created by using LIO-SAM, which works by receiving data from a 3D LiDAR and an IMU in order to estimate the robot's state and trajectory. It formulates the problem as a Maximum a Posteriori Probability (MAP) estimate by using a factor graph to solve it. By doing this, a comparison between NeoSLAM and RatSLAM improves as they can be compared directly for the same types of data in this case. Once again, NeoSLAM obtained a better result.

The evidence presented in this study supports the effectiveness of the proposed NeoSLAM method in achieving consistent topological mapping and a real-time performance in real-world indoor and outdoor environments. NeoSLAM showcases robustness against various challenges, making it suitable for practical robotic applications.

The scope of this study was limited in terms of the changes in the robot's environment caused by continuously changing appearances. A future study could assess the long-term effects of dark environments and the degree of change in viewpoint, which can significantly impact the challenge associated with VPR. Besides, some limitations were observed during the experiments regarding the complexity and number of parameters of this model. The algorithm has theoretical limitations that require further investigation.

In conclusion, this study not only builds upon the foundational knowledge established by previous works but also offers practical applications. We consider the performance to be an attractive choice for adaptation within field robot applications. These findings contribute to the advancement of SLAM techniques, mainly for long-term robot operation.

Author Contributions: Conceptualization, C.A.P.P., R.R.C. and P.A.V.; methodology, C.A.P.P., R.R.C. and P.A.V.; software, C.A.P.P.; validation, C.A.P.P., R.R.C. and P.A.V.; formal analysis, R.R.C. and P.A.V.; investigation, C.A.P.P. and D.M.; writing—original draft preparation, C.A.P.P.; writing—review and editing, R.R.C. and P.A.V. All authors have read and agreed to the published version of the manuscript.

Funding: This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior–Brasil (CAPES)—Finance Code 001.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available in https://github.com/cappizzino/neoslam_ws.

Acknowledgments: The authors would like to express their sincere gratitude to Ingeniarius Lda (Alfena, Portugal) for providing data related to its robot in the cow barn experiment.

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

Abbreviations

The following abbreviations are used in this manuscript:

BC	Border Cells
BoVW	Bag-of-Visual-Words
BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
CANN	Continuous Attractor Neural Network
CNN/ConvNet	Convolutional Neural Network
FAB-MAP	Fast-Appearance-Based Mapping technique
FAST	Features from accelerated segment test
GC	Grid cells
HD	Head-direction cells
HP	Hippocampus
HTM	Hierarchical temporal memory
LCD	Loop-closure detection
LIO-SAM	LiDAR-Inertial Odometry Smoothing and Mapping
MAP	Maximum a Posteriori Probability
MEC	Medial Entorhinal Cortex
ORB	Oriented FAST and Rotated BRIEF
ORE	Offshore Renewable Energy sector
PC	Place cell
ROS	Robot Operating System
SLAM	Simultaneous Localization and Mapping
SDR	Sparse distributed representations
SIFT	Scale-Invariant Feature Transform

SP	Spatial Pooler
SURF	Speeded-Up Robust Features
TM	Temporal memory
VLAD	Vector of Locally Aggregated Descriptors
VPR	Visual place recognition

References

1. Thrun, S.; Burgard, W.; Fox, D. *Probabilistic Robotics*; MIT Press: Cambridge, MA, USA, 2005.
2. Mitchell, D.; Blanche, J.; Zaki, O.; Roe, J.; Kong, L.; Harper, S.; Robu, V.; Lim, T.; Flynn, D. Symbiotic System of Systems Design for Safe and Resilient Autonomous Robotics in Offshore Wind Farms. *IEEE Access* **2021**, *9*, 141421–141452. [CrossRef]
3. Mitchell, D.; Blanche, J.; Harper, S.; Lim, T.; Gupta, R.; Zaki, O.; Tang, W.; Robu, V.; Watson, S.; Flynn, D. A review: Challenges and opportunities for artificial intelligence and robotics in the offshore wind sector. *Energy AI* **2022**, *8*, 100146. [CrossRef]
4. Harper, S.T.; Mitchell, D.; Nandakumar, S.C.; Blanche, J.; Lim, T.; Flynn, D. Addressing Non-Intervention Challenges via Resilient Robotics Utilizing a Digital Twin. In Proceedings of the 2023 IEEE International Conference on Omni-layer Intelligent Systems (COINS), Berlin, Germany, 23–25 July 2023; pp. 1–8. [CrossRef]
5. Cheah, W.; Garcia-Nathan, T.B.; Groves, K.; Watson, S.; Lennox, B. Path Planning for a Reconfigurable Robot in Extreme Environments. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 10087–10092. [CrossRef]
6. Mitchell, D.; Emor Baniqued, P.D.; Zahid, A.; West, A.; Nouri Rahmat Abadi, B.; Lennox, B.; Liu, B.; Kizilkaya, B.; Flynn, D.; Francis, D.J.; et al. Lessons learned: Symbiotic autonomous robot ecosystem for nuclear environments. *IET Cyber-Syst. Robot.* **2023**, *5*, e12103. [CrossRef]
7. Baskoro, C.H.A.H.B.; Saputra, H.M.; Mirdanies, M.; Susanti, V.; Radzi, M.F.; Aziz, R.I.A. An Autonomous Mobile Robot Platform for Medical Purpose. In Proceedings of the 2020 International Conference on Sustainable Energy Engineering and Application (ICSEEA), Tangerang, Indonesia, 18–20 November 2020; pp. 41–44. [CrossRef]
8. Siegwart, R.; Nourbakhsh, I.R.; Scaramuzza, D. *Introduction to Autonomous Mobile Robots*; MIT Press: Cambridge, UK, 2011.
9. Lowry, S.; Sünderhauf, N.; Newman, P.; Leonard, J.J.; Cox, D.; Corke, P.; Milford, M.J. Visual Place Recognition: A Survey. *IEEE Trans. Robot.* **2016**, *32*, 1–19. [CrossRef]
10. Lowe, D. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
11. Bay, H.; Ess, A.; Tuytelaars, T.; Gool, L.V. Speeded-Up Robust Features (SURF). *Comput. Vis. Image Underst.* **2008**, *110*, 346–359. [CrossRef]
12. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2564–2571. [CrossRef]
13. Leutenegger, S.; Chli, M.; Siegwart, R.Y. BRISK: Binary Robust invariant scalable keypoints. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2548–2555. [CrossRef]
14. Sivic, J.; Zisserman, A. Video Google: A text retrieval approach to object matching in videos. In Proceedings of the Ninth IEEE International Conference on Computer Vision, Nice, France, 13–16 October 2003; Volume 2, pp. 1470–1477. [CrossRef]
15. Sünderhauf, N.; Shirazi, S.; Dayoub, F.; Upcroft, B.; Milford, M. On the performance of ConvNet features for place recognition. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015; pp. 4297–4304. [CrossRef]
16. Perronnin, F.; Liu, Y.; Sánchez, J.; Poirier, H. Large-scale image retrieval with compressed Fisher vectors. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 3384–3391. [CrossRef]
17. Jégou, H.; Douze, M.; Schmid, C.; Pérez, P. Aggregating local descriptors into a compact image representation. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 3304–3311. [CrossRef]
18. Sousa, R.B.; Sobreira, H.M.; Moreira, A.P. A systematic literature review on long-term localization and mapping for mobile robots. *J. Field Robot.* **2023**, *40*, 1245–1322. [CrossRef]
19. Skrede, S. Nordlandsbanen: Minute by Minute, Season by Season. 2013. Available online: https://nrkbeta.no/2013/01/15/nordlandsbanen-minute-by-minute-season-by-sea/son (accessed on 21 June 2020).
20. Schubert, S.; Neubert, P. What makes visual place recognition easy or hard? *arXiv* **2021**, arXiv:2106.12671.
21. Milford, M.; Wyeth, G. Hippocampal models for simultaneous localisation and mapping on an autonomous robot. In Proceedings of the 2003 Australasian Conference on Robotics and Automation, Brisbane, Australia, 1–3 December 2003; Wyeth, G., Roberts, J., Eds.; Australian Robotics and Automation Association: Brisbane, Australia, 2003; pp. 1–10.
22. Milford, M.; Wyeth, G. Mapping a Suburb With a Single Camera Using a Biologically Inspired SLAM System. *IEEE Trans. Robot.* **2008**, *24*, 1038–1053. [CrossRef]
23. Ball, D.; Heath, S.; Wiles, J.; Wyeth, G.; Corke, P.; Milford, M. OpenRatSLAM: An open source brain-based SLAM system. *Auton. Robot.* **2013**, *34*, 149–176. [CrossRef]

24. Silveira, L.; Guth, F.; Drews-Jr, P.; Ballester, P.; Machado, M.; Codevilla, F.; Duarte-Filho, N.; Botelho, S. An Open-source Bio-inspired Solution to Underwater SLAM. *IFAC-PapersOnLine* **2015**, *48*, 212–217. [CrossRef]
25. Yuan, M.; Tian, B.; Shim, V.A.; Tang, H.; Li, H. An Entorhinal-Hippocampal Model for Simultaneous Cognitive Map Building. In Proceedings of the AAAI, Austin, TX, USA, 25–30 January 2015; Bonet, B., Koenig, S., Eds.; AAAI Press: Washington, DC, USA, 2015; pp. 586–592.
26. Lu, H.; Xiao, J.; Zhang, L.; Yang, S.; Zell, A. Biologically inspired visual odometry based on the computational model of grid cells for mobile robots. In Proceedings of the 2016 IEEE International Conference on Robotics and Biomimetics (ROBIO), Qingdao, China, 3–7 December 2016; pp. 595–601. [CrossRef]
27. Kazmi, S.M.A.M.; Mertsching, B. Gist+RatSLAM: An Incremental Bio-inspired Place Recognition Front-End for RatSLAM. *EAI Endorsed Trans. Creat. Technol.* **2016**, *3*, e3. [CrossRef]
28. Zhou, S.C.; Yan, R.; Li, J.X.; Chen, Y.K.; Tang, H. A brain-inspired SLAM system based on ORB features. *Int. J. Autom. Comput.* **2017**, *14*, 564–575. [CrossRef]
29. Zeng, T.; Si, B. Cognitive Mapping Based on Conjunctive Representations of Space and Movement. *Front. Neurorobotics* **2017**, *11*, 61. [CrossRef]
30. Yu, F.; Chancan, M.; Shang, J.; Hu, Y.; Milford, M. NeuroSLAM: A Brain Inspired SLAM System for 3D Environments. *Biol. Cybern.* **2019**, *113*, 515–545. [CrossRef] [PubMed]
31. Çatal, O.; Jansen, W.; Verbelen, T.; Dhoedt, B.; Steckel, J. LatentSLAM: Unsupervised multi-sensor representation learning for localization and mapping. In Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, 30 May–5 June 2021; pp. 6739–6745. [CrossRef]
32. Salimpour Kasebi, S.; Seyedarabi, H.; Musevi Niya, J. Hybrid navigation based on GPS data and SIFT-based place recognition using Biologically-inspired SLAM. In Proceedings of the 2021 11th International Conference on Computer Engineering and Knowledge (ICCKE), Mashhad, Iran, 28–29 October 2021; pp. 260–266. [CrossRef]
33. Fan, C.; Chen, Z.; Jacobson, A.; Hu, X.; Milford, M. Biologically-Inspired Visual Place Recognition with Adaptive Multiple Scales. *Robot. Auton. Syst.* **2017**, *96*, 224–237. [CrossRef]
34. Neubert, P.; Ahmad, S.; Protzel, P. A Sequence-Based Neuronal Model for Mobile Robot Localization. In Proceedings of the KI 2018: Advances in Artificial Intelligence, Berlin, Germany, 24–28 September 2018; Trollmann, F., Turhan, A.Y., Eds.; Springer: Cham, Switzerland, 2018; pp. 117–130.
35. Neubert, P.; Schubert, S.; Protzel, P. A Neurologically Inspired Sequence Processing Model for Mobile Robot Place Recognition. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3200–3207. [CrossRef]
36. Hawkins, J.; Ahmad, S. Why Neurons Have Thousands of Synapses, a Theory of Sequence Memory in Neocortex. *Front. Neural Circuits* **2016**, *10*, 23. [CrossRef]
37. Li, J.; Tang, H.; Yan, R. A Hybrid Loop Closure Detection Method Based on Brain-Inspired Models. *IEEE Trans. Cogn. Dev. Syst.* **2022**, *14*, 1532–1543. [CrossRef]
38. Squire, L.; Berg, D.; Bloom, F.; du Lac, S.; Ghosh, A.; Spitzer, N.; Squire, L. *Fundamental Neuroscience*; Elsevier Science: Amsterdam, The Netherlands, 2008.
39. Rolls, E.T. *Cerebral Cortex: Principles of Operation*; Oxford University Press: Oxford, UK, 2016.
40. Rolls, E.T. Chapter 4.2—The primate hippocampus and episodic memory. In *Handbook of Behavioral Neuroscience*; Dere, E., Easton, A., Nadel, L., Huston, J.P., Eds.; Elsevier: Amsterdam, The Netherlands, 2008; Volume 18, pp. 417–626. [CrossRef]
41. Hawkins, J.; Ahmad, S.; Dubinsky, D. Hierarchical Temporal Memory including HTM Cortical Learning Algorithms. Version 0.2.1. Available online: https://www.numenta.com/assets/pdf/whitepapers/hierarchical-temporal-memory-cortical-learning-algorithm-0.2.1-en.pdf (accessed on 30 November 2023).
42. Cui, Y.; Ahmad, S.; Hawkins, J. Continuous Online Sequence Learning with an Unsupervised Neural Network Model. *Neural Comput.* **2016**, *28*, 2474–2504. [CrossRef] [PubMed]
43. Hawkins, J.; Ahmad, S.; Cui, Y. A Theory of How Columns in the Neocortex Enable Learning the Structure of the World. *Front. Neural Circuits* **2017**, *11*, 81. [CrossRef]
44. Cui, Y.; Ahmad, S.; Hawkins, J. The HTM Spatial Pooler—A Neocortical Algorithm for Online Sparse Distributed Coding. *Front. Comput. Neurosci.* **2017**, *11*, 111. [CrossRef] [PubMed]
45. Mnatzaganian, J.; Fokoue, E.; Kudithipudi, D. A Mathematical Formalization of Hierarchical Temporal Memory's Spatial Pooler. *Front. Robot. AI* **2016**, *3*, 81. [CrossRef]
46. Celeghin, A.; Borriero, A.; Orsenigo, D.; Diano, M.; Guerrero, C.A.M.; Perotti, A.; Petri, G.; Tamietto, M. Convolutional neural networks for vision neuroscience: Significance, developments, and outstanding issues. *Front. Comput. Neurosci.* **2023**, *17*, 1153572. [CrossRef] [PubMed]
47. Krizhevsky, A.; Sutskever, I.; Hinton, G. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
48. Ahmad, S.; Hawkins, J. Properties of Sparse Distributed Representations and their Application to Hierarchical Temporal Memory. *arXiv* **2015**, arXiv:1503.07469.
49. Wyeth, G.; Milford, M. Spatial cognition for robots. *IEEE Robot. Autom. Mag.* **2009**, *16*, 24–32. [CrossRef]

50. Hawkins, J.; Ahmad, S.; Purdy, S.; Lavin, A. Biological and Machine Intelligence (BAMI). Initial Online Release 0.4. Available online: https://numenta.com/resources/biological-and-machine-intelligence/ (accessed on 30 November 2023).
51. Shan, T.; Englot, B.; Meyers, D.; Wang, W.; Ratti, C.; Rus, D. LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 5135–5142. [CrossRef]

sensors

MDPI

Article

Implementation of an Artificially Empathetic Robot Swarm

Joanna Siwek [1,†], Patryk Żywica [1,†], Przemysław Siwek [2], Adrian Wójcik [2], Witold Woch [1], Konrad Pierzyński [1] and Krzysztof Dyczkowski [1,*]

[1] Department of Artificial Intelligence, Faculty of Mathematics and Computer Science, Adam Mickiewicz University, Uniwersytetu Poznańskiego 4, 61-614 Poznań , Poland; jsiwek@amu.edu.pl (J.S.); bikol@amu.edu.pl (P.Ż.); witwoc@st.amu.edu.pl (W.W.); konpie1@st.amu.edu.pl (K.P.)

[2] Institute of Robotics and Machine Intelligence, Faculty of Automatic Control, Robotics and Electrical Engineering, Poznan University of Technology, Piotrowo 3A, 60-965 Poznań, Poland; przemyslaw.siwek@put.poznan.pl (P.S.); adrian.wojcik@put.poznan.pl (A.W.)

* Correspondence: krzysztof.dyczkowski@amu.edu.pl

† These authors contributed equally to this work.

Abstract: This paper presents a novel framework for integrating artificial empathy into robot swarms to improve communication and cooperation. The proposed model uses fuzzy state vectors to represent the knowledge and environment of individual agents, accommodating uncertainties in the real world. By utilizing similarity measures, the model compares states, enabling empathetic reasoning for synchronized swarm behavior. The paper presents a practical application example that demonstrates the efficacy of the model in a robot swarm working toward a common goal. The evaluation methodology involves the open-source physical-based experimentation platform (OPEP), which emphasizes empirical validation in real-world scenarios. The paper proposes a transitional environment that enables automated and repeatable execution of experiments on a swarm of robots using physical devices.

Keywords: artificial empathy; swarm; fuzzy sets; similarity measure; open simulation; physical experimentation

Citation: Siwek, J.; Żywica, P.; Siwek, P.; Wójcik, A.; Woch, W.; Pierzyński, K.; Dyczkowski, K. Implementation of an Artificially Empathetic Robot Swarm. *Sensors* **2024**, *24*, 242. https://doi.org/10.3390/s24010242

Academic Editors: Yuanlong Xie, Shiqi Zheng, Zhaozheng Hu and Shuting Wang

Received: 14 December 2023
Revised: 27 December 2023
Accepted: 29 December 2023
Published: 31 December 2023

1. Introduction

The paper describes research motivated by the application of artificial empathy algorithms in a swarm of mobile robots. The goal is to transfer the biological mechanisms of the human brain, such as empathy, to computer systems. This improves the quality of the robots by extending them with cognitive aspects, such as learning and adaptation mechanisms.

We present preliminary results of computer simulations that identified problems causing divergence between simulations and reality, such as significant resource consumption, which limits the number and range of parameters. This limitation prevented the effective evaluation of the algorithms. Therefore, we propose a solution to address the issue of discrepancies between the simulation environment and real experiments. Our solution allows for the verification of complex algorithms and offers universality, accessibility, standardization, and repeatability of results.

The construction of a platform that achieves those objectives requires intensive research. This paper presents the assumptions made and solutions used for the construction of two versions of mobile robot prototypes and an experimentation arena. Particular attention is paid to the possibility of modeling empathetic behavior in the swarm, which had a significant impact on many technical requirements for the devices built.

1.1. Motivation

In the field of robotics, the standardization and reproducibility of research results are primary challenges, particularly in the field of AI. Research outcomes often depend

on various factors, such as hardware configuration, software version, or environmental factors. Simulators, such as CoppeliaSim [1], available in the field, may not accurately reflect reality. One issue in AI research is the significant centralization of the research and development process. Major academic institutions possess substantial resources, which limits opportunities for smaller universities and individual researchers to engage in state-of-the-art research.

One challenge in AI and swarm robotics research is the high entry barrier. Smaller research units may find it difficult to acquire the specialized tools and infrastructure required for this type of research. To address this issue, a proposed solution is to provide an affordable and standardized experimental environment for swarm robotics. Remote access to an environment enables experiments to be conducted without physical access to the test platform, enhancing the potential for international and interdisciplinary collaboration. Cloud access in a pay-as-you-go model could reduce initial expenses on research infrastructure [2].

The transition from the research stage to industrial implementation highlights the challenge of manufacturing and testing swarms in operational conditions. Furthermore, there is a shortage of variable and modifiable test environments for swarms, particularly those with remote access capabilities. The restricted applicability of existing cloud-based testing platforms is also a concern, as demonstrated by the case of DeepRacer [2], which is exclusively designed for robot racing. Imperfections in existing robotics simulators, such as CoppeliaSim, may result in inaccurate representations of reality. Enabling the modification of the real test environment would greatly facilitate the testing of swarm solutions at an early stage of development, leading to the development of more effective products. For example, the DeepRacer platform has a race track that cannot be modified, which limits the number of testing scenarios. Remote access to an experimental platform with mobile robots would enable the testing of swarm algorithms without requiring physical access to the equipment. This approach could reduce capital costs for companies utilizing AI in swarm robotics solutions and redirect capital toward operational activities.

To address this problem, it is necessary to test algorithms in a standardized and predictable environment. This can be achieved by providing greater versatility and computational power than other commonly used tools, such as Kilobots [3]. To explore various swarm robot operation scenarios in different environments and ultimately improve efficiency, it is recommended to introduce a modifiable experimental platform. The creation of a highly automated experimental environment using physical devices could resolve issues that are inherent to simulation software. This would open up new research opportunities for scientists working on challenges related to robotics and artificial intelligence.

In summary, the use of advanced technologies and sensors in artificial intelligence research is essential. These innovations not only simplify and enhance the research process but also broaden the accessibility of AI technologies. Standardized experimental environments, particularly in the realm of swarm robot experiments, address issues related to result reproducibility, fostering international and interdisciplinary collaboration in AI.

The flexibility and universality of the experimental platform are crucial for algorithm testing in diverse environments, leading to the identification of more efficient solutions. The implementation of a highly automated experimental environment, grounded in physical devices, mitigates challenges associated with simulator imperfections and limitations of simulation software.

In addition, remote access to the experimental platform is a game-changer for researchers, as it allows them to conduct experiments without the need for physical presence. This not only streamlines research efforts but also facilitates educational activities in schools, overcoming barriers such as high costs and challenges associated with engaging in AI research. The integration of advanced technologies and the establishment of robust experimental frameworks are essential for advancing AI research and promoting its broader application.

This article presents the process of developing the concept of such an environment, introduces the proposed model of artificial empathy, and discusses the methodology and results obtained in simulation-based experiments and work on building two prototype versions of the solution.

1.2. Empathy Modeling

The human decision process relies highly on a person's knowledge, intelligence, and experience. Emotional intelligence that constitutes a large part of general intelligence [4] plays a significant role in action-taking, especially involving cooperation [5]. In such a case communication is crucial, and a significant part of every message consists of signaling emotions and the inner state of the sender [5]. Empathy is the ability to put oneself in the "mental shoes" of another person, to understand their emotions and feelings [6]. It allows one to predict what kind of behavior can be expected from the target of empathy and to plan own actions accordingly. Since human reasoning is highly imprecise, one does not have complete information about the inner state of the empathy target and can only reason based on imperfect, highly subjective knowledge. When creating artificial intelligence systems, especially cooperative ones like swarms, omitting empathy seems to be wasteful. The attempt to transfer the concept of human empathy to artificial systems is called artificial empathy [7,8].

Human reasoning, particularly the aspect connected with emotions, is imprecise. Therefore, recreating decision mechanisms that include empathy requires tools that allow the model to understand imprecision. Empathy necessitates comparing the emotions and states of others to one's own knowledge, thus requiring imprecise representations of those states. As complete information about the inner state of the empathy target is unavailable, deductions can only be made based on a signal that is not ideal and subjectively produced and understood. Fuzzy sets and linguistic variables offer useful solutions.

Artificial empathy is an increasingly popular research topic [9]. Its inclusion in various fields, including marketing ([10]) or robotics ([11,12]), brings many benefits to end users. The most common application of artificial empathy can be found in medicine [13–15]. Another application of artificial empathy is in computer games. The ability of AI systems to emulate empathy can help create more realistic characters and environments that allow players to feel part of the game world. Artificial empathy systems are also used in games to increase players' empathy toward certain social groups or characters in the game. This allows players to better understand and identify with the characters, which in turn can influence their in-game decisions [16].

In the paper, we will deal with fuzzy agents, i.e., agents that decide with fuzzy knowledge [17]. The decision is whether an egoistic or empathetic action should be taken to achieve a shared goal. Egoistic action is focused on a local goal, compatible with the general goal of the swarm. Empathetic action is cooperative. The problem at hand is whether the swarm performance can be further optimized by integrating socio-psychological aspects of empathy in swarm control.

Agent actions are based on generalized knowledge, represented as a table of a fixed number of fuzzy sets, modeling action–consequence pairs. Fuzzy feature representation converts the collected information into membership degrees [18]. Resulting degrees are taken as a description of the agent's state. Before an agent performs an action, its current state is assigned a reward, based on the similarity to known states and their outcomes. After the performed action, the state is assigned a realized reward, and is then remembered as "knowledge". By replacing each value in the input data with their corresponding degree of membership to parameter realization we obtain a fuzzified set of labeled states [19].

Since empathy requires comparing the states of the agents, it is important to introduce similarity measures for fuzzy sets. From [20,21], we can ascertain that fuzzy sets are the proper tool to describe the similarity in such a situation. An agent will compare known, "experienced" states to a new one, for which the reward is not yet known or stored in any knowledge base available for the agent. The empathetic approach has great potential for

use in swarms, as there are numerous opportunities for one agent to "help" another in achieving a shared goal.

1.3. Structure of the Paper

Simulation of swarm behavior is a crucial step before implementation on a target platform. However, simulations are often inaccurate and unrealistic, particularly for the empathetic swarm, due to the critical role of communication. In software simulations, it is not possible to reliably reflect problems or errors in signal transmission or hardware failures. This paper proposes a new approach to simulating empathetic behaviors in swarms using an open-source physical-based experimentation platform (OPEP). OPEP serves as an intermediate solution between software-based simulation and the target environment. The paper presents the architecture and current implementation of the platform.

The main purpose of this paper is twofold. The first one is to introduce a novel model of swarm control with the use of artificial empathy. The second is to establish an open-source physical-based experimentation platform, a cost-effective and reliable way to evaluate various models and scenarios in robotics, focused on modeling emphatic behaviors.

The paper is organized as follows: Section 2 presents materials and methods that include definitions of artificial empathy and its theoretical models. It also briefly reviews the available methods for conducting experiments in a swarm of robots. Section 3 contains the main results and introduces the idea of an empathetic swarm, presents simulation results of the proposed swarm model, and proposes the open-source physical-based experimentation platform. Section 4 concludes the obtained results and gives some ideas for further research.

2. Materials and Methods

2.1. Empathy Theory

Artificial empathy is the ability of computational systems to understand and respond to the thoughts, feelings, and emotions of humans [8]. Most definitions of artificial empathy describe it as the artificial version of human empathy [22]. Human empathy is said to consist of three components: emotional, cognitive, and motivational. Emotional and motivational empathy is more biological and allows for "automatic" responses to emotions elicited by internal or external factors. Cognitive empathy is more inductive—it allows an agent to understand the inner state of another agent, based on the signals they broadcast (e.g., expressions), and the situation they are in.

It is easy to notice that in swarms emotional empathy has no significant applications (for now, since robots do not yet commonly present or feel emotions [23]). Yet, there is a vast scope of possible applications of cognitive empathy, since cognitive empathy is connected with learning and deducing about behaviors in a certain environment. Knowledge sharing by cognitive empathy can help a swarm of robots learn effective behaviors faster.

In the literature, we can find three main models of artificial empathy, each of them created for different use cases.

1. Emotional and cognitive empathy model: The model was developed from medical and neuroscientific research of the human brain and has its justification in the brain structure. It assumes that empathy can be divided into parts: (1). responsible for recognizing and reacting to emotions; (2). a part responsible for cognitive, more logical, and deductive mechanisms of understanding the inner states of others [24].
2. Russian doll model: The model assumes that empathy is learned during human life—it resembles a Russian doll, with layers of different levels of understanding others. The first, most inner layers are mimicry and automatic emotional reactions, the next layers are understanding others' feelings and the outer layers are taking the perspective of others, sympathizing, and experiencing schadenfreude [25].
3. Multi-dimensional model: This model assumes that we have four dimensions of empathy—antecedents, processes, interpersonal outcomes, and intrapersonal outcomes. Antecedents encompass the agent's characteristics: biological capacities, learning history, and situation. Processes produce empathetic behaviors: non-cooperative

mechanisms, simple cognitive mechanisms, and advanced cognitive mechanisms. Intrapersonal outcomes are to resonate or not with the empathy target, and interpersonal outcomes are relationship related [26].

Artificial empathy has found its applications in a growing number of fields and practical problems:

- Medical, e.g., in the detection of autism or Parkinson's disease, depression treatment [13,15,27]
- Psychology, e.g., ethics, artificial companions [14,28];
- Marketing, e.g., personalized adverts [29];
- Entertainment e.g., games with high immersion or biofeedback [16,30,31];
- Education, e.g., empathetic tutors [32,33];
- Car industry [34].

Swarm applications also seem to open up an opportunity to effectively utilize empathy.

The idea of 'swarm' comes from nature—insects, animals, and people work together in cooperative groups to achieve goals that would be impossible to reach for a single agent. Examples are ants, bees, fish, etc.

The main characteristics of a swarm are a large number of agents, the absence of external control, and simple behaviors exhibited by individual agents. More advanced and complex swarm behaviors emerge automatically, based on environmental conditions and the capabilities of the agents.

One of the most important research areas of swarms is collective decision-making [35]. The problem has also an application in empathetic swarms, where the decision of which agent to help can be crucial in achieving the goal. An example can be found in [36].

Current research of empathetic swarm behaviors emerging automatically from broadcast and received signals include [37,38]. Also, the theory of mind is used to model collective behaviors of artificially modeled rescue teams [39,40]. The study by Huang et al. [41] considers a simulated swarm of a few caribou agents to escape from a superior wolf agent. The introduction of empathy in the form of an additional parameter (the distance from the chased caribou to the wolf), allowed for a significant increase in the number of learned successful escape strategies. This kind of approach is very limited—the decision process is automatic and based only on one parameter. It resembles more of an emotional contagion (one of the primitive levels of empathy), rather than a cognitive empathetic process. In our model, we would like to emphasize the role of learning, deduction, and experience in empathizing with other agents.

2.2. Available Experimental Environments

An analysis of various approaches and tools for conducting robot swarm experiments was conducted. The three main categories of available experimental environments are described below.

1. Stand-alone robots, allowing the construction and modeling of swarm behavior.
    - Kilobot [3]: This is a swarm-adapted robot with a diameter of 3.3 cm, developed in 2010 at Harvard University. It operates in a swarm of up to a thousand copies, carrying out user-programmed commands. The total cost of Kilobot parts was less than USD 15. Kilobots move in a vibration-based manner. In addition, they are capable of recognizing light intensity, communicating, and measuring the distance to nearby units. Currently, the project is not under active development, but it is still popular among researchers.
    - e-puck2 [42]: This is a 7 cm diameter mini mobile robot developed in 2018 at the Swiss Federal Institute of Technology in Lausanne. It supports Wi-Fi and USB connectivity. It has numerous sensors, including IR proximity, sound, IMU, distance sensor, and a camera. The project is being developed using open-source and open-hardware principles.

- MONA [43]: This is an open-hardware/open-source swarm research robotic platform developed in 2017 at the University of Manchester. MONA is a small, round robot with a diameter of 8 cm, equipped with 5 IR transmitters, based on Arduino architecture.
- Colias [44]: This is an inexpensive 4 cm diameter micro-robot for swarm simulation, developed in 2012 at the University of Lincoln. Long-range infrared modules with adjustable output power allow the robot to communicate with its immediate neighbors at a range of 0.5 cm to 2 m. The robot has two boards—an upper board responsible for high-level functions (such as communication), and a lower board for low-level functions such as power management and motion control.
- SwarmUS [45]: This is a project that helps create swarms of mobile robots using existing devices. It is a generic software platform that allows researchers and robotics enthusiasts to easily deploy code in their robots. SwarmUS provides the basic infrastructure needed for robots to form a swarm: a decentralized communication stack and a localization module that helps robots locate each other without the need for a common reference. The project is not in development as of 2021.

2. Robot simulation software
    - AWS Robomaker: This is a cloud-based simulation service released in 2018 by Amazon, allowing robotics developers to run, scale, and automate simulations without the need to manage any infrastructure. It enables the creation of user-defined, random 3D environments. Using the simulation service, one can speed up application testing and create hundreds of new worlds based on templates that one defines.
    - CoppeliaSim [1]: This is a robotics simulator with an integrated development environment; it is based on the concept of distributed control. Each object/model can be individually controlled using a built-in script, plug-in, ROS node, remote API client, or another custom solution. This makes it versatile and ideal for multi-robot modeling applications. It is used for rapid algorithm development, simulation automation of complex processes, rapid prototyping and verification, and robotics-related education.
    - EyeSim [46]: This is a virtual reality mobile robot simulator based on the Unity engine, which is able to simulate all the main functions of RoBIOS-7. Users can build custom 3D simulation environments, place any number of robots, and add custom objects to the simulation. Thanks to Unity's physics engine, robot motion simulations are highly realistic. Users can also add bugs to the simulation, using built-in simulated bug functions.
3. Comprehensive services including simulator and hardware platform.
    - AWS DeepRacer: This is a 1/18 scale fully autonomous racing car designed in 2017 by Amazon and controlled by Reinforcement Learning algorithms. It offers a graphical user interface that can be used to train the model and evaluate its performance in a simulator. AWS DeepRacer, on the other hand, is a Wi-Fi-enabled physical vehicle that can drive autonomously on a physical track using a model created in simulations.
    - Kilogrid [47]: This is an open-source Kilobot robot virtualization and tracking environment. It was designed in 2016 at the Free University of Brussels to extend Kilobot's sensorimotor capabilities, simplify the task of collecting data during experiments, and provide researchers with a tool to precisely control the experiment's configuration and parameters. Kilogrid leverages the robot's infrared communication capabilities to provide a reconfigurable environment. In addition, Kilogrid enables researchers to automatically collect data during an experiment, simplifying the design of collective behavior and its analysis.

2.3. Fuzzy Sets and Their Similarity

In the presented paper, the agent's states will be represented in the form of fuzzy sets. Fuzzy sets allow us to represent precise data, and since agents will communicate using non-ideal communication and will interpret the signals subjectively, fuzzy sets seem to be a proper tool to use.

Let $U = \{u_1, u_2, \ldots, u_n\}$ be a crisp universal set. A mapping $A\colon U \to [0,1]$ is called a fuzzy set (FS) in U (denoted also by capital letters A, B, ...). For each $1 \leq i \leq n$, the value $A(u_i)$ (A_i for short) represents the membership grade of u_i in A. We say that fuzzy set A is a subset of fuzzy set B ($A \subset B$) if $A(u_i) \leq B(u_i)$ for all $u_i \in U$. Let $\mathcal{F}(U)$ be the family of all fuzzy sets in U.

A similarity measure of fuzzy sets is defined as a function $s\colon \mathcal{F}(U) \times \mathcal{F}(U) \to [0,1]$ such that

(T1) for each $(A, B) \in \mathcal{F}(U) \times \mathcal{F}(U)$, we have $s(A, B) = s(B, A)$,

(T2) for each $(A, D) \in \mathcal{F}(U) \times \mathcal{F}(U)$ and $(B, C) \in \mathcal{F}(U) \times \mathcal{F}(U)$ such that $A \subset B \subset C \subset D$ we have

$$s(A, D) \leq s(B, C), \tag{1}$$

(T3) for each $X \subset U$ such that $(\mathbb{1}_X, \mathbb{1}_{X^c}) \in \mathcal{F}(U) \times \mathcal{F}(U)$ we have $s(\mathbb{1}_X, \mathbb{1}_{X^c}) = 0$ and $s(\mathbb{1}_X, \mathbb{1}_X) = 1$.

This definition coincides with the classical one proposed by Xuecheng [48] (see also [49–51]). The higher measure values indicate a higher similarity of its arguments. It is usually assumed that all fuzzy sets are comparable by a given similarity measure. However, some similarity measures cannot be formally defined over the whole Cartesian product $\mathcal{F}(U) \times \mathcal{F}(U)$. The most commonly used similarity measure is the Jaccard index, defined as

$$s(A, B) = \frac{A \cap B}{A \cup B}. \tag{2}$$

3. Results

3.1. Artificial Empathy of a Swarm

The general idea of artificial empathy in swarm applications comes from observing cooperative behavior in a group of agents (humans, animals) and realizing what types of knowledge and experience are needed to create successful behaviors and strategies.

In human cooperation, it is easy to see that taking the perspective of another person and trying to understand their point of view greatly improves cooperation and leads to better results. The two actions mentioned actually define cognitive empathy, i.e., drawing one's own conclusions from the state broadcasted by another agent and the environment, thus anticipating the target's behaviors based on inferred knowledge. The proposed model of an artificially empathetic swarm is based on how humans cooperate. People collect experience and knowledge and decide what action to take based on it. While cooperating, they take into account the experience and knowledge of others. Yet, it is impossible to access other people's minds directly—one has to interpret signals sent from cooperating partners. These signals constitute largely of emotions [5]. After receiving a signal, the empathizer imagines what knowledge the signal represents, and what could be the possible consequences [6]. Based on the person's own knowledge, they envision the state of the sender. Finally, an action choice is made—whether to use one's own knowledge and capabilities or to combine efforts with others to improve performance.

The proposed model consists of six parts: the module for evaluating egoistic behaviors, the module for evaluating artificially empathetic behaviors, the memory module for storing knowledge, the decision module for choosing the action type, and egoistic and artificially empathetic behavior controllers for choosing and executing particular actions. Modules are described in the following subsections and interactions between them are depicted in Figure 1.

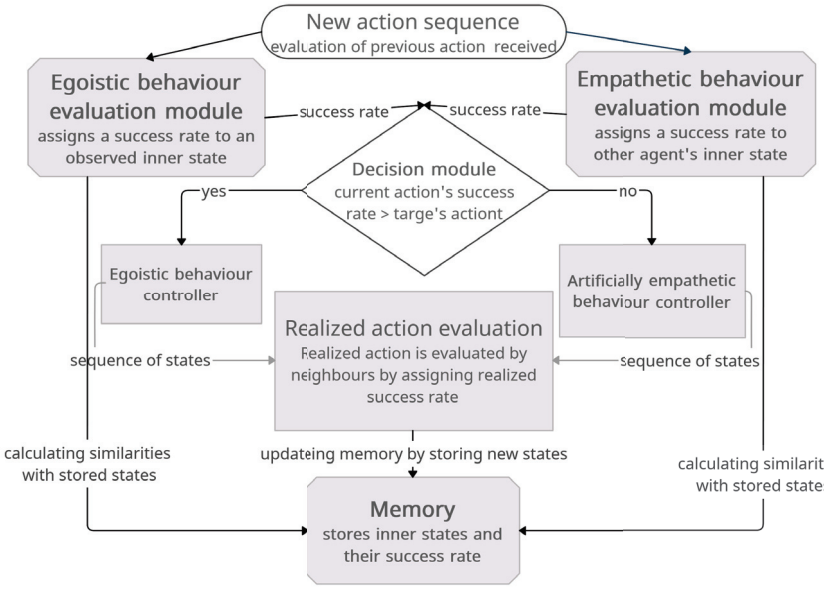


Figure 1. Swarm behavior control system with artificial empathy. Schema presents component modules for egoistic and empathetic control and behavior evaluations, decisions, and memories.

3.1.1. Egoistic Behavior Evaluation Module

The egoistic behavior evaluation module assigns a reward to an egoistic action, based on the state vector representing the state of the agent and perceived environment, and the knowledge of the agent. This state is defined as a fuzzy set:

$$A = (a_1, a_2, \ldots, a_n)\,, \tag{3}$$

where $a_i \in [0,1]$, $i = 1, \ldots, n$ are membership values, representing the satisfaction level of a ith state variable. All a_i should not be correlated. The state should be updated in time. The state of the agent, which includes the perceived environment and the knowledge of the agent, is composed of two parts: emotional state (subjective knowledge) and cognitive state (objective facts). The first group may include information about the agent's internal state (battery level, current action, call for help) and the second one may include information about the perceived environment (proximity to the goal, neighbors, obstacles).

Each of the states is evaluated to decide whether it can bring the agent closer to the goal. The state A^i is assigned a reward $\mathrm{r}(A^i)$ based on the similarity to the known states and their outcomes, stored in memory and representing the agent's knowledge. The agent has an initial set of states and their rewards, to be able to generalize in an unknown environment. Those first states can be understood as instincts or basic knowledge. The reward from the evaluated state A^i is then calculated as

$$\mathrm{r}(A^i) = \frac{\sum_{j=1}^{m} \mathrm{s}(A^j, A^i) \cdot \mathrm{r}(A^j)}{m}\,, \tag{4}$$

where m is the number of states and the rewards $\left(A^j, \mathrm{r}_j\right)$ stored in memory and s is a fuzzy similarity measure.

3.1.2. Artificially Empathetic Behavior Evaluation Module

The target of empathy broadcasts its state, which the empathizing agent interprets based on their own knowledge. In our model, the empathizing agent (A) evaluates the target's (B) state by comparing its broadcast state to the agent's own knowledge. Reward

$\mathrm{r}(B)$ is calculated using the same knowledge as the agent's rewards according to (4). The rewards calculated for both the empathizing agent and the empathy target are used as decision parameters in choosing the agent's next action.

3.1.3. Memory Module

The memory module is responsible for storing the agent's knowledge, obtained by performing actions. It is composed of state–reward pairs (A^i, r_i). In the beginning, the agent only has initial states. With time and performed actions, new knowledge is added to this base. Namely, each action taken by the agent is evaluated and assigned a realized reward. The state representing the action and the corresponding reward are then added to the database and can be compared with other, new states to calculate similarity and assign rewards. Since the platforms used to implement the model have finite resources, it is assumed that the memory module stores only a limited number of states. To reduce redundancy, only clusters of state–reward pairs are stored. Any clustering strategy may be used, including k-means [52]. Since the decision process is based on similarity, this method maintains its generality and allows for the omission of the problem of "perfect knowledge", i.e., remembering every detail of an action (since instead of all details, only action representatives are being stored).

This method of storing information also allows for updating the state/reward pairs. If a state is remembered and it appears again with a different reward, the clustering algorithm can recalculate the representative of the action, with a different reward.

3.1.4. Decision Making

The decision module is responsible for choosing between the controllers that will provide the next action: the egoistic or empathetic behavior controller. The first one provides egoistic actions that bring the agent closer to the local goal. The second provides cooperative actions with the same goal but possibly synergistic results. The choice is made based on the reward assigned to the current state of the agent in the egoistic behavior evaluation and, if a signal is received, it considers the reward of empathy target B, calculated by the artificially empathetic behavior evaluation module. The module performs the following comparison:

$$\mathrm{r}(A) \geq \mathrm{r}(B) \tag{5}$$

where A is the agent's current state and B is the empathy target's broadcast state. If the $\mathrm{r}(A)$ value is higher than $\mathrm{r}(B)$, the egoistic behavior of the agent has a greater chance of success in achieving the goal than stopping the current action and helping the neighboring agent. If the contrary is true, acting with empathy may result in a greater chance of success, so the current course of action should be dropped. In the case of multiple incoming signals, only the first one is considered.

3.1.5. Learning

After a full action sequence is performed (i.e., a set of actions resulting in a particular change in environment), the realized action is evaluated. Each agent's actions are evaluated, based on signals from its neighbors. The sequence is defined as a series of atomic actions that begin with a starting action (the first one without a realized reward) and ends with a last action before an evaluation signal is received:

$$seq_i = (A^i_1, A^i_2, \ldots, A^i_k) \tag{6}$$

where A^i_j, $j = 1, \ldots, k$ is the state vector and k is the number of actions performed before the evaluation signal is received. The realized reward $\hat{\mathrm{r}}_i$ is the first evaluation signal received by the agent, weighted according to the set goal:

$$\hat{\mathrm{r}}_i = \hat{A} \cdot W \tag{7}$$

where $\hat{A}$ is the received evaluation signal and the weights W are assigned according to the defined agent's goal. The reward is then assigned to the action process vector, forming new knowledge:

$$\mathrm{ev}_{\mathrm{seq}_i} = (A_1^i, A_2^i, \ldots, A_k^i, \hat{\mathrm{r}}_i) \tag{8}$$

The aggregated action and the realized reward are then stored in the memory module as newly learned behavior. Given the current action, we find the closest matching action sequence in the knowledge base. Then, if it is similar enough to the current one, we replace the original sequence with its strengthened version. The behavior formulated in this way allows to strengthen the currently existing rules in the knowledge base. If the closest matching action sequence is too far from the current one, a new action sequence is formed.

3.2. Simulations

3.2.1. Problem Description

Let us consider the warehouse that stores grain. The stacks of grain often change places due to normal warehouse operations. The warehouse is infested with rats. We introduce a swarm of empathetic guarding robots that collaborate in order to detect rats and find their nests. The robots operate without a central control system but can communicate between themselves. They patrol the dynamic environment. When a rat is spotted, the robot broadcasts a signal containing information about the spotted target, its own chances of success in chasing it, and other information, like the battery level. Robots that receive that signal calculate whether it is better to continue the current action (e.g., patrolling another part of the warehouse, going to the charging station) or to approach the broadcasting robot and assist in chasing the rat. The described environment is an enclosed space with a static obstacle in the form of grain, defined walls, and mobile hostile objects (rats). The robot's task is to detect and follow the rat, engaging in group encirclement. The rat's goal is to reach the grain. Different views on virtual experimentation environments are given in Figure 2.

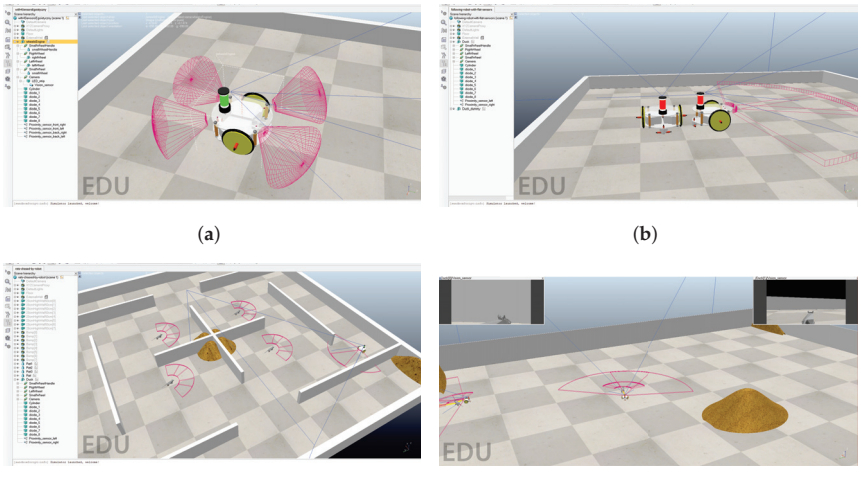
The analysis of robot behavior regarding the influence of artificial empathy was conducted only on chasing robots, as they had the most possible actions to perform and could process the most information among all the robots. In physical experiments, rats were also represented by robots, but they were not equipped with empathetic modules.

Patrolling robots could perform their tasks individually or through communication with other robots. Each robot could signal its own state through an LED strip, displaying information such as the robot class or the currently performed action. This included:

- Call for help;
- Encircling the rat;
- Helping;
- Another robot nearby;
- Rat nearby.

In contrast to the control group, where robots were not equipped with empathetic modules, experiments on empathetic robots indicate that robots have much more information to process before taking specific actions. Similar to the control group, the rat is searched for in the camera image. The empathetic model difference lies in the fact that each patrolling robot additionally signals information about its state and surroundings on the LED strip. It also has the ability to analyze this information from other robots.

This enables robots to make decisions based not only on their own observations but also on those collected from the surrounding environment. Before taking any action, the robot calculates the reward for performing a specific action, i.e., how much it contributes to achieving the global goal. Rewards are calculated for both the currently performed action and the planned action. If the reward for the new action is greater than the currently performed action, the robot interrupts it and starts a new one. Using the artificial empathy module, robots could make decisions that were optimal for the entire group.



(**a**) (**b**)

(**c**) (**d**)

Figure 2. Empathetic swarm simulation in CoppeliaSim. (**a**) Visualization of the agent; (**b**) visual communication; (**c**) warehouse example; (**d**) camera view.

The performed experiments considered the following list of scenarios:

- Detection of a rat in the warehouse—solitary pursuit.
  - Robot 1 patrols the warehouse;
  - Robot 1 notices a rat;
  - Robot 1 starts chasing the rat;
  - Robot 1 catches the rat, meaning it approaches the rat to a certain distance.
- Detection of a rat in the warehouse—pursuit handover.
  - Robot 1 patrols the warehouse;
  - Robot 1 notices a rat in the adjacent area;
  - Robot 1 lights up the appropriate color on the LED tower to inform Robot 2 that there is a rat in Robot 2's area;
  - Robot 2, noticing the appropriate LED color, starts chasing the rat;
  - Robot 2 catches the rat, meaning it approaches the rat to a certain distance.
- Detection of a rat in the warehouse—collaboration.
  - Robot 1 patrols the warehouse;
  - Robot 1 notices a rat;
  - Robot 1 starts chasing the rat;
  - The rat goes beyond Robot 1's patrol area;
  - Robot 1 lights up the appropriate color on the LED tower to inform Robot 2 that the rat entered its area;
  - Robot 2, noticing the appropriate LED color, continues chasing the rat;
  - Robot 2 catches the rat, meaning it approaches the rat to a certain distance.
- Change of grain color.
  - Robot 1 patrols the warehouse;
  - Robot 1 notices that the grain color is different than it should be;
  - Robot 1 records the event in a report;
  - Robot 1 continues patrolling.
- Change of grain color—uncertainty.
  - Robot 1 patrols the warehouse;
  - Robot 1 notices that the grain color is possibly different than it should be—uncertain information;

  - Robot 1 lights up the appropriate color on the LED tower;
  - Robot 2, noticing the appropriate LED color, expresses a willingness to help and approaches Robot 1;
  - Robot 2 from the adjacent area checks the grain color and confirms or denies Robot 1's decision;
  - Robot 1 records the event in a report if confirmed by Robot 2;
  - Robot 2 from the adjacent area returns and continues patrolling;
  - Robot 1 also continues patrolling.
- Weak battery.
  - Robot 1 has a weak battery;
  - Robot 1 lights up the appropriate color on the LED tower, expressing a desire to recharge its battery;
  - Robot 2, noticing the appropriate LED color, agrees to let Robot 1 recharge the battery;
  - Robot 1 goes to recharge;
  - Robot 2 additionally takes over Robot 1's area for patrolling.
- Exchange of patrol zones.
  - Robot 1 passes through its patrol area several times without any events;
  - Robot 1 lights up the appropriate color on the LED tower, expressing a desire to exchange the patrol area;
  - Robot 2, noticing the appropriate LED color, expresses a desire to exchange the patrol area;
  - Robot 1 and Robot 2 exchange patrol areas.

3.2.2. Implementation

The simulations were performed in CoppeliaSim (V4.4.0). Up to ten robots, equipped with virtual cameras, LED communication, and touch sensors were to detect and chase four rats in a synthetic warehouse environment. Robots broadcast state signals to other agents, which receive them via camera and decide whether to take egoistic or empathetic action.

The YOLOv2 real-time object detection system [53] was used to detect objects in the camera images, and the VGG16 [54] convolutional neural network was used to determine the status of other robots (sent via LED strip). All simulation scripts were implemented in Lua (internal CoppeliaSim scripting) and Python (external backend service).

All the fuzzy descriptions like "far" or "long" are modeled with linguistic variables and terms—the value of a parameter is actually the value of a membership function for each of the considered terms.

Vectors of parameters in Tables 1 and 2 are used as initial knowledge. Each new state that arises is compared to those, and the similarity is calculated to decide if the new state has a chance of success or not. Here, we use the similarity measure:

$$\mathrm{s}\left(A^j, A^i\right) = 1 - \frac{\sqrt{\sum_{k=1}^{n} |x_k^j - x_k^i|^2}}{n}. \tag{9}$$

Table 1. Parameters describing the state of the agent.

Name	Sym	Description of Boundary Values
others close	a	1 many other agents in the vicinity, 0 for none
in touch	n	1 for long contact time, 0 for none
long search	t	1 for the long duration of the current search, 0 for not searching
calling for help	c	1 for calling for a long time, 0 for not calling
neutralized	e	1 if "I am inactive" signal was received from the newly inactive agent; 0 if not

Table 1. *Cont.*

Name	Sym	Description of Boundary Values
close to neighbor	d	1 if the distance to the neighbor is 0; 0 if the distance to the neighbor is far
target at right	p	1 for the agent at the immediate right, 0 for the agent not in sight
target at left	l	1 for the agent at the immediate left, 0 for the agent not in sight
fully charged	f	1 for the fully charged robot, 0 for not charged
helping	h	1 for the long duration of helping, 0 for not helping
reward	s_i	describes the chance of success of the current action sequence

Table 2. Agent's initial knowledge—states and rewards.

Parameter	x_1	x_2	x_3	x_4	x_5	x_6
a	0.5	1.0	0.5	0.5	0.0	0.1
n	0.5	0.5	0.5	0.0	1.0	1.0
t	0.5	0.5	0.5	0.5	0.5	0.5
c	1.0	1.0	1.0	0.5	1.0	0.5
d	0.5	1.0	0.5	0.5	0.0	0.1
p	0.5	0.5	0.5	0.5	0.5	0.5
l	0.5	0.5	0.5	0.5	0.5	0.5
f	0.5	0.5	0.5	0.5	0.5	0.5
h	0.5	0.5	0.5	1.0	0.5	1.0
r_i	1.0	1.0	1.0	1.0	0.0	0.0

3.2.3. Simulation Results

Robots cooperate in order to detect and chase the rats, and empathetic behaviors are visible. Due to the limitations of CoppeliaSim, mainly the lack of repeatability and poor performance when using virtual cameras, simulations did not allow for a reliable comparison between egoistic and empathetic behaviors. These problems lead directly to the OPEP project. OPEP allows for the inclusion of such factors as acceleration, friction, light intensity, and reflections, while maintaining high control over the environment and experiment course.

During the simulation, the time in which the robots achieve the global goal, i.e., detecting and catching all rats, was measured. For each case, empathic and non-empathic models, 10 experiments were conducted, measuring the time to achieve the global goal. An important aspect was that objects in the arena were randomly distributed each time to ensure diversity in the observed behaviors.

After conducting experiments on a swarm of five robots, it was decided to double the number of objects in the scene. This change introduced more opportunities for interactions between individual units, and the simulation could proceed differently. Additionally, the larger the group of patrolling robots, the more the positive impact of empathic behaviors can be observed, allowing for a focus on the analysis of behaviors between neighboring objects.

As in previous experiments, objects before each simulation were randomly distributed, and the simulation ended when the global goal was achieved, i.e., when all rats were detected and surrounded.

In this way, a total of 40 experiments were conducted in 4 variants. This material was further analyzed, with a primary focus on the analysis of model behaviors using artificial empathy and those without it. In many cases, the empathic model recorded lower times to achieve the global goal. However, the differences are small, with the effectiveness of empathy being the most visible in larger groups. In such situations, the true power of unit cooperation, forming the entire swarm, can be observed.

An interesting phenomenon was the significant differences in times between individual simulations. The shortest simulation time for five patrolling robots was only 58 s, while the longest was as much as 116 s, nearly a twofold difference. The average simulation time in the egocentric model was 84.1 s, and in the empathy-utilizing model, it was 81.6 s.

Summing up all experiments in this section, the empathetic swarm of robots performed its tasks, on average, 2.5 s faster. For 10 patrolling robots, the fastest achievement of the goal occurred after 88 s, while the longest took 205 s. In this case, there were many more possible interaction scenarios for 10 robots, influencing the disparities in simulation times. The average neutralization time for all viruses in the egocentric model was 148.5 s, while for the empathetic model, it was 137.3 s. The significant positive impact of using the artificial empathy module is evident, with a difference of 11.2 s, confirming the effectiveness of the empathetic model. Unfortunately, a more accurate and reliable statistical analysis is not possible. This is due to the huge discrepancies between repetitions and the influence of many uncontrolled random factors resulting from the CoppeliaSim simulator on the experiment.

We prepared a few visualizations of the proposed empathetic model, along with a comparison with the egoistic one (videos available on https://github.com/open-pep/coppelia-simulations accessed on 13 December 2023).

1. Egoistic, two rats. Shortly after starting a patrol, both robots spot the same rat and start chasing it. Meanwhile, the second rat destroys the grain located in the middle of the arena. After neutralizing the first rat, one of the robots begins chasing the second pest.
2. Empathetic, two rats. The robot on the right spots a rat and signals it with an LED strip. The second robot, noticing this, continues to patrol the surroundings in search of other pests. After a while, it detects the second rat and starts following it. As a result, both rats are neutralized and grain loss is reduced.
3. Egoistic, robots run out of battery. Robots detect the same rat. During the chase, the robots interfere with each other, making it difficult to follow and neutralize the rat. Eventually, the rat is neutralized, but before the robots can spot and begin their pursuit of the other pest, both of them run out of battery and the second rat escapes.
4. Empathetic, low battery help. The robot on the right starts chasing the detected rat. During this action, the agent signals with an LED strip that it needs assistance, due to a low battery level. The other robot notices this and decides to help to catch the weaker rat. After neutralizing it, the second robot starts searching for other pests.

Two initial visualizations are summarized in Figure 3. The movement trajectories of all agents are shown by a dashed and dotted lines. It can be seen that in the egoistic variant, both robots undertake the pursuit of the same rat. While in the empathetic variant, thanks to communication, the robots started a chase after both rats, so that the grain resources were not damaged.

3.3. Open-Source Physical-Based Experimentation Platform

In the field of robotics, experiments are crucial for the development and verification of new algorithms and technologies. However, conducting experiments in this area is challenging, costly, and comes with a range of problems.

Simulations—one of the ways to conduct experiments—are often simplified and inaccurate due to the multitude of parameters that need to be considered. On the other hand, accurate simulations are very time-consuming. It is difficult to include all parameters in the simulation, as some may be unidentified or challenging to model. CoppeliaSim is one tool used for simulating the motion of robot swarms, but it has significant limitations and does not consider all parameters of the real environment, leading to unrealistic simulation results. Time-consuming robot swarm simulations also pose a problem, making it difficult to test various scenarios and restricting frequent algorithm changes.

Real-world experiments, on the other hand, are costly, requiring the purchase of components and the creation of physical experimental platforms, which is time-consuming. However, experiments using dedicated hardware lack high repeatability and reproducibility of results, making it challenging to compare models experimentally. Additionally, there is a lack of standards and a unified approach to the design and implementation of mobile robots, requiring individualized approaches for each experiment and resulting in significant time delays and increased costs. These issues exacerbate discrepancies between simulations and experiments.

Therefore, in this work, we propose the concept of the open-source physical-based experimentation platform (OPEP), which is an intermediate environment between simulators and full experiments on dedicated equipment. In the following, we will present general assumptions toward the offered functionalities, the architecture, and the implementation of two versions of prototypes of the postulated solution.

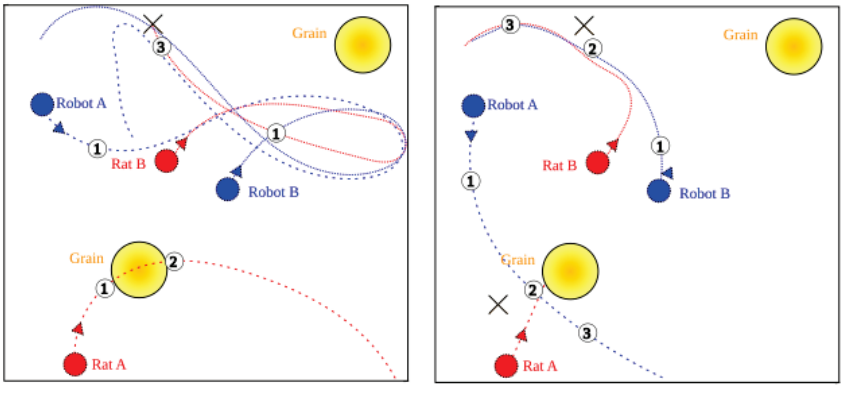


Figure 3. The movement trajectories of all agents for egoistic (**left**) and empathetic (**right**) visualisations. The numbered points in the figure depict points in time. (**Left**) (1) Both robots spot Rat B while Rat A starts to damage grain. (2) Rat A starts searching for a new target. (3) Both robots finally neutralize Rat B. (**Right**) (1) Both robots spot rats, due to cooperation, they start to chase different targets. (2) Both robots start neutralizing rats. (3) After neutralizing the rats, both robots start to patrol the area.

3.3.1. Proposed Platform Features and Architecture

The open-source physical-based experimentation platform (OPEP) consists of several parts—a swarm of autonomous, mobile robots, an arena for controlled experiments, charging stations, an overhead controller (camera and mini PC), and a web application for remote experiment control, as depicted in Figure 4. The platform allows for performing experiments on physical robots remotely, in a controlled environment. It stands as a middle step between fallible and imperfect simulations and costly physical implementation.

The proposed experimentation platform contributes to the creation of a completely new product: a universal, remote service for experimenting with and testing artificial intelligence algorithms on a hardware swarm of robots, provided in a cloud computing model. Its main features are outlined in the following paragraphs.

1. Comprehensive support for swarm design process using hardware platform. This feature corresponds to the need to verify AI algorithms in a hardware environment, including early-stage development, consideration of environmental parameters that are unavailable in simulations, and the ability to study algorithms, considering variable environments and interactions. In this area, there are two alternatives: comprehensive algorithm evaluation (Kilogrid + Kilobots, DeepRacer) and simulation software (CoppeliaSim v4+, DynaVizXMR, EyeSim v1.5+, Microsoft Robotics). Alternative solutions only support the design process in simulated environments or require significant financial investments for prototyping, limiting accessibility in early development stages.
2. Low cost of building and size of swarm robots. This feature corresponds to the need to evaluate complex behaviors and the latest AI algorithms in a large swarm of robots, considering the requirements for low cost and easy availability of solutions. Alternatives include miniature robots like Kilobots and mini sumo robots. Those solutions are expensive, with costs often including additional resources and services. Additionally, computational power drastically decreases with the robot's size, limiting capabilities such as running a vision system.

3. Remote programming of robots. This feature addresses the need to share research/educational infrastructure without physical access, fostering interdisciplinary and international research collaborations. Alternatives include cloud-based robot simulators like AWS Robomaker and DeepRacer. In competitive solutions, this functionality is only available in simulations or limited environments and specific research areas.
4. Standardization and Scalability of experimental environment. This feature corresponds to the need to adapt and expand the experimental platform to different projects while maintaining standardization for experiment repeatability and reproducibility, facilitating comparisons across research centers. Alternatives include open-source software and hardware projects like SwarmUS, Kilobots, colias.robot, as well as simulation software. They lack the ability to expand robot software in any way using high-level languages. Moreover, existing solutions are not designed for result repeatability (e.g., randomness in Kilobots' movements).
5. Open specification and hardware. This feature corresponds to the need for independently building a complete experimental platform. Current solutions include open-source software and hardware projects, such as SwarmUS, Kilobots, and colias.robot. Most competitive solutions are closed, and open solutions often have limited computational resources.

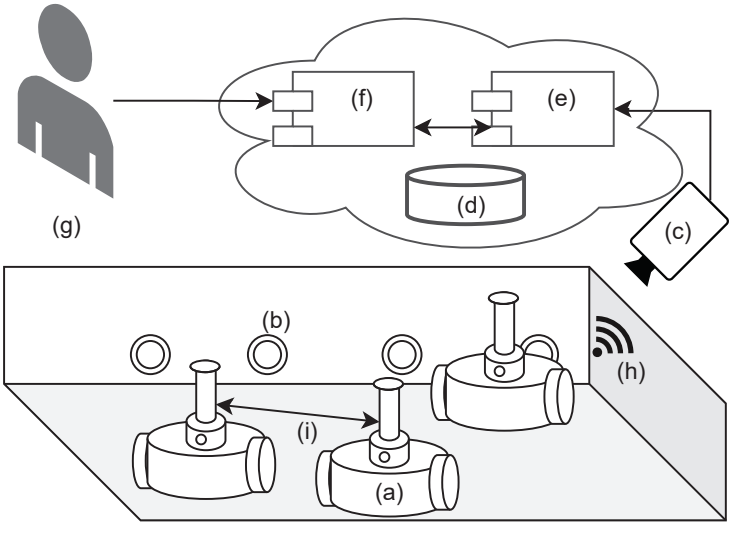


Figure 4. Context architectural diagram for OPEP. (a) Independent robots that form a swarm; (b) automatic charging stations; (c) overhead controller (camera + miniPC); (d) data store; (e) experiment control API module; (f) simulation web API; (g) researcher interacts with the system via a web browser; (h) wireless communication with overhead (maintenance only); (i) visual communication between robots.

3.3.2. Platform Implementation

First Prototype

The authors conducted a series of analyses and experiments, creating a swarm of 8 early prototypes of robots equipped with mobility, vision, and communication systems, enabling the realization of simple empathetic behaviors. The prototype robots had a diameter of 14.9 cm and a height of 15 cm, and were equipped with a 4000 mAh Li-Poly battery, allowing about 40 min of continuous operation (Figures 5 and 6). Based on team member experiences during the creation of the prototype robot swarm, several phenomena and issues crucial for proper robot operation in a real environment were observed. These would likely be overlooked in computer simulations, including mirror reflections of robots

in the arena walls, slipping robot wheels, uneven traction, imperfections in mechanical components (e.g., motors, gears), and the impact of the environment on the performance of the vision system (too strong or weak room lighting).

The first versions of robots were built with the use of Raspberry Pi Zero 2 W microcomputers that took care of robot control, vision, and all decision-making. Agents used two motors to power the wheels. The communication was performed via a custom 360° RGB LED strip, placed on a rotating turret, which also held the OV5647 5MPx camera, which gave an effective 240° angle of view. The use of 8 RGB LED lights allowed expressing about 2 M distinguishable inner states. The communication was imperfect since robots had to detect the signal that could be disturbed by light level, reflection, other robots in view, etc. The outer case was 3D-printed and was designed for minimal collision damage.

Each robot in the swarm runs on 64-bit Raspberry Pi OS Lite (5.15 kernel). The vision system is implemented using *picamera2* (0.1.1) and *opencv-python* (4.6.0.66) libraries. Intra-process communication is handled using Redis (7.0.5) pub/sub feature. Web-based user interface is still being actively developed. It uses Python 3 and Go 1.19 for API implementation. The overhead controller uses both visual monitoring (experiment recording) as well as a wireless network (software upload, swarm maintenance). The ongoing research and development of the project can be tracked on GitHub [55] (the project is in the process of migrating from an internal repository to GitHub and not all components are available yet).

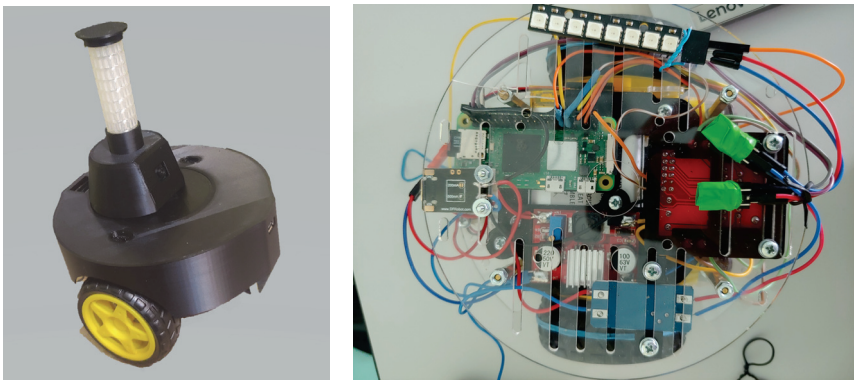


Figure 5. Hardware implementation and view on internal components of the first prototype.

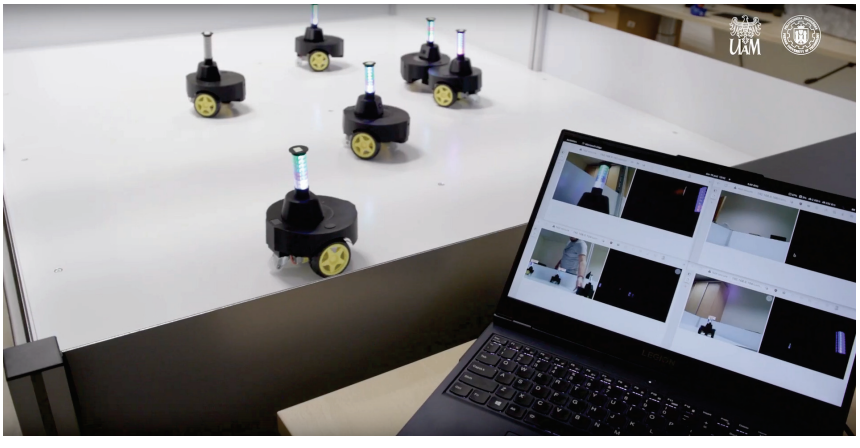


Figure 6. First version prototypes moving on the arena with the live monitoring of cameras.

Second Prototype

The first prototype has some problems that need to be solved so that more effective research can be conducted:

- The robots need to be stopped and physically plugged in for charging when the batteries run out. This causes delays in conducting experiments.
- Current robots are characterized by large dimensions compared to the work area. This minimizes the simultaneous number of robots that can move around the arena.
- The presence of an experimenter is required to activate the robots. This makes it impossible to conduct remote experiments.

For the development of the next, more efficient, and miniaturized iteration of the platform, it is necessary to solve the above problems by creating a custom, compact version of the PCB. This requires conducting a thorough analysis of available circuit boards and selecting the best solutions to use in designing the electrical as well as mechanical structure of the robots that make up the swarm. This will allow the robots to respond to their environment in the best possible way, and allow researchers to conduct experiments on artificial empathy remotely.

Due to the minimization of the robot's dimensions, its design envisions two circuit boards connected above each other. The lower board will be responsible for interacting with the environment, and the upper board will be responsible for information processing and decision-making. The second version of the prototype is less than 8 cm in diameter and has a much smaller height (Figure 7).

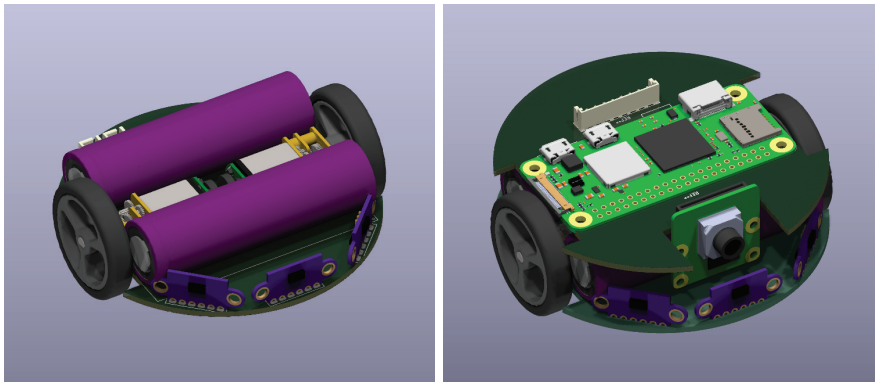


Figure 7. Visualization of the second prototype, with (**left**) and without (**right**) the upper board.

The robot's movement will be accomplished using two miniature-geared motors manufactured by Pololu (Figure 8). Each is equipped with a magnetic encoder based on TLE4946-2K Hall effect sensors, specifically designed for this scenario. The rotation of the motors will additionally be monitored by current sensors, one for each motor, to achieve accurate speed control. To eliminate losses on typical shunt resistor current sensors, a circuit that measures the current based on the Hall effect can be used. Examples of such sensors are the ACS712 or ACS724. To power the motors, DRV8833 will be used as an executive circuit. This controller enables the operation of two motors and is characterized by its requirement of only two control lines per motor to regulate the speed and direction of rotation. Additionally, it is easy to operate using hardware Timers in the microcontroller.

To provide the robot with precise orientation in the field, it will be equipped with an integrated BNO055 chip, consisting of a Bosch accelerometer, gyroscope, and magnetometer. This chip will support the robot's motion algorithms for maximum precision in its maneuvers. In addition, the VL53L0X sensors will provide information about the distance of obstacles in front of the robot. The use of these sensors introduces increased complexity in both the board design and control algorithms compared to traditional push buttons

and limiters. Nevertheless, this change contributes to reducing mechanical contact with the environment, which translates into minimizing the risk of damage to the robot and increasing its reliability.

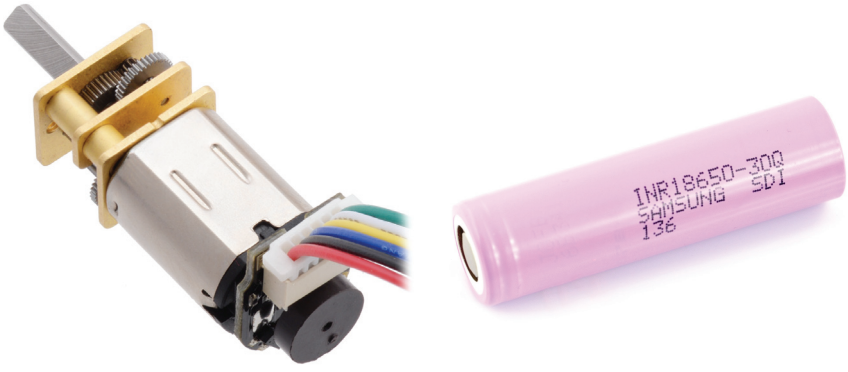


Figure 8. Engine and battery used in the second prototype.

Two Samsung INR18650-35E lithium-ion cells with a total capacity of 7 Ah will be used to power the robot (Figure 8). Due to the robot's small size, the design will utilize cylindrical cells instead of flat lithium-polymer batteries, necessitating the addition of protection circuits (Figure 7). In order to ensure safe power management, a DW01 battery protection circuit will be employed in conjunction with the executive transistors. This circuit is designed to monitor the battery voltage to prevent exceeding the permissible range and to disconnect the voltage in case of a short circuit. The charging process will be supervised by the TP5100 chip, which not only provides up to two amps of current but also ensures that the cell charging process adheres to the constant current constant voltage (CC CV) charging specification. One of the project's goals is to enable the robot to charge without human intervention. To achieve this, two contact pads will be placed on the bottom of the lower circuit board. Through these, the robot will be able to charge its batteries by hovering over a special charging station. This solution is extremely simple, occupies minimal space, and does not introduce energy transfer losses.

The robot, despite its small size, will be equipped with systems that require high power consumption. To meet their expectations, the project will use two inverters, one with an output voltage of 5 V—U3V70A, which is capable of supplying up to 10 A at peak demand even for a few seconds, and the second inverter, this time for circuits powered by 3.3 V—U7V8F3, will provide the currents for all sensors and the microcontroller. It is worth noting that the 3.3 V inverter will operate in two modes: step-down when the battery is charged and step-up when the battery is closer to being discharged.

The bottom board will also house a 32-bit microcontroller from the STM32 family. Its high computing power, flexibility, and popularity will allow the implementation of almost arbitrarily complex algorithms. It will control all the above circuits and communicate with the robot's main processor.

The role of the device's brain will be played by the Raspberry Pi Zero 2W; it is a quad-core, single-board computer on a top circuit board, which will react to its environment and other robots with the help of the Raspberry Pi Cam V3. All artificial intelligence and empathy algorithms will be implemented right on this microcomputer. Signaling of its own internal state will be achieved by individually addressable RGB LEDs—WS2812B, arranged in 8 rows, each with 3 LEDs. This will make it possible to display at least one and a half million different states. A detailed diagram showing the internal layout of the proposed device is shown in Figure 9.

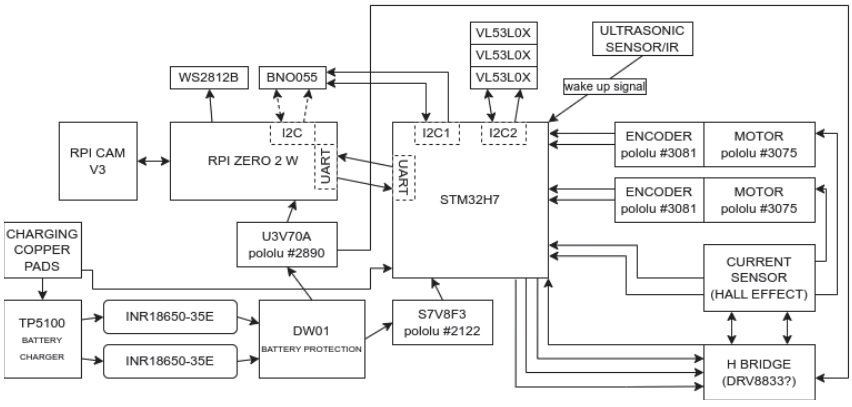


Figure 9. Block diagram showing the internal layout of the second prototype.

Experimentation Arena

One of the challenges was to create a suitable arena that serves as an environment for controlled experiments (Figure 10). Another still not fully solved challenge is the inability to modify and introduce variability to experimental platforms, limiting the number of test scenarios and hindering research in different applications. This arena was designed to simulate various environmental conditions and allow the study of swarm behavior in different situations, added to the arena as modular modifications (e.g., obstacles).

This will allow testing whether and how (with what energy expenditure and difficulties) the robot can perform simple tasks in the presence of designed variable elements in the environment. Conducting these experiments will provide insights into the requirements for the experimental arena and robot prototypes, allowing for further considerations in subsequent prototyping iterations.

Figure 10. Visualization and realization of the controlled environment for the empathetic robot swarm arena.

4. Conclusions

This paper proposes a model for swarm behavior control through artificial empathy, using the fuzzy set theory and similarity measures. The model emulates human empathetic

decision processes to optimize behavior toward achieving a common goal. The authors' main contribution is redefining the theoretical cognitive empathy model into a particular swarm control model, with a significant focus on knowledge representation and decision-making processes.

Our model differs from existing swarm models of artificial empathy, such as [36,41], in that it incorporates empathetic communication, knowledge representation, and the use of similarity measures to convey empathy mechanisms. Unlike those models, which utilize a simple parameter to describe the empathy target's state, our approach is more comprehensive and effective. This approach is inspired by the emotional/cognitive empathy model in neural science and filters agent-broadcasted states based on similarity to known situations.

The example presented here demonstrates the model's versatility across platforms, emphasizing only a fraction of its applicability. The model's independence from a specific application allows for the separate development of empathetic decision-making modules and platform controllers. It is worth noting that the model could be applied in dynamically changing environments, where agents can 'borrow' knowledge from others. Incorporating the perceived rewards of other agents into the decision-making process could enable neighboring agents to learn from experience by comparing differences in perceived and realized rewards. This paper lays the groundwork for further exploration and development of empathetic swarm control models in diverse environments.

The integrated experimental environment proposed in the article is an important step in the development of research on swarm robot control algorithms. This, in turn, will increase the possibility of implementing the results of scientific research in practice. The proposed experimental platform, its scope, and architecture are based on the experience that has been developed while working on two of its earlier prototypes. As a result, it was possible to optimize many technical and operational parameters of the developed solution. It is also not insignificant that the work on the hardware environment is strongly connected with research on empathetic robotic swarms, imposing real requirements on the direction of the project's development. The most significant example of this is the use of innovative vision communication using LED towers.

In further research, we want to upgrade the model to more accurately recreate human empathy. The egoistic behavior evaluation module will be implemented using a neural network, and the similarity of other agents' states will be determined based on internal knowledge represented by the net. Also, reinforcement learning is to be used to teach the net new data and adapt to the environments and behaviors of other agents.

Our future work will aim to validate the artificial empathy model as an effective learning method by simulating empathetic behavior in isolation from the physical layer of the robots in the swarm. Further research should express the proposed learning model in the language of reinforcement learning and use the tools available in this area to obtain reproducible quantitative results that demonstrate the effectiveness of empathetic methods.

The variability of the experimental environment is a distinctive feature of the proposed experimental platform. Research efforts will be necessary to identify available and feasible strategies to achieve this goal. Further research will consider elements of a variable environment, such as changing parameters (e.g., lighting) and altering the structure of the arena (obstacles, cooperative logical puzzles). A key assumption is the automation of the process for introducing modifications to the arena. This is crucial for achieving repeatability in experiment results and enabling remote access to the platform.

Author Contributions: Conceptualization, J.S., P.Ż. and K.D.; methodology, J.S. and P.Ż.; software, P.Ż., P.S., A.W., W.W. and K.P.; formal analysis, J.S. and P.Ż.; resources, P.S., A.W., W.W. and K.P.; writing—original draft preparation, J.S., P.Ż. and K.D.; writing—review and editing, P.Ż.; visualization, J.S. and P.Ż.; supervision, J.S.; project administration, J.S.; funding acquisition, J.S. and K.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

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

References

1. Rohmer, E.; Singh, S.P.N.; Freese, M. CoppeliaSim (formerly V-REP): A Versatile and Scalable Robot Simulation Framework. In Proceedings of the International Conference on Intelligent Robots and Systems (IROS),Tokyo, Japan, 3–7 November 2013.
2. Balaji, B.; Mallya, S.; Genc, S.; Gupta, S.; Dirac, L.; Khare, V.; Roy, G.; Sun, T.; Tao, Y.; Townsend, B.; et al. DeepRacer: Educational Autonomous Racing Platform for Experimentation with Sim2Real Reinforcement Learning. *arXiv* **2019**, arXiv:cs.LG/1911.01562.
3. Rubenstein, M.; Ahler, C.; Nagpal, R. Kilobot: A low cost scalable robot system for collective behaviors. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 14–18 May 2012; pp. 3293–3298. [CrossRef]
4. Drigas, A.S.; Papoutsi, C. A new layered model on emotional intelligence. *Behav. Sci.* **2018**, *8*, 45. [CrossRef] [PubMed]
5. Barbey, A.K.; Colom, R.; Grafman, J. Distributed neural system for emotional intelligence revealed by lesion mapping. *Soc. Cogn. Affect. Neurosci.* **2014**, *9*, 265–272. [CrossRef]
6. Decety, J.; Lamm, C. Human empathy through the lens of social neuroscience. *Sci. World J.* **2006**, *6*, 1146–1163. [CrossRef] [PubMed]
7. Xiao, L.; Kim, H.j.; Ding, M. An introduction to audio and visual research and applications in marketing. *Rev. Mark. Res.* **2013**, *10*, 213–253.
8. Yalçın, Ö.N.; DiPaola, S. Modeling empathy: building a link between affective and cognitive processes. *Artif. Intell. Rev.* **2020**, *53*, 2983–3006. [CrossRef]
9. Mariani, M.M.; Machado, I.; Magrelli, V.; Dwivedi, Y.K. Artificial intelligence in innovation research: A systematic review, conceptual framework, and future research directions. *Technovation* **2023**, *122*, 102623. [CrossRef]
10. Liu-Thompkins, Y.; Okazaki, S.; Li, H. Artificial empathy in marketing interactions: Bridging the human-AI gap in affective and social customer experience. *J. Acad. Mark. Sci.* **2022**, *50*, 1198–1218. [CrossRef]
11. Christov-Moore, L.; Reggente, N.; Vaccaro, A.; Schoeller, F.; Pluimer, B.; Douglas, P.K.; Iacoboni, M.; Man, K.; Damasio, A.; Kaplan, J.T. Preventing antisocial robots: A pathway to artificial empathy. *Sci. Robot.* **2023**, *8*, eabq3658. [CrossRef]
12. Chen, J.; Liu, B.; Qu, Z.; Wang, C. Empathy structure in multi-agent system with the mechanism of self-other separation: Design and analysis from a random walk view. *Cogn. Syst. Res.* **2023**, *79*, 175–189. [CrossRef]
13. Morris, R.R.; Kouddous, K.; Kshirsagar, R.; Schueller, S.M. Towards an artificially empathic conversational agent for mental health applications: system design and user perceptions. *J. Med. Internet Res.* **2018**, *20*, e10148. [CrossRef] [PubMed]
14. Leite, I.; Pereira, A.; Castellano, G.; Mascarenhas, S.; Martinho, C.; Paiva, A. Modelling empathy in social robotic companions. In Proceedings of the Advances in User Modeling: UMAP 2011 Workshops, Girona, Spain, 11–15 July 2011; Revised Selected Papers 19; Springer: Berlin/Heidelberg, Germany, 2012; pp. 135–147.
15. Vargas Martin, M.; Pérez Valle, E.; Horsburgh, S. Artificial empathy for clinical companion robots with privacy-by-design. In Proceedings of the Wireless Mobile Communication and Healthcare: 9th EAI International Conference, MobiHealth 2020, Virtual Event, 19 November 2020; Proceedings; Springer: Berlin/Heidelberg, Germany, 2021; pp. 351–361.
16. Blanchard, L. Creating Empathy in Video Games. Master's Thesis, The University of Dublin, Dublin, Ireland, 2016.
17. Fougères, A.J. A modelling approach based on fuzzy agents. *arXiv* **2013**, arXiv:1302.6442.
18. Yulita, I.N.; Fanany, M.I.; Arymuthy, A.M. Bi-directional long short-term memory using quantized data of deep belief networks for sleep stage classification. *Procedia Comput. Sci.* **2017**, *116*, 530–538. [CrossRef]
19. Mohmed, G.; Lotfi, A.; Pourabdollah, A. Enhanced fuzzy finite state machine for human activity modelling and recognition. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *11*, 6077–6091. [CrossRef]
20. Dubois, D.; Prade, H. The three semantics of fuzzy sets. *Fuzzy Sets Syst.* **1997**, *90*, 141–150. [CrossRef]
21. Żywica, P.; Baczyński, M. An effective similarity measurement under epistemic uncertainty. *Fuzzy Sets Syst.* **2022**, *431*, 160–177. [CrossRef]
22. Asada, M. Development of artificial empathy. *Neurosci. Res.* **2015**, *90*, 41–50. [CrossRef]
23. Suga, Y.; Ikuma, Y.; Nagao, D.; Sugano, S.; Ogata, T. Interactive evolution of human-robot communication in real world. In Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, Edmonton, AB, Canada, 2–6 August 2005; IEEE: Piscataway, NJ, USA, 2005; pp. 1438–1443.
24. Asada, M. Towards artificial empathy. *Int. J. Soc. Robot.* **2015**, *7*, 19–33. [CrossRef]
25. De Waal, F.B. The 'Russian doll'model of empathy and imitation. In *On Being Moved: From Mirror Neurons to Empathy*; John Benjamins Publishing Company: Amsterdam, The Netherlands, 2007; pp. 35–48.
26. Yalçın, Ö.N. Empathy framework for embodied conversational agents. *Cogn. Syst. Res.* **2020**, *59*, 123–132. [CrossRef]
27. Fiske, A.; Henningsen, P.; Buyx, A. Your robot therapist will see you now: ethical implications of embodied artificial intelligence in psychiatry, psychology, and psychotherapy. *J. Med. Internet Res.* **2019**, *21*, e13216. [CrossRef]
28. Possati, L.M. Psychoanalyzing artificial intelligence: the case of Replika. *AI Soc.* **2023**, *38*, 1725–1738. [CrossRef]
29. Affectiva Inc. *Media Analytics*; Affectiva Inc.: Boston, MA, USA. Available online: https://go.affectiva.com/affdex-for-market-research (accessed on 20 February 2023).

30. Leite, I.; Mascarenhas, S.; Pereira, A.; Martinho, C.; Prada, R.; Paiva, A. Why can't we be friends? An empathic game companion for long-term interaction. In Proceedings of the International Conference on Intelligent Virtual Agents, Philadelphia, PA, USA, 20–22 September 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 315–321.
31. Sierra Rativa, A.; Postma, M.; Van Zaanen, M. The influence of game character appearance on empathy and immersion: Virtual non-robotic versus robotic animals. *Simul. Gaming* **2020**, *51*, 685–711. [CrossRef]
32. Aylett, R.; Barendregt, W.; Castellano, G.; Kappas, A.; Menezes, N.; Paiva, A. An embodied empathic tutor. In Proceedings of the 2014 AAAI Fall Symposium Series, Arlington, TX, USA, 13–15 November 2014.
33. Obaid, M.; Aylett, R.; et al. Endowing a robotic tutor with empathic qualities: design and pilot evaluation. *Int. J. Humanoid Robot.* **2018**, *15*, 1850025. [CrossRef]
34. Affectiva Inc. *Interior Sensing*; Affectiva Inc.: Boston, MA, USA. Available online: http://go.affectiva.com/auto (accessed on 20 February 2023).
35. Ebert, J.T.; Gauci, M.; Nagpal, R. Multi-feature collective decision making in robot swarms. In Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, Stockholm, Sweden, 10–15 July 2018; pp. 1711–1719.
36. Huang, F.W.; Takahara, M.; Tanev, I.; Shimohara, K. Effects of Empathy, Swarming, and the Dilemma between Reactiveness and Proactiveness Incorporated in Caribou Agents on Evolution of their Escaping Behavior in the Wolf-Caribou Problem. *SICE J. Control. Meas. Syst. Integr.* **2018**, *11*, 230–238. [CrossRef]
37. Witkowski, O.; Ikegami, T. Swarm Ethics: Evolution of Cooperation in a Multi-Agent Foraging Model. In Proceedings of the First International Symposium on Swarm Behavior and Bio-Inspired Robotics, Kyoto, Japan, 28–30 October 2015.
38. Chen, J.; Zhang, D.; Qu, Z.; Wang, C. Artificial Empathy: A New Perspective for Analyzing and Designing Multi-Agent Systems. *IEEE Access* **2020**, *8*, 183649–183664. [CrossRef]
39. Li, H.; Oguntola, I.; Hughes, D.; Lewis, M.; Sycara, K. Theory of Mind Modeling in Search and Rescue Teams. In Proceedings of the IEEE International Conference on Robot and Human Interactive Communication, Napoli, Italy, 29 August–2 September 2022; pp. 483–489. [CrossRef]
40. Li, H.; Zheng, K.; Lewis, M.; Hughes, D.; Sycara, K. Human theory of mind inference in search and rescue tasks. *Proc. Hum. Factors Ergon. Soc. Annu. Meet.* **2021**, *65*, 648–652. [CrossRef]
41. Huang, F.; Takahara, M.; Tanev, I.; Shimohara, K. Emergence of collective escaping strategies of various sized teams of empathic caribou agents in the wolf-caribou predator-prey problem. *IEEJ Trans. Electron. Inf. Syst.* **2018**, *138*, 619–626. [CrossRef]
42. Mondada, F.; Bonani, M.; Raemy, X.; Pugh, J.; Cianci, C.; Klaptocz, A.; Magnenat, S.; Zufferey, J.C.; Floreano, D.; Martinoli, A. The e-puck, a robot designed for education in engineering. In Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions, Castelo Branco, Portugal, 7 May 2009; IPCB: Instituto Politécnico de Castelo Branco: Castelo Branco, Portugal, 2009; Volume 1, pp. 59–65.
43. Arvin, F.; Espinosa, J.; Bird, B.; West, A.; Watson, S.; Lennox, B. Mona: an affordable open-source mobile robot for education and research. *J. Intell. Robot. Syst.* **2019**, *94*, 761–775. [CrossRef]
44. Arvin, F.; Murray, J.; Zhang, C.; Yue, S. Colias: An autonomous micro robot for swarm robotic applications. *Int. J. Adv. Robot. Syst.* **2014**, *11*, 113. [CrossRef]
45. Villemure, É.; Arsenault, P.; Lessard, G.; Constantin, T.; Dubé, H.; Gaulin, L.D.; Groleau, X.; Laperrière, S.; Quesnel, C.; Ferland, F. SwarmUS: An open hardware and software on-board platform for swarm robotics development. *arXiv* **2022**, arXiv:2203.02643.
46. Bräunl, T. *The EyeSim Mobile Robot Simulator*; Technical report, CITR; The University of Auckland: Auckland, New Zealand, 2000.
47. Valentini, G.; Antoun, A.; Trabattoni, M.; Wiandt, B.; Tamura, Y.; Hocquard, E.; Trianni, V.; Dorigo, M. Kilogrid: a novel experimental environment for the Kilobot robot. *Swarm Intell.* **2018**, *12*, 245–266. [CrossRef]
48. Xuecheng, L. Entropy, distance measure and similarity measure of fuzzy sets and their relations. *Fuzzy Sets Syst.* **1992**, *52*, 305–318. [CrossRef]
49. Zadeh, L.A. Similarity relations and fuzzy orderings. *Inf. Sci.* **1971**, *3*, 177–200. [CrossRef]
50. Cross, V.V.; Sudkamp, T.A. *Similarity and Compatibility in Fuzzy Set Theory. Assessment and Applications*; Physica: Heidelberg, Germany, 2002.
51. Couso, I.; Garrido, L.; Sánchez, L. Similarity and dissimilarity measures between fuzzy sets: A formal relational study. *Inf. Sci.* **2013**, *229*, 122–141. [CrossRef]
52. MacQueen, J.; et al. Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Oakland, CA, USA, 21 June–18 July 1967; Volume 1, pp. 281–297.
53. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525. [CrossRef]
54. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**. [CrossRef]
55. Żywica, P.; Wójcik, A.; Siwek, P. Open-source Physical-Based Experimentation Platform Source Code Repository. Available online: https://github.com/open-pep (accessed on 20 February 2023).

sensors

MDPI

Article

An Integrated Autonomous Dynamic Navigation Approach toward a Composite Air–Ground Risk Construction Scenario

Da Jiang [1], Meijing Wang [1], Xiaole Chen [1,*], Hongchao Zhang [1], Kang Wang [2], Chengchi Li [2], Shuhui Li [1] and Ling Du [1]

[1] National Engineering Laboratory for Wheeled Vehicle, China North Vehicle Research Institute, Beijing 100072, China; ziangdar@sina.com (D.J.); 15201615492@163.com (M.W.); d202187005@hust.edu.cn (H.Z.); dreamy_on2023@163.com (S.L.); duling0615@163.com (L.D.)

[2] School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China; m202170779@hust.edu.cn (K.W.); d202280230@hust.edu.cn (C.L.)

* Correspondence: chenxl4227@163.com

Abstract: Unmanned transportation in construction scenarios presents a significant challenge due to the presence of complex dynamic on-ground obstacles and potential airborne falling objects. Consequently, the typical methodology for composite air–ground risk avoidance in construction scenarios holds enormous importance. In this paper, an integrated potential-field-based risk assessment approach is proposed to evaluate the threat severity of the environmental obstacles. Meanwhile, the self-adaptive dynamic window approach is suggested to manage the real-time motion planning solution for air–ground risks. By designing the multi-objective velocity sample window, we constrain the vehicle's speed planning instructions within reasonable limits. Combined with a hierarchical decision-making mechanism, this approach achieves effective obstacle avoidance with multiple drive modes. Simulation results demonstrate that, in comparison with the traditional dynamic window approach, the proposed method offers enhanced stability and efficiency in risk avoidance, underlining its notable safety and effectiveness.

Keywords: motion planning; hierarchical decision; self-adaptive dynamic window approach; risk assessment

Citation: Jiang, D.; Wang, M.; Chen, X.; Zhang, H.; Wang, K.; Li, C.; Li, S.; Du, L. An Integrated Autonomous Dynamic Navigation Approach toward a Composite Air–Ground Risk Construction Scenario. *Sensors* **2024**, *24*, 221. https://doi.org/10.3390/s24010221

Academic Editor: Wenling Li

Received: 7 November 2023
Revised: 18 December 2023
Accepted: 27 December 2023
Published: 30 December 2023

1. Introduction

Real-time motion planning holds significant importance within the realm of autonomous driving applications. Extensive research has been conducted on motion planning in structured environments for unmanned ground vehicles (UGVs) [1–6]. However, quantity research focuses on general ground risk-avoidance motion planning, overemphasizing the target identification or kinematic-level reactive maneuvers, while downplaying the vehicle's dynamics-level planning constraints. Moreover, it overlooks the need for reactive evasive maneuvers in response to abrupt intrusions by airborne threats. The issue is particularly pronounced in the navigation problem of unmanned transport vehicles (UTVs) in construction scenarios. On the one hand, the control system needs to balance vehicle dynamics constraints and the rapid dynamic obstacle avoidance requirements online. On the other hand, the UTV needs to swiftly perform evasive maneuvers in extreme response times to mitigate high-speed falling objects while ensuring compliance with vehicle dynamic capabilities.

This article aims to design an auxiliary motion function module for motion planning in construction scenarios with coupled mobility risks. Once an aerial threat is detected, the module is promptly activated to real-time plan the reactive maneuver trajectory for the vehicle within an extremely short response time, aiming to maximize vehicle safety toward the rapid descent hazards.

Despite efforts in some research to quantify airborne threats as either general ground risks or ground risks with altitude information [7,8], practical applications of UTVs reveal that spatial threats cannot be adequately characterized as typical ground threats. This distinction is especially pronounced in construction scenarios. Firstly, within the maneuvering operation time scale, not all detected obstacles need to be classified as potential risks. For instance, in construction scenarios, a multitude of ground obstacles such as people, vehicles, buildings, and construction materials are present, whereas only targets closely positioned in the expected trajectory's spatial proximity will significantly impact or pose a threat to the planning trajectory. Furthermore, airborne objects like falling leaves and small balloons, although detected as airborne targets in opposing motion, do not pose risks to the travel of UTVs. Therefore, through isolating or discounting redundant low-risk factors, risk quantification assessment for the various detected targets needs to be employed to reduce planning costs within the short-term planning window.

The anticipated on-ground distance metric generally has a positive effect during the ground risk assessment. However, this metric should not be directly migrated to the risk assessment process for airborne objects, as the vehicle's altitude would affect the risk assessment [9–12]. For instance, once an airborne object approaches UTVs on a horizon or slightly inclined trajectory, its anticipated landing spot would be at a significant distance from the vehicle. Thus, the original metric would deem this trajectory as safe, while the collision would indeed occur. In addition, the presence of airborne objects results in an extremely short maneuvering time scale for UTVs (on the order of seconds). This greatly emphasizes the importance of risk quantification assessment in filtering objects considered redundant. Moreover, the short time scale for risk avoidance operations could frequently lead to vehicles employing aggressive control maneuvers, imposing significant dynamic burden or danger, such as the heightened risk of rollovers during a high-speed turning maneuver. So, it is essential to integrate the dynamics constraints into the planning. Above all these considerations, it is necessary to conduct reasonable risk quantification assessments for both ground and airborne obstacles, to streamline redundant obstacles by maintaining the planning effectiveness of the UTV, and eventually to reduce the computation costs of large-scale multi-object motion planning in stereospace.

Our contribution to this study is offering a hierarchical self-adaptive motion planning method for UTVs in construction scenarios, which successfully addresses the integrated air–ground risk avoidance problem. This work provides an efficient planning solution for the coupled risk field in the construction scene. The organization of this paper is as follows: Section 2 describes our proposed planning scheme and UTVs' dynamics constraint model for the construction scenario. Section 3 explains the theoretical methodology of the hierarchical decision and self-adaptive planning method in detail. In Section 4, simulations are conducted and the result is presented and discussed. Finally, the conclusion and future work are addressed in Section 5.

2. Related Works

Local planning for unmanned vehicles is a crucial component of autonomous driving systems, aiming to enable vehicles to navigate safely and efficiently in complex and dynamic environments. In recent years, researchers have focused on addressing the challenge of coupled air–ground obstacle avoidance to enhance the reliability and adaptability of unmanned vehicles in complex scenarios.

There is existing prior research that independently analyzes the dynamic obstacle avoidance for airborne risk. Chen et al. [13] proposed an A*-cubic-spline-based dynamic planning method for an unmanned vehicle under sudden threats. Once a sudden threat is encountered, the corresponding parameters are set according to the cubic spline second-order continuity, and multiple candidate trajectories are generated. Then, the optimal trajectory is obtained according to the set objective function. This method does not meet the time margin for abrupt intrusions by airborne threats. Feng et al. [14] proposed a new dynamic path planning algorithm based on the modified artificial potential field

algorithm. Once an obstacle enters the observation range of the vehicle, the movement information of the obstacle will be observed and recorded in the obstacle grid map. The prediction of obstacle position is obtained with the trajectory evaluation algorithm based on the Markov chain. Zhou et al. [15] proposed a bio-inspired path planning algorithm, which utilized the A* search algorithm to explore the generated probability map, and designed an artificial-field-based objective function, to obtain an original optimal collision-free path. The computational cost of this coupling algorithm is also relatively high. Recent decades have witnessed that deep reinforcement learning provides new perspectives for optimizing control problems in unmanned vehicles [16,17]. Singla et al. [18] proposed a deep reinforcement learning planning method based on a recurrent neural network and temporal attention. This method enables the UAV controller to collect, and store relevant observations gathered over time and use them to make better obstacle avoidance decisions. Kulathunga et al. [19] proposed a hybrid approach that combines a Montecarlo tree search and an RL-based approach to solve the 3D path planning problem. The end-to-end data-driven method is primarily tailored to specific vehicle models, exhibiting a high dependence on environmental modeling and limited generalization capability.

However, these approaches have not considered the real-time dynamic limits of the vehicle, and require calculations for all aerial obstacles, incurring high computational costs. Hence, these approaches are not suitable for a construction scenario with high maneuverability demands. In summary, researchers have conducted in-depth studies on path planning and control methods to address the challenge of the coupled air–ground obstacle avoidance problem. These efforts provide important theoretical foundations and practical experiences for achieving safe and efficient local path planning for unmanned vehicles in complex environments.

3. Problem Statements

The conceptual sketch illustrating the motion planning mechanism process for the construction navigation scenario is presented in Figure 1. During the UTV executing the transportation mission at the construction site, the environment perception system continuously detects and identifies surrounding obstacles, including on-ground and airborne objects. Under normal circumstances, the vehicle primarily engages in real-time navigation and obstacle avoidance within ground-level environments. Once the perception system perceives descending airborne obstacles, the planning system would be swiftly handed over to the auxiliary motion planning module specialized in addressing aerial-to-ground obstacle scenarios. This module performs the entire planning and decision-making process, encompassing risk quantitative assessment, hierarchical motion decision, and real-time evasion planning function.

Consequently, we can quantify specialized spatial obstacles into general 3D obstacles with the constraints of planning time scale, vehicle's rapid maneuvering capabilities and security logic, etc. Thus, the UTV emergency planning problem with aerial–ground risks in construction scenarios can be transformed into a dynamic closed-loop optimization problem:

$$\begin{gathered} \min : J \\ s.t. \quad \Phi_{\mathrm{t}} \leq 0 \quad \Phi_{\mathrm{m}} \leq 0 \quad \Phi_{\mathrm{s}} \leq 0 \end{gathered} \tag{1}$$

where Φ_{t}, Φ_{m}, and Φ_{s} denote the constrain items of planning time scale, maneuvering capabilities, and security logic, respectively. J denotes the objective function value of the planning optimization problem. In this paper, the detail objective function is defined as:

$$J(v,\omega) = -\sigma(\mu \cdot heading(v,\omega) + \beta \cdot dist_{\mathrm{g}}(v,\omega) + \eta \cdot dist_{\mathrm{o}}(v,\omega) + \gamma \cdot vel(v,\omega)) \tag{2}$$

where $heading$, $dist_{\mathrm{g}}$, $dist_{\mathrm{o}}$, and vel denote the objective terms related to the target heading, target distance, obstacle distance, and velocity performance, respectively. σ, μ, β, η, and γ denote the corresponding coefficients, respectively.

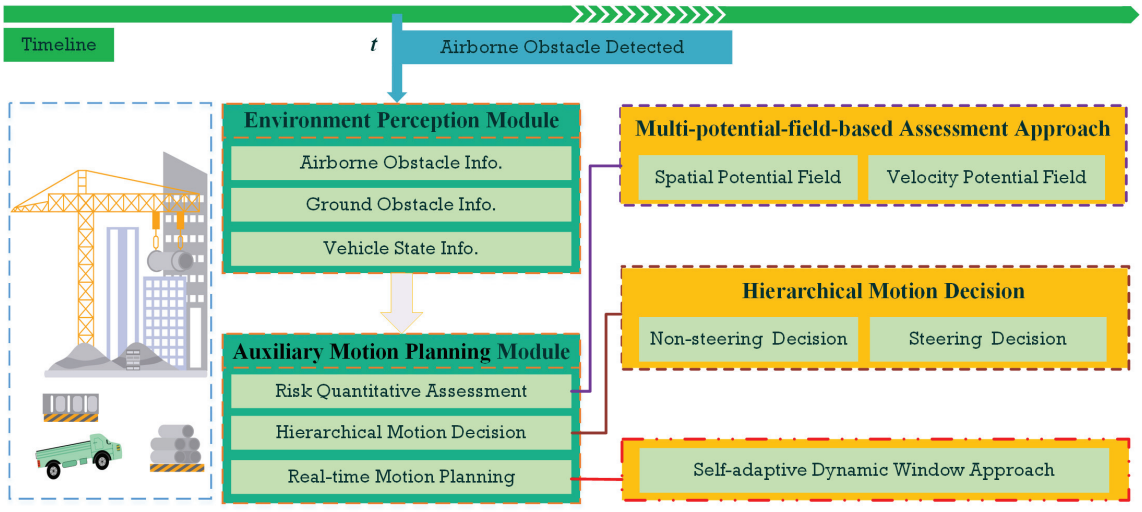


Figure 1. Conceptual overall sketch for the proposed motion planning method.

3.1. Vehicle Prediction Model

During the actual traveling process, it is challenging to obtain real-time acceleration information from the vehicle due to the variation fluctuations and interference noise. Since speed metrics are relatively easier to obtain and have lower interference noise, to facilitate UTV's trajectory prediction analysis, this paper employs the constant turn rate and velocity (CTRV) model to describe the UTV's motion mechanism in the planning problem. In the CTRV prediction model, the vehicle's linear velocity and steering rate in each time step are considered constant, as shown in Figure 2.

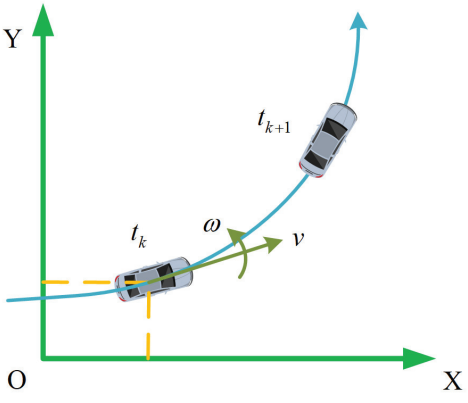


Figure 2. CTRV prediction model sketch.

3.2. Risk Quantitative Assessment

Building upon the adopted CTRV vehicle prediction model, the primary illustration of the risk quantitative assessment sketch for an air–ground obstacle is presented in Figure 3. The assessment system conducts real-time calculations based on incoming air–ground obstacle data. It employs a systematic coarse filtering method to reduce redundant risk points. Following this initial screening, the system applies an additional approach to analyze potential risks associated with airborne obstacles by pinpointing their expected impact points on the vehicle. This risk assessment process is continuously executed at

each sampling time step in parallel. Therefore, the high-risk obstacles are seamlessly incorporated into the real-time motion planning problem.

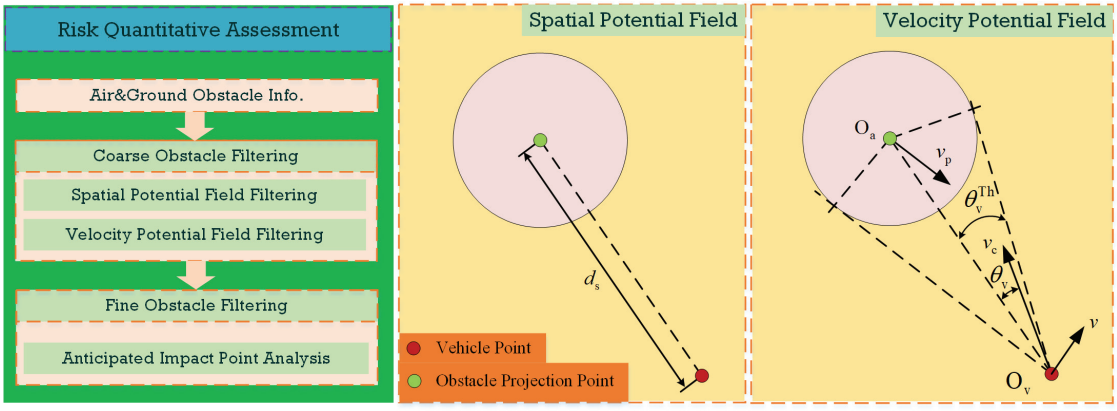


Figure 3. Risk quantitative assessment sketch.

The spatial potential field model is employed to assess both ground and airborne obstacles simultaneously. It is achieved by analyzing whether the anticipated relative distance d_s is projected to reach the danger distance threshold d_s^{Th}. If the threshold is exceeded, the assessment system would hold that the target in question poses a threat to the vehicle at that future moment. For instance, when dealing with airborne obstacles, the system would detect the corresponding trajectory sequence points. Building upon this, the anticipated landing time and landing coordinates of the airborne obstacle could be calculated. Thus, by comparing the calculated distance d_s from the anticipated vehicle to the landing coordinates with the predetermined threshold d_s^{Th}, the system achieves a spatial-potential-field-based risk assessment. The corresponding assessment Boolean indicator R_s can be denoted as:

$$R_s = \begin{cases} \text{True} & d_s < d_s^{Th} \\ \text{False} & d_s > d_s^{Th} \end{cases} \tag{3}$$

Once an airborne obstacle approaches the vehicle in a horizontal or low-angle direction, the obstacle's anticipated landing point would be distant from the vehicle's anticipated position. In such a case, the spatial potential field model is not applicable to the obstacle impacting a vehicle body scene. To address this situation, the velocity potential field model is introduced to supplement the evaluation of the risk posed by horizontal/low-angle-approaching airborne obstacles. This model assesses the anticipated risk by considering the angle between the current obstacle–vehicle composite velocity vector v_c and the obstacle direction vector $\overrightarrow{O_vO_a}$. For instance, assuming the current translational velocity of the vehicle and airborne obstacle as v and v_p, respectively, we can obtain the composite velocity vector $v_c = v - v_p$. When the angle θ_v between v and v_p falls within the corresponding expansion angle θ_v^{Th} in the velocity potential field, the system identifies this airborne obstacle as a danger. The corresponding assessment Boolean indicator R_v can be denoted as:

$$R_v = \begin{cases} \text{True} & \theta_v < \theta_v^{Th} \\ \text{False} & \theta_v > \theta_v^{Th} \end{cases} \tag{4}$$

For the airborne obstacles identified as R_v = True during the assessment, an additional refined risk assessment is performed through the anticipated impact point analysis for the vehicle. The coordinates of the incoming obstacle's trajectory consequence points are

transformed into the coordinates under the vehicle's reference frame, denoted as P_{lm}. If P_{lm} falls outside the threshold range of the vehicle's bounding box, it indicates that the incoming obstacle would not collide with the vehicle body. While if P_{lm} falls within the threshold range of the vehicle's bounding box, it signifies that the incoming obstacle is on a collision course with the vehicle at that moment. Then, the risk assessment system would output the coordinates of the impact point and the corresponding timestamp.

At this stage, based on the spatial/velocity potential field and anticipated impact point analysis, the risk assessment module could real-time filter out obstacle data that pose a genuine threat indeed. These filtered data serve as the basis for subsequent real-time motion planning assessments.

4. Methodology

In intricate construction environments with complex air–ground risks, the restricted maneuvering time scale for UTVs necessitates swift obstacle avoidance, corresponding to a local motion planning issue. Thus, this paper introduces a real-time self-adaptive local motion planning approach based on the dynamic window approach (DWA). This method is employed to resolve dynamic optimization problems posed by the construction scenario motion planning. The local motion planning framework presented in this paper is depicted in Figure 4.

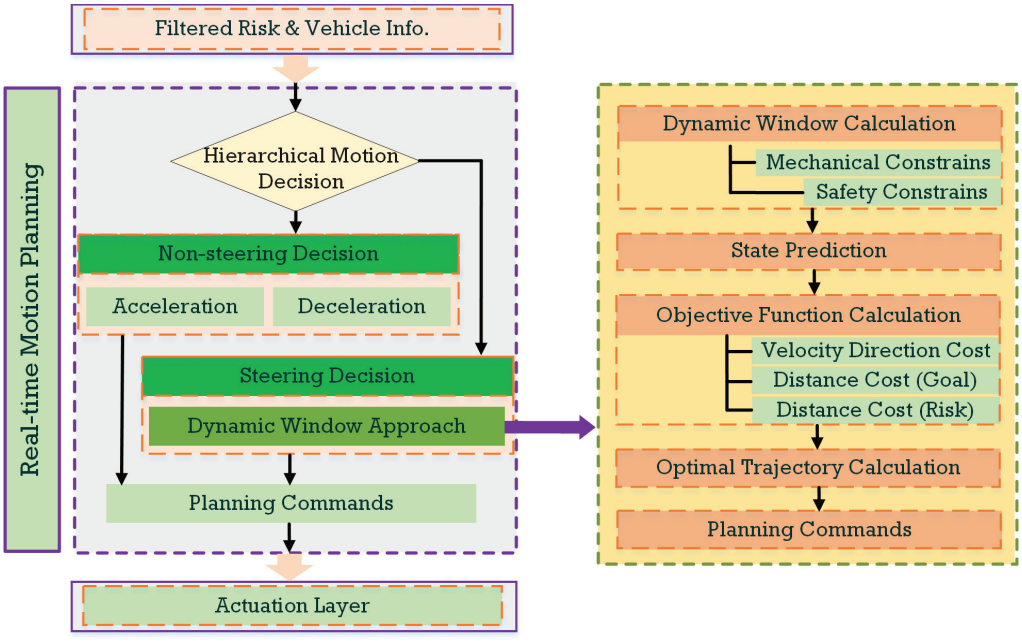


Figure 4. The proposed local motion planning framework.

4.1. Hierarchical Motion Decision

The primary distinction between airborne and ground obstacles centers on the temporal dimension. Airborne threats are inherently time-dependent, existing as potential risks only within a short time scale from the moment of detection until they land on the ground. As a result, the hierarchical motion decision approach, combining the non-steering avoidance decision and steering avoidance decision, is deemed effective for managing airborne obstacles.

Differing from typical hierarchical decision approaches, which are based on temporal or informational flow [20,21], the proposed non-steering/steering-based hierarchical ap-

proach in this paper aims to decrease the destabilizing risks associated with rapid steering actions. Non-steering avoidance involves trajectory adjustments through pure acceleration or deceleration without altering the vehicle's original path, primarily focusing on longitudinal obstacle avoidance. Conversely, steering avoidance employs the vehicle's steering mechanism to execute avoidance maneuvers that account for both lateral and longitudinal aspects of its movement.

Thus, the hierarchical motion decision, encompassing both non-steering and steering avoidance is deployed. Notably, rapid and sharp steering maneuvers, especially at high speed, pose a certain risk to vehicle stability, potentially resulting in skidding or rollovers and, consequently, intensifying the risk. The steering avoidance method, which considers rollover prevention mechanisms, necessitates additional iterations within the dynamic window search, leading to lengthier computational processes and placing greater demands on the control system. Conversely, non-steering avoidance through acceleration and deceleration provides smoother vehicle control, mitigates the risk of instability, and allows the control system a higher response time. Consequently, in emergency scenarios, non-steering approaches are typically given precedence. Within each time window, the vehicle first assesses the feasibility of applying non-steering avoidance strategies. If a viable non-steering solution cannot be generated, steering avoidance strategies are then considered.

4.2. Dynamic-Window-Approach-Based Real-Time Planning Method in Construction Scenario

With the insights of the anticipated impact point, the 3D air–ground obstacle avoidance problem can be converted into a 2D problem. The proposed self-adaptive dynamic window approach (ADWA) for steering-based local motion planning in construction scenarios is illustrated in Algorithm 1. In contrast to the traditional DWA [22], the ADWA introduces an adaptive prediction horizon mechanism that tunes the length of the prediction horizon based on obstacle information. This mechanism effectively mitigates the issues encountered in DWA, where a fixed prediction horizon can result in insolvable scenarios in dense obstacle-laden environments. In contrast, the adaptive dynamic window approach adjusts the prediction horizon length by considering the number of nearby obstacles, providing a more flexible response to complex environments.

Theoretically, the ADWA excels in its adaptability to the dynamic nature of the environment. As obstacle density increases, the adaptive dynamic window automatically reduces the prediction horizon length to respond more sensitively to potential collision risks. Conversely, in relatively open environments, the prediction horizon length can be moderately increased to enhance planning efficiency. This flexibility makes the adaptive dynamic window approach more suitable for a variety of real-world scenarios, balancing path planning safety and efficiency in narrow passages as well as open areas. In summary, the adaptive dynamic window approach overcomes the limitations of traditional dynamic window methods in dealing with complex environments by dynamically adjusting the prediction horizon length during the planning process, thereby improving the adaptability and robustness of the planned trajectory.

Furthermore, during the calculation of admissible velocities within the dynamic window, ADWA takes into account critical physical constraints, including the maximum mechanical steering angle and critical rollover velocity. This ensures a more practical and realistic planning approach in rapid maneuvering scenarios.

During the optimization process outlined in Algorithm 1, the vehicle's dynamic window is influenced by various factors, including speed limits, acceleration constraints, and obstacle avoidance requirements. The corresponding sampled spaces are as follows:

$$\begin{aligned} V_{\mathrm{s}} &= \{(v,\omega)||v| \leq v_{\max} \cap |\omega| \leq \omega_{\max}\} \\ V_{\mathrm{d}} &= \{(v,\omega)|v \in [v_0 - a_{\max}\Delta t, v_0 + a_{\max}\Delta t] \cap \omega \in [\omega_0 - \alpha_{\max}\Delta t, \omega_0 + \alpha_{\max}\Delta t]\} \\ V_{\mathrm{a}} &= \left\{(v,\omega)\middle||v| \leq \sqrt{2dist(v,\omega)a_{\max}}\right\} \end{aligned} \tag{5}$$

where v_{max}, ω_{max}, a_{max}, and α_{max} denotes the maximum for the vehicle's translational velocity, rotational velocity (angle velocity), translational acceleration, and rotational acceleration, respectively.

Algorithm 1. Self-adaptive dynamic window approach

1. Exploration domain: The exploration domain of the dynamic window can be restricted by:
(a) Circular trajectories: The possible trajectories of the ADWA can be described by the velocity pairs (v, ω).
(b) Restricted velocities: Vehicles need to leave sufficient braking distance for each obstacle; hence, the velocity pairs (v, ω) need to be restricted.
(c) Dynamic window: The dynamic performance of the vehicle constrains the variation range of the dynamic window at each sampling moment.
(d) Adaptive prediction domain: Self-adaptive prediction domain enables the vehicle to adjust the trajectories' length according to the obstacles' information to avoid the insolvable solution state.
2. Objective function: The multi-objective function is defined as
$J(v, \omega) = -\sigma(\mu \cdot heading(v, \omega) + \beta \cdot dist_g(v, \omega) + \eta \cdot dist_o(v, \omega) + \gamma \cdot vel(v, \omega))$
With respect to the current position, target position, and orientation of the vehicle this function trades off the following aspects:
(a) Target heading: *heading* is a measure of progress toward the target position. It is maximal if the vehicle moves directly toward the target.
(b) Target distance: $dist_g$ is the distance to the goal location on the end of the prediction trajectory. The smaller the distance to the goal location, the higher the vehicle's desire to move around it.
(c) Clearance: $dist_o$ is the minimum distance from the obstacles along the current planned trajectory. The smaller the distance between the vehicle and obstacles, the higher the safety risk of the vehicle.
(d) Velocity: *vel* is the translational velocity of the vehicle.
The parameter σ smooths the global weighted sum of the four components.

Furthermore, the vehicle's motion is also subject to mechanical operational limits, where the steering angle of the wheels imposes restrictions on the vehicle's lateral movement. The corresponding sampled space is as follows:

$$V_b = \{(v, \omega) || \theta_{wheel}| \leq \theta_{wheel}^{max} \} \tag{6}$$

where θ_{wheel}^{max} denotes the maximum steering angle of the vehicle's front wheels.

Additionally, it is imperative to consider the vehicle's dynamic safety performance to mitigate the risk of vehicle rollovers stemming from excessive steering speeds during high-speed maneuvers. In accordance with the literature [23–25], the critical rollover prevention velocity model for the Ackermann steering model with rear-wheel drive and front axle steering is employed as:

$$\begin{aligned} V_c &= \{(v, \omega) | v \leq v_c \} \\ v_c &= \begin{cases} v_{max} & \theta_{steer} < 0.005 \\ \sqrt{\frac{g h_g}{|\tan \theta_{steer}|}} & \theta_{steer} \geq 0.005 \end{cases} \end{aligned} \tag{7}$$

where v_c and h_g denotes the corresponding critical rollover prevention velocity and height of gravity center, respectively.

Hence, the dynamic window V_r can be calculated by the intersection of the above velocity sample spaces:

$$V_r = V_s \cap V_d \cap V_a \cap V_b \cap V_c \tag{8}$$

5. Simulations and Discussions

It is essential to emphasize that the rapid descent speed of airborne risks places constraints on the available planning time. Therefore, this paper confines its investigation

to scenarios in which the vehicle's air–ground risk avoidance problem theoretically has feasible solutions indeed.

In the simulation, the vehicle commences its mission from a stationary position toward the designated target location. The specific vehicle parameters and simulation settings are outlined in Tables 1 and 2, respectively. The simulation pertaining to airborne factors holds significance only when an aerial object is deemed a risk through the risk assessment method, indicating a real possibility of a collision with the vehicle. To facilitate a meaningful comparison, a comparative simulation approach is employed. Hence, in each simulation, three airborne obstacles are introduced from different directions and velocities. If the vehicle maintains its current trajectory without alterations, it will face consecutive impacts with these obstacles. At three specific time points, synchronized launches of three aerial trajectories occur, all with identical velocity and initial altitude, thus allowing for a comprehensive performance evaluation. In the comparative simulations, to demonstrate the validity of the airborne obstacles, we maintain their direction and velocity properties while adjusting corresponding initial positions, to ensure that these airborne obstacles genuinely pose a threat to the UTV in the comparative simulations. In the different simulations, the shared attributes of the three airborne obstacles are configured in Table 3. The real-time planning objective is considered achieved when the vehicle approaches within a 0.5-m radius of the target location.

Table 1. Physical parameters of the UTV.

Parameter	$v_{\max}$ (m)	$\omega_{\max}$ (deg/s)	$a_{\max}$ (m/s^2)	$\alpha_{\max}$ (deg/s^2)	$\theta_{\text{wheel}}^{\max}$ (deg)	Body Size (m)	Mass Center Height (m)	Wheelbase (m)
Value	15.0	60.0	3.2	60.0	30.0	$3.5 \times 2.0 \times 1.6$	1.0	2.0

Table 2. Simulation settings.

Parameter	Initial Location (m)	Target Location (m)	Sample Frequency (Hz)	Perception Distance (m)	Safe Radius (m)	Air Resistance Coefficient	Air Density (kg/m^3)
Value	(0.0, 0.0)	(90, 90)	10	30.0	3.0	0.2	1.2

Table 3. Shared attributes of the airborne risks.

Parameter	Initial Height (m)	Initial Velocity Vector (m/s)	Mass (kg)	Launching Time (s)
risk 1	15.0	$(-10.0, 3.0, -5.0)$	2.0	2.0
risk 2	25.0	$(-3.0, -5.0, -5.0)$	7.0	7.0
risk 3	25.0	$(4.0, -4.0, 0.0)$	10.0	10.0

The movement settings of the given 10 obstacles are shown in Appendix A. Figure 5 presents the traditional DWA-based hierarchical local planning simulation with air–ground risks in the construction scenario. The labels "Air risk 1–3" in the figure correspond to the time window, which spans from the risk assessment to the risk disappearance (impact to the ground). The labels "Acceleration/Deceleration" in the figure correspond to the non-steering-based command of the hierarchical decision, and the rest red lines correspond to the steering-based command. It is illustrated that the traditional DWA-based planning process takes 25.6 s, but results in a continuous exceeding toward the rollover critical velocity v_c. Moreover, influenced by the nearby obstacles and mechanical constraints, the UTV ultimately follows a much redundant trajectory to the goal destination. As a result, this approach is deemed unsuccessful and ineffective.

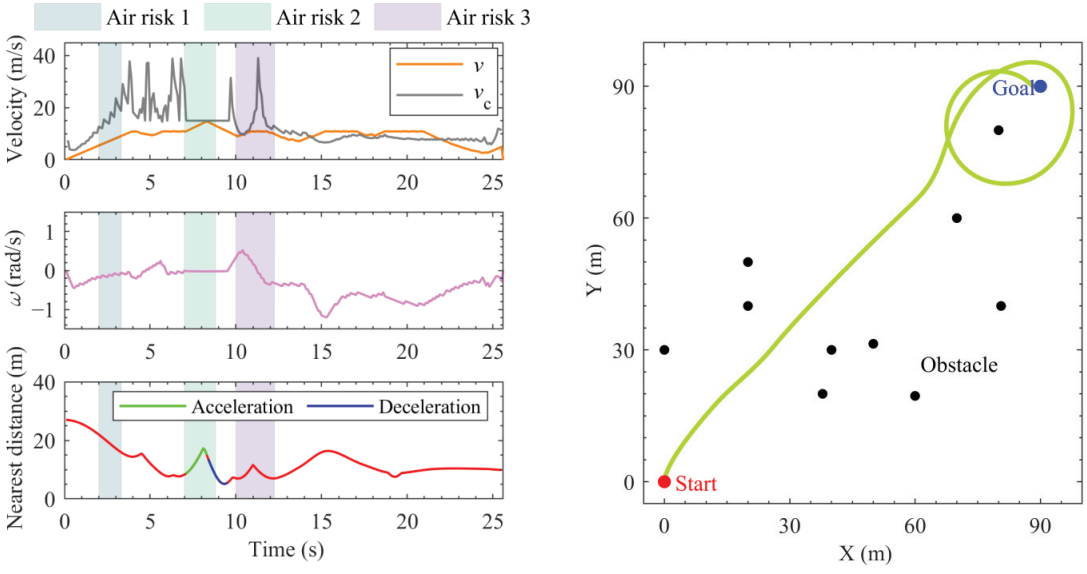


Figure 5. The motion planning results for the traditional DWA.

Figure 6 presents the proposed ADWA-based hierarchical local planning simulation with air–ground risks in the construction scenario. The designed adaptive prediction horizon is set as:

$$t_p = \tau t_{pm} \qquad \tau = clip\left\{\frac{1.0}{n_o + \zeta}, [0.4, 1.0]\right\} \tag{9}$$

where t_{pm} denotes the max prediction time, set to 1.5 s in this paper. n_o denotes the number of obstacles around within 20 m. ζ is a positive value to prevent the calculation error when $n_o = 0$, set to one. *clip* denotes the boundary-constraining function. Thus, when there are excessive obstacles near the vehicle, the system will automatically reduce the prediction horizon to prevent unsolvable scenarios resulting from the over-large prediction scale. And the calculation cost can be correspondingly reduced.

Throughout the planning process, hierarchical planning decision operations are executed, involving obstacle avoidance maneuvers achieved through pure acceleration and deceleration. This planning effect illustrates the necessity and effectiveness of layered control. It is depicted that the proposed ADWA-based planning process takes 20.9 s, in which the vehicle's velocity remains consistently below the rollover critical velocity v_c. The distance from the UTV to the real-time nearest obstacle consistently adheres to the designed safety distance threshold. Additionally, in the final phase of planning around the goal destination, the corresponding planning trajectory still performs a smooth tendency. This overall planning process reflects that the proposed ADWA exhibits clear advantages in terms of safety and feasibility compared to the traditional DWA.

The corresponding aerial view is illustrated in Figure 7. It shows that the landing point of the airborne risk intersects with the trajectory, which seems to be a collision. In fact, combining Figures 6 and 7b, the UTV adopts a series-coupled acceleration/deceleration and steering maneuver to circumvent the risk presented by airborne risk 1. This adjustment effectively redirected the initial collision trajectory of risk 1 to the vehicle's lateral side. Similar avoidance strategies were employed to evade risks 2 and 3. The simulation results demonstrate the effectiveness of the hierarchical decision mechanism and the proposed ADWA method for the air–ground risk in construction scenarios.

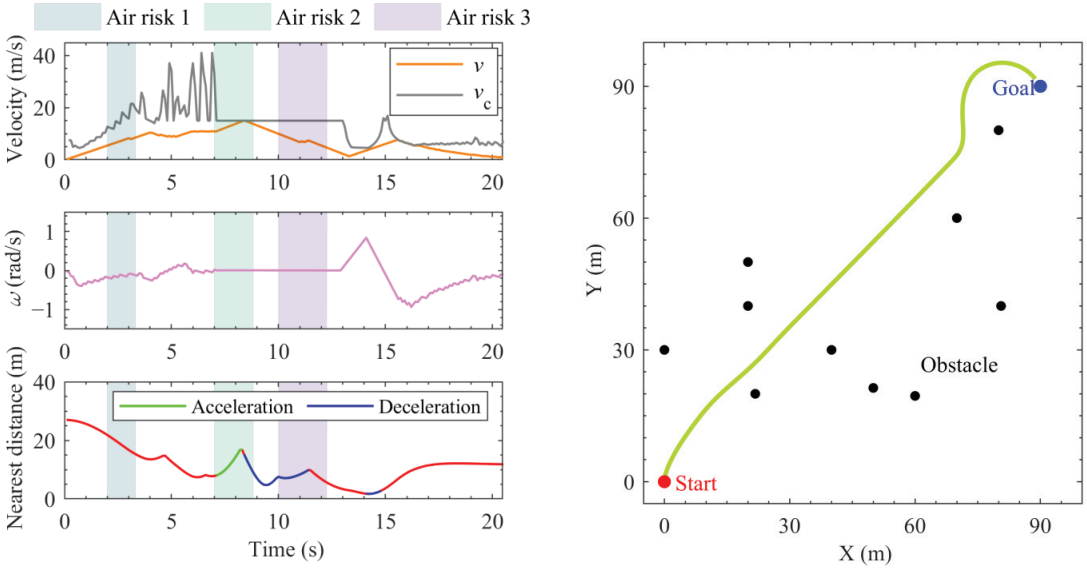


Figure 6. The motion planning results for the proposed ADWA.

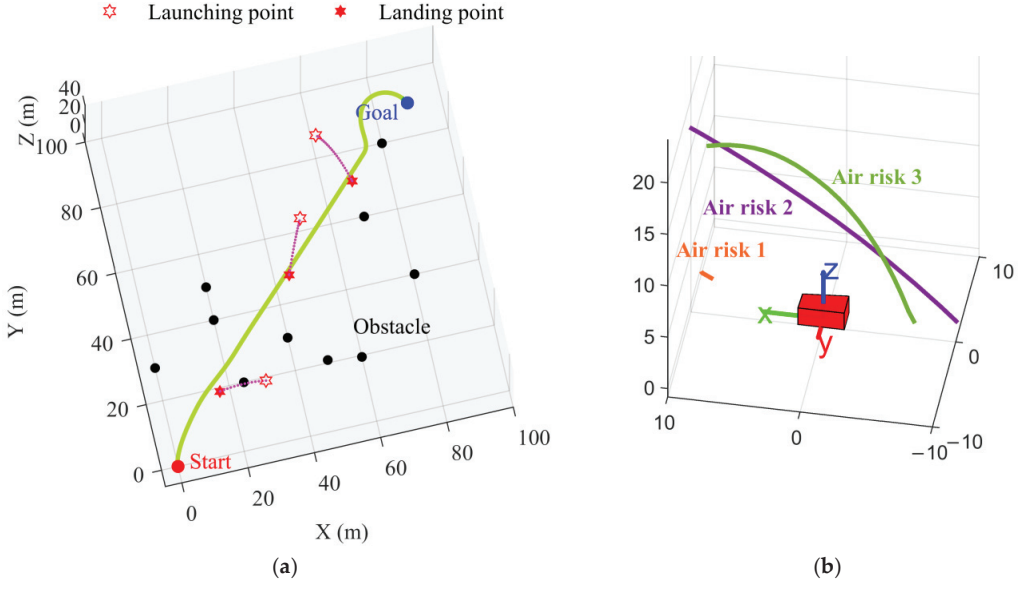


Figure 7. The motion planning results for the proposed ADWA: (**a**) Aerial view; (**b**) Local view.

6. Conclusions

In this paper, we have demonstrated a novel integrated autonomous dynamic navigation approach for construction scenarios with composite air–ground risks. In response to the potential falling obstacle risk of the construction scene, a hierarchical self-adaptive DWA algorithm has been accordingly proposed. Through the hierarchical decision mechanism, the vehicle is directed to minimize air–ground risks through non-steering maneuvers. In the proposed ADWA algorithm, the self-adaptive prediction horizon and the rollover critical velocity window were proposed, enabling the vehicle to fine-tune planning win-

dows in high-risk scenarios while ensuring planning safety. Simulation results show that the proposed hierarchical self-adaptive dynamic window approach demonstrates higher planning efficiency and safety toward the traditional dynamic window approach, with the UTV consistently staying within the safety constraints of the construction scenario. This work provides an efficient solution for vehicle motion planning under air–ground-coupled risks. However, there is still enhancement potential in the algorithm's adaptive capability. Future work will focus on improving algorithm performance and conducting relevant experimental validation.

Author Contributions: Conceptualization, D.J., M.W. and K.W.; methodology, D.J., X.C., H.Z. and K.W.; software, M.W.; validation, D.J., K.W. and X.C.; formal analysis, S.L.; investigation, S.L.; resources, L.D.; data curation, C.L.; writing—original draft preparation, K.W.; writing—review and editing, D.J.; visualization, L.D.; supervision, L.D.; project administration, M.W. and H.Z.; funding acquisition, M.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Central Military Commission Science and Technology Commission Project. The grant number is currently confidential.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

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

Appendix A

Table A1. Initial obstacle information.

Obstacle ID	Initial Location (m)	$v_x^o(0)$ (m/s)	$v_y^o(0)$ (m/s)
1	(0, 30)	0.0	0.0
2	(60, 20)	7.0	0.0
3	(50, 50)	0.0	5.5
4	(20, 40)	0.0	1.0
5	(20, 50)	0.0	1.0
6	(60, 20)	0.0	−8.0
7	(70, 40)	10	0
8	(40, 30)	0	0
9	(70, 60)	0	0
10	(80, 80)	0	0

In Table A1, $v_x^o(0)$ and $v_y^o(0)$ denote the initial horizontal and longitudinal velocity of the obstacles, respectively. Once the motion of obstacles extends beyond the rectangular boundary that originated from the coordinates (20, 20) to (80, 80), the subsequent speed will be set to:

$$\begin{cases} v_x^o(t) = -0.7v_x^o(0) \\ v_y^o(t) = -0.7v_y^o(0) \end{cases} \tag{A1}$$

References

1. Feng, S.; Sun, H.; Yan, X.; Zhu, H.; Zou, Z.; Shen, Y.; Liu, H. Dense reinforcement learning for safety validation of autonomous vehicles. *Nature* **2023**, *615*, 620–627. [CrossRef] [PubMed]
2. Thoresen, M.; Nielsen, N.; Mathiassen, K.; Pettersen, K. Path planning for UGVs based on traversability hybrid A. *IEEE Robot. Autom. Lett.* **2021**, *6*, 1216–1223. [CrossRef]
3. Liu, J.; Anavatti, S.; Garratt, M.; Abbass, H. Modified continuous ant colony optimisation for multiple unmanned ground vehicle path planning. *Expert Syst. Appl.* **2022**, *196*, 116605. [CrossRef]
4. Zhou, X.; Yu, X.; Zhang, Y.; Luo, Y.; Peng, X. Trajectory planning and tracking strategy applied to an unmanned ground vehicle in the presence of obstacles. *IEEE Trans. Autom. Sci. Eng.* **2020**, *18*, 1575–1589. [CrossRef]

5. Chen, D.; Wang, Z.; Zhou, G.; Li, S. Path planning and energy efficiency of heterogeneous mobile robots using Cuckoo–beetle swarm search algorithms with applications in UGV obstacle avoidance. *Sustainability* **2022**, *14*, 15137. [CrossRef]
6. Baras, N.; Dasygenis, M. UGV Coverage Path Planning: An Energy-Efficient Approach through Turn Reduction. *Electronics* **2023**, *12*, 2959. [CrossRef]
7. Asadi, K.; Suresh, A.K.; Ender, A.; Gotad, S.; Maniyar, S.; Anand, S. An integrated UGV-UAV system for construction site data collection. *Autom. Constr.* **2020**, *112*, 103068. [CrossRef]
8. Pan, L.; Zhang, Y. Scene image classifying via the partially connected neural network. In Proceedings of the 2010 5th International Conference on Computer Science & Education, Hefei, China, 24–27 August 2010; Volume 4, pp. 25–33.
9. Wang, P.; Gao, S.; Li, L.; Sun, B.; Cheng, S. Obstacle avoidance path planning design for autonomous driving vehicles based on an improved artificial potential field algorithm. *Energies* **2019**, *12*, 2342. [CrossRef]
10. Wahid, N.; Zamzuri, H.; Amer, N.H.; Dwijotomo, A.; Saruchi, S.; Mazlan, S. Vehicle collision avoidance motion planning strategy using artificial potential field with adaptive multi-speed scheduler. *IET Intell. Transp. Syst.* **2020**, *14*, 1200–1209. [CrossRef]
11. Huang, Y.; Ding, H.; Zhang, Y.; Wang, H.; Cao, D.; Xu, N. A motion planning and tracking framework for autonomous vehicles based on artificial potential field elaborated resistance network approach. *IEEE Trans. Ind. Electron.* **2019**, *67*, 1376–1386. [CrossRef]
12. Rostami, S.; Sangaiah, A.; Wang, J.; Liu, X. Obstacle avoidance of mobile robots using modified artificial potential field algorithm. *EURASIP J. Wirel. Commun. Netw.* **2019**, *2019*, 70. [CrossRef]
13. Chen, X.; Zhao, M.; Yin, L. Dynamic path planning of the UAV avoiding static and moving obstacles. *J. Intell. Robot. Syst.* **2020**, *99*, 909–931. [CrossRef]
14. Feng, J.; Zhang, J.; Zhang, G.; Xie, S.; Ding, Y.; Liu, Z. UAV dynamic path planning based on obstacle position prediction in an unknown environment. *IEEE Access* **2021**, *9*, 154679–154691. [CrossRef]
15. Zhou, Y.; Su, Y.; Xie, A.; Kong, L. A newly bio-inspired path planning algorithm for autonomous obstacle avoidance of UAV. *Chin. J. Aeronaut.* **2021**, *34*, 199–209. [CrossRef]
16. Pham, T.; Dao, P. Disturbance observer-based adaptive reinforcement learning for perturbed uncertain surface vessels. *ISA Trans.* **2022**, *130*, 277–292.
17. Chen, L.; Dai, S.; Dong, C. Adaptive optimal tracking control of an underactuated surface vessel using Actor–Critic reinforcement learning. *IEEE Trans. Neural Netw. Learn. Syst.* 2023, *early access*. [CrossRef]
18. Singla, A.; Padakandla, S.; Bhatnagar, S. Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 107–118. [CrossRef]
19. Kulathunga, G. A reinforcement learning based path planning approach in 3D environment. *Procedia Comput. Sci.* **2022**, *212*, 152–160. [CrossRef]
20. Nguyen, K.; Dang, V.; Pham, D.; Dao, P. Formation control scheme with reinforcement learning strategy for a group of multiple surface vehicles. *Int. J. Robust Nonlinear Control* 2023, *early access*. [CrossRef]
21. Qin, Z.; Zhu, H.; Wang, S.; Xin, Y.; Sun, J. A reinforcement learning-based near-optimal hierarchical approach for motion control: Design and experiment. *ISA Trans.* **2022**, *129*, 673–683. [CrossRef] [PubMed]
22. Fox, D.; Burgard, W.; Thrun, S. The dynamic window approach to collision avoidance. *IEEE Robot. Autom. Mag.* **1997**, *4*, 23–33. [CrossRef]
23. Jiang, F.; Dong, M.; Fan, Y.; Wang, Q. Research on motor speed control method based on the prevention of vehicle rollover. *Energies* **2022**, *15*, 3609. [CrossRef]
24. Seyedi, M.; Jung, S.; Wekezer, J.; Gepner, B. Rollover crashworthiness analyses–an overview and state of the art. *Int. J. Crashworthiness* **2020**, *25*, 328–350. [CrossRef]
25. Li, B.; Bei, S. Research method of vehicle rollover mechanism under critical instability condition. *Adv. Mech. Eng.* **2019**, *11*, 1687814018821218. [CrossRef]

sensors

MDPI

Article

TSG-SLAM: SLAM Employing Tight Coupling of Instance Segmentation and Geometric Constraints in Complex Dynamic Environments

Yongchao Zhang [1], Yuanming Li [2,3] and Pengzhan Chen [1,3,*]

[1] School of Intelligent Manufacturing, Taizhou University, Taizhou 318000, China; yczhang@tzc.edu.cn
[2] Department of Electrical Engineering, Ganzhou Polytechnic, Ganzhou 341000, China; lym_0111@163.com
[3] School of Electrical and Automation Engineering, East China Jiaotong University, Nanchang 330013, China
* Correspondence: cyxcpz@163.com

Abstract: Although numerous effective Simultaneous Localization and Mapping (SLAM) systems have been developed, complex dynamic environments continue to present challenges, such as managing moving objects and enabling robots to comprehend environments. This paper focuses on a visual SLAM method specifically designed for complex dynamic environments. Our approach proposes a dynamic feature removal module based on the tight coupling of instance segmentation and multi-view geometric constraints (TSG). This method seamlessly integrates semantic information with geometric constraint data, using the fundamental matrix as a connecting element. In particular, instance segmentation is performed on frames to eliminate all dynamic and potentially dynamic features, retaining only reliable static features for sequential feature matching and acquiring a dependable fundamental matrix. Subsequently, based on this matrix, true dynamic features are identified and removed by capitalizing on multi-view geometry constraints while preserving reliable static features for further tracking and mapping. An instance-level semantic map of the global scenario is constructed to enhance the perception and understanding of complex dynamic environments. The proposed method is assessed on TUM datasets and in real-world scenarios, demonstrating that TSG-SLAM exhibits superior performance in detecting and eliminating dynamic feature points and obtains good localization accuracy in dynamic environments.

Keywords: SLAM; complex dynamic environment; fundamental matrix; semantic segmentation; multi-view geometric constraint

Citation: Zhang, Y.; Li, Y.; Chen, P. TSG-SLAM: SLAM Employing Tight Coupling of Instance Segmentation and Geometric Constraints in Complex Dynamic Environments. *Sensors* **2023**, *23*, 9807. https://doi.org/10.3390/s23249807

Academic Editors: Yuanlong Xie, Shiqi Zheng, Zhaozheng Hu and Shuting Wang

Received: 10 October 2023
Revised: 10 December 2023
Accepted: 11 December 2023
Published: 13 December 2023

1. Introduction

SLAM technology is a crucial element for mobile robots to achieve highly intelligent tasks in unknown work environments. Visual SLAM, which relies on visual sensors to perceive surroundings, can acquire images with rich semantic information about environmental targets. Environmental semantic information is of significant importance to intelligent robots as it can assist them in positioning, build environmental semantic maps, and is the basis of human–computer interaction.

In 2007, Davison et al. [1] proposed Mono-SLAM, which achieved the realization of monocular real-time SLAM and initiated research in the field of visual SLAM. Klein et al. [2] proposed PTAM, which creatively divides the entire SLAM system into tracking and mapping threads, successfully applying feature points. Leutenegger et al. [3] proposed the OKVIS visual–inertial odometry framework, while Mur-Artal et al. proposed ORB-SLAM [4], ORB-SLAM2 [5], and ORB-SLAM3 [6] based on feature points.

Most visual SLAM systems are built based on static scenarios, and when there are moving objects in the scenario, the system's localization and mapping accuracy is greatly affected. In addition, the scene maps constructed by visual SLAM systems are usually based on the geometric information of the scene, such as sparse landmark maps and sparse

point cloud maps, which are insufficient to enable mobile robots to understand complex working environments. Thus, it is necessary to process moving objects in complex environments, eliminate their interference in the visual SLAM system, and integrate environmental semantic information to construct semantic maps.

Semantic maps include scene-oriented semantic maps and object-oriented semantic maps. The former integrate semantic information into 3D point clouds to build a scene semantic map, while the latter only retain the semantic information of some objects in the scene semantic map, with most of the semantic information independent of the map in the form of clustering. An object-oriented semantic map is more helpful for a robot to perceive a scene and improve map practicality. McCormac et al. [7] proposed a voxel-based online semantic SLAM system, Hoang et al. [8] proposed the Object-RPE system, and Hosseinzadeh et al. [9] proposed a method to represent objects in the form of quadratic surfaces, while Oberlander et al. [10] proposed a mapping method that combines topological, metric, and semantic information. Hybrid map representation, which combines topological, metric, and semantic information, has been an important direction in the field of mobile robotics research for a long time [11]. Luo et al. [12] used object recognition algorithms to classify scenes, fused the classification results with topological nodes, and assigned semantic information to each topological node. Lin et al. [13] proposed a novel closed-loop approach based on object modeling and semantic graph matching. Object-level features in a scene are modeled using voxels and cuboids, and the scene is further represented as a semantic graph with topological information. Yang et al. [14] proposed a semantic and topological method of automatically representing indoor spaces using floor-plan raster maps to reconstruct indoor spaces with semantic and topological structures. The dynamic visual SLAM method based on the semantic segmentation module proposed by Jin et al. [15] uses semantic labels and depth images to create a 3D point cloud map with semantic information. In short, the fusion of topological and semantic information generally only fuses semantic information with topological nodes, ignoring many environmental details. Although it can help a mobile robot to move to a certain scene quickly, it cannot allow the robot to intelligently interact with the physical objects in the scene.

The rest of this paper is organized as follows: Section 2 provides an overview of related works regarding SLAM methods in dynamic environments. Section 3 describes the proposed system framework. Section 4 presents a dynamic feature removal method in detail. Section 5 presents the semantic map construction module. Section 6 presents the experimental results and performance analysis. Finally, Section 7 concludes the paper and discusses future research directions.

2. Related Work

Semantic SLAM systems face challenges in accurately localizing and mapping complex environments with a large number of dynamic objects. To address this issue, four methods [16] have been proposed to eliminate dynamic features: multi-sensor information compensation, an enhanced RANSAC algorithm, foreground/background model construction, and semantic information integration.

Using information obtained from an IMU, a wheel odometer, and other sensors as prior motion knowledge for a camera can assist a system in segmenting dynamic targets. The SLAM system designed by Yao et al. [17] includes tracking threads, feature extraction threads, and local mapping threads. One of the tasks of the tracking thread is to utilize the transformation matrix obtained from an IMU and combine it with the reprojection error to determine the dynamic nature of feature points. To avoid long-term drift in wheel odometry calculations, Yang et al. [18] only used data between two adjacent frames to estimate the initial pose over a short period of time. In order to speed up the detection of dynamic regions, two optimization measures were adopted. First, the dynamic nature of the image regions obtained via clustering was determined instead of individual pixels. Second, when judging the clustered regions, only a subset of feature points was selected instead of all feature points in the region. In addition, the object detection framework

YOLOv3 [19] was integrated into the system. The data fusion and joint calibration resulting from the use of multiple sensors pose challenges to the system's stability and accuracy, while errors and noise are also issues that cannot be neglected.

For SLAM in dynamic environments, it is common to use the RANSAC algorithm to obtain a rough transformation matrix result to determine the dynamic status of landmarks in the environment. Sun et al. [20] used tthe RANSAC algorithm to compute a perspective transformation. The calculated transformation matrix was applied to each pixel in the previous frame image, resulting in a difference image. The moving object can be roughly identified using the non-zero pixels in the differencing results. This process was used to obtain segmentation results. Sun et al. [21] replaced the standard RANSAC algorithm with a least-median-of-squares algorithm (LMedS) for their calculations. The Prior-based Adaptive Random Sample Consensus (PARSAC) algorithm proposed by Tan et al. [22] leveraged prior knowledge about the background, leading to a reduced proportion of inliers in the estimation of the camera's motion model. The major drawback of using the RANSAC algorithm to eliminate dynamic objects is that as the number of dynamic objects increases or they become closer to the camera, the static background area becomes too small, leading to ineffective dynamic object removal.

Building a foreground model in advance is equivalent to directly segmenting out moving objects. Then, the areas outside the foreground are used for the localization and mapping of mobile robots. Wang et al. [23] first calculated optical flow trajectories between consecutive image frames. They performed a clustering analysis on these trajectories, merging regions with similar motion tendencies. Assuming that static regions dominate the majority of the image, the largest area with merged regions could be used to compute the corresponding fundamental matrix. This process further refined the dynamic regions, forming the image foreground. In subsequent camera localization and dense mapping using the Dense Visual Odometry (DVO) SLAM system [24], the foreground parts of the images were discarded. The challenge in constructing foreground models lies in the identification and removal of non-rigid bodies, such as pedestrians and animals.

Adding semantic information to the SLAM system in dynamic environments allows for the preliminary assessment and segmentation of objects with high motion probability based on prior knowledge. By removing these high-motion-probability target regions from the images, estimating the camera's motion and pose becomes much more reliable compared to directly using the RANSAC algorithm to remove outliers. Yu et al. [25] introduced the DS-SLAM system, which excluded the person area in the image and eliminated dynamic matching points using motion consistency. Bescos et al. proposed DynaSLAM [26] and DynaSLAM II [27]. DynaSLAM combined multi-view geometry and target masks to remove predefined moving objects and proposed a background restoration method to fix occluded backgrounds. Dyna-SLAM II simultaneously estimated camera poses, sparse static 3D maps, and the trajectories of multiple moving objects using a new bundle adjustment method. You et al. [28] proposed a multimodal semantic SLAM system (MISDSLAM) which can reconstruct the static background with semantic information. Liu et al. [29] applied an algorithm to obtain as the latest semantic information possible, thereby making it possible to use segmentation methods with different speeds in a uniform way. Zhao et al. [30] proposed KSF-SLAM, which added an efficient semantic tracking module to remove dynamic objects in dynamic environments. Gonzalez et al. [31] introduced TwistSLAM, which created point clusters based on semantic categories and modeled constraints between clusters to remove dynamic features and improve motion estimation quality. Kuang et al. [32] obtained potential motion areas through semantic segmentation, combined dynamic point features to determine dynamic areas, and removed point and line features in dynamic areas to enhance localization accuracy and stability. Runz et al. [33] presented the mask fusion system, which used geometric segmentation to produce precise object boundaries to overcome the limitations of imperfect boundaries provided by semantic segmentation. Xu et al. [34] proposed the MID-Fusion system, which provided the geometric, semantic, and motion attributes of objects in an environment. Li et al. [35] presented the DP-SLAM

system, which tracked dynamic matching points in a Bayesian probability estimation framework to overcome geometric constraints and semantic segmentation bias. Wu et al. [36] proposed YOLO-SLAM, which combined object detection and geometric constraint methods to reduce the influence of dynamic objects. Li et al. [37] utilized semantic information and global dense optical flow as constraints to generate dynamic-static masks and eliminate dynamic objects. Xing et al. [38] presented DE-SLAM, which utilized a dynamic detection and tracking module of semantic and metric information to improve localization accuracy by eliminating features on dynamic objects.

In the aforementioned literature, most dynamic visual SLAM schemes adopted existing mature object detection and semantic segmentation network frameworks to perform the initial division of dynamic regions. The network architectures included SegNet [39], Mask R-CNN [40], and the YOLO series. Similar to Mask R-CNN, SOLOv2 [41] is a simple, fast, and accurate instance segmentation framework. It surpasses most current advanced open-source instance segmentation methods in terms of segmentation speed and accuracy, and it also performs well in segmenting moving targets. Hence, we chose SOLOv2 to complete the instance segmentation task. Based on instance-level semantic information, we can acquire the prior motion information of objects. If the label corresponding to the object feature is person, it is considered dynamic, and if the label is desk, it is considered static, both with high confidence. However, when the label is chair, it is usually static, but there is a significant likelihood it might move due to the influence of other moving objects (such as human activity). Therefore, it is challenging to definitively categorize chairs as either static or dynamic; these are potentially dynamic objects.

In conventional semantic segmentation-based dynamic SLAM systems, dynamic features are removed, static features are preserved, and potentially dynamic features are generally treated either as all static or as all dynamic. Treating all potentially dynamic features as dynamic and removing them can reduce the accuracy of feature matching. Conversely, treating all potentially dynamic features as static can lead to many incorrect correspondences in feature matching. Both situations negatively affect the system's localization accuracy and mapping precision. In essence, while the SOLOv2 algorithm can segment potential dynamic targets and provide semantic labels, it cannot accurately determine their actual motion state. In addition, the unavoidable fuzziness in SOLOv2's segmentation results near object edges can lead to a small number of feature points being misjudged at the edges where dynamic and static objects meet. Therefore, we cannot rely solely on SOLOv2's semantic information and need to combine it with other methods to jointly determine the motion state of target features.

Instance segmentation methods are used in conjunction with other methods, such as using a bundle adjustment with multi-view geometric constraints or optical flow fields. When combined, a voting mechanism is typically employed to process dynamic objects. Generally, there are two types of voting mechanism: the first is that if both judgment results are dynamic, the final result is dynamic, and the second is that if any one result is dynamic, the final result is dynamic. We consider both combination methods loosely coupled approaches, merely combining the results of the two methods through a simple mechanism. In fact, this loosely coupled approach is unreliable and can lead to misjudgment.

This paper proposes a dynamic feature removal method that tightly couples instance segmentation and multi-view geometric constraints to detect and remove dynamic feature points and integrates instance semantic information into environment map construction to generate global environment instance-level semantic point cloud maps. The main contributions of this paper are as follows. First, a system framework for the SLAM of mobile robots in complex environments is constructed based on ORB-SLAM3. Second, a dynamic feature removal method is designed which uses a tightly coupled method to closely combine the instance segmentation SOLOv2 algorithm with multi-view geometric constraints to accurately detect and remove dynamic feature points. Third, a semantic map construction module is designed, which extracts a 3D semantic point cloud using the semantic information of the target obtained via the instance segmentation algorithm, generates the

corresponding target semantic tag, and builds an instance-level target semantic tag library to construct an environmental 3D semantic map.

3. System Overview

To eliminate the impact of dynamic targets on the SLAM system and create a map containing environmental semantic information, this paper presents a mobile robot simultaneous localization and semantic mapping system based on the ORB-SLAM3 system. The system's overall framework is illustrated in Figure 1, and it can handle dynamic targets with excellent anti-interference ability, extract the instance-level semantic information of various objects, and support intelligent robots to perform tasks in complex indoor environments.

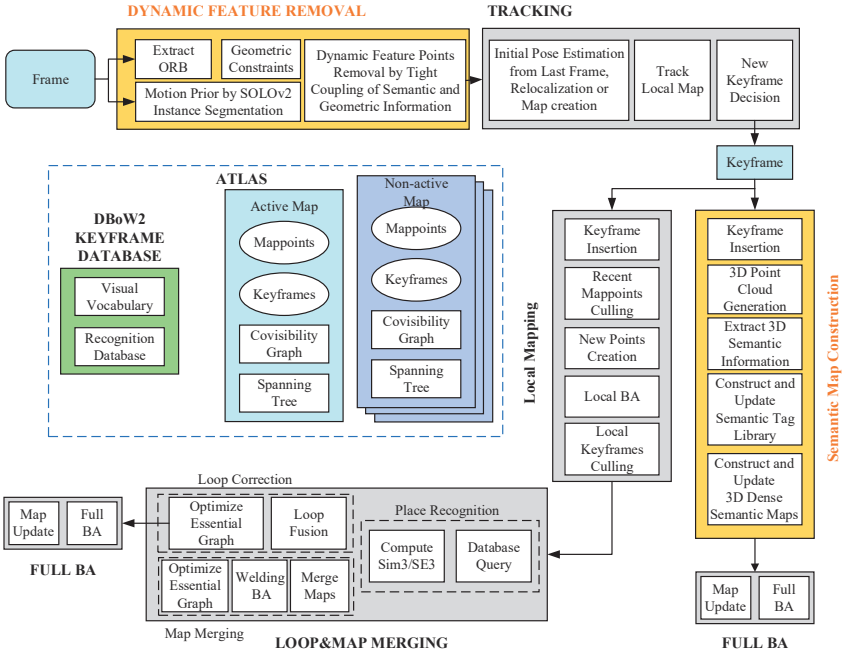


Figure 1. Overall SLAM system framework.

The TSG-SLAM system introduces two additional parallel threads to the classic three threads of ORB-SLAM3: the dynamic feature removal thread and the semantic map construction thread. The dynamic feature removal thread is responsible for eliminating the dynamic features of objects, ensuring the system's localization and mapping accuracy. The semantic map construction thread constructs a 3D dense semantic map with instance-level semantic information, which enables intelligent robots to navigate and interact intelligently in complex environments.

4. Dynamic Feature Removal Method

A dynamic feature detection and removal method is proposed in this paper. The method tightly integrates semantic information and multi-view geometric constraint information, as shown in the algorithm framework in Figure 2.

Firstly, ORB features are extracted from the current frame image, and instance segmentation results are obtained using SOLOv2 on both the current and previous frames. This allows for the removal of features belonging to dynamic targets and potential dynamic targets. Subsequently, feature matching is performed based on the remaining static targets, and the fundamental matrix is calculated. Finally, dynamic feature points in the current

frame are precisely detected and removed with multi-view geometric constraints, leaving only the static feature points.

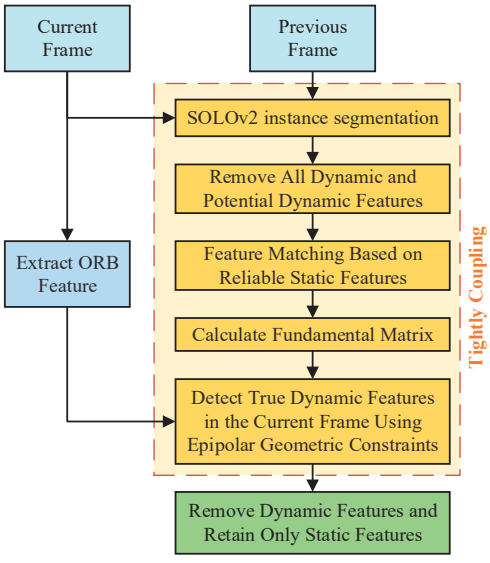


Figure 2. Framework of dynamic feature removal algorithm.

This method ensures accurate localization and mapping in complex environments by removing the impact of dynamic targets and contributes to constructing a 3D dense semantic map with instance-level semantic information for intelligent tasks such as navigation and interaction in complex indoor environments.

4.1. SOLOv2 Instance Segmentation

The SOLOv2 network architecture is shown in Figure 3, and it mainly comprises Fully Convolutional Network (FCN) feature extraction, a kernel branch, and a feature branch. The convolution kernel matrix is denoted as G, while the mask feature matrix is represented by F. SOLOv2 divides the image into S × S grids, treating each grid as a potential target instance. After the original image is passed through the FCN, the feature map is obtained, which then enters both the kernel branch and the feature branch. The kernel branch predicts the dynamic kernel to obtain different kernels for different inputs, while the feature branch predicts the features for each point on the feature map. Finally, the outputs of the kernel branch and feature branch are convolved to obtain the mask of the target in the image.

The COCO dataset [42] is used for pre-training to obtain network parameters, which include most moving objects that may appear in real-life scenarios, making it very suitable for the application scenario of this article.

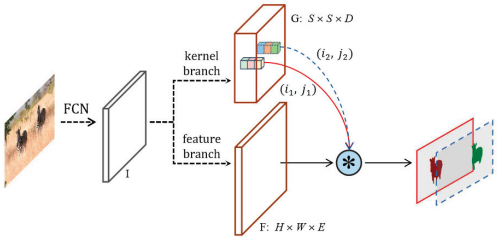


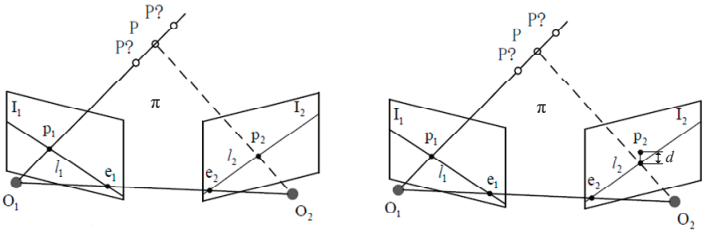
Figure 3. Framework of SOLOv2 network.

4.2. Dynamic Feature Detection

The multi-view geometry constraints utilizing epipolar geometry characteristics can be used to detect the motion state of target feature points in the environment. The features that satisfy the epipolar constraints are static features, while the features that do not satisfy the epipolar constraints are dynamic features. These constraints can be used to identify the position and motion of feature points in a given environment.

Figure 4a shows the relationship between static object points and their corresponding feature points in two frames. P is a static target point which is imaged in two consecutive image frames corresponding to feature points p_1 and p_2 in frame I_1 and frame I_2, respectively. P? represents the possible position of point P in the presence of uncertain factors. O_1 and O_2 are the camera centers corresponding to frame I_1 and frame I_2, respectively. Polar plane π intersects image planes I_1 and I_2 at polar lines l_1 and l_2, respectively, and baseline O_1O_2 intersects image planes I_1 and I_2 at poles e_1 and e_2, respectively. P lies on rays $\overrightarrow{O_1p_1}$ and $\overrightarrow{O_2p_2}$, and p_2 lies on epipolar line l_2. The multi-view geometric constraint describes the corresponding epipolar mapping from the points on frame I_1 and frame I_2, and the mapping relationship can be described by the fundamental matrix F_m.

$$p_2^T F_m p_1 = 0 \tag{1}$$



(a) Ideal Epipolar Geometric Constraints

(b) Epipolar Geometric Constraints with Uncertain Factors

Figure 4. Multi-view epipolar geometry constraints.

Given p_1 in frame I_1 and the fundamental matrix F_m, Equation (1) provides the constraints that p_2 must satisfy when P is a static target point. Therefore, we can use this constraint to judge whether the target point corresponding to the ORB feature point is dynamic. Due to the uncertainty in the process of extracting features and estimating F_m, there is a high probability that the two image points in the static map do not strictly satisfy Equation (1), that is, p_2 in Figure 4b should be located on l_2. If the distance d between p_2 and l_2 is smaller than a predetermined threshold, the motion state of the target point corresponding to the image point is regarded as static; otherwise, it is regarded as dynamic.

Use the classic eight-point algorithm to estimate the fundamental matrix F_m. Let $F_m = \begin{pmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{pmatrix}$, $p_1 = [u_1, v_1, 1]^{\mathrm{T}}$,$p_2 = [u_2, v_2, 1]^{\mathrm{T}}$, where (u_1, v_1) and (u_2, v_2) are the pixel coordinates of p_1 and p_2, respectively. According to Equation (1), we can obtain

$$(u_2, v_2, 1) \begin{pmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{pmatrix} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} = 0 \tag{2}$$

Let f_m denote the vector containing all elements of F_m.

$$f_m = [f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9]^{\mathrm{T}} \tag{3}$$

Equation (2) can be written as a linear equation about f_m.

$$[u_2u_1, u_2v_1, u_2, v_2u_1, v_2v_1, v_2, u_1, v_1, 1] \cdot f_m = 0 \tag{4}$$

When there are eight pairs of corresponding image points between two consecutive frames, we can solve Equation (4) to calculate F_m. Once F_m is obtained, we can use Equation (2) to determine the state of the target feature.

While using multi-view epipolar geometry constraints to detect dynamic features is a useful approach, it presents a fundamental contradiction. In order to calculate the fundamental matrix F_m required to detect dynamic feature points, correspondences of static feature points are needed as the target features in the keyframes used for feature matching must be static. This means that dynamic feature points must be removed before F_m can be calculated. In the general feature-matching process, the Random Sample Consensus (RANSAC) algorithm is often used to filter out dynamic feature points and reduce the impact of incorrect correspondences. However, RANSAC is limited in its ability to remove a large number of dynamic feature points, which can negatively affect its overall performance.

4.3. Dynamic Feature Removal

We propose a tight-coupling approach that utilizes the fundamental matrix F_m as a bridge between semantic information and geometric constraint information. Firstly, we employ SOLOv2 to perform instance segmentation on frames of the scenario to obtain motion priors, which identify all moving and potential moving targets. We then use the instance segmentation results as a mask to remove the correspondences of dynamic and potential dynamic features, retaining only reliable static feature correspondences. Based on these static feature correspondences, we perform feature matching and compute the reliable fundamental matrix. Finally, we use multi-view geometry constraints to detect and remove true dynamic features, retaining only static feature points for subsequent tracking and mapping. When judging the motion state of a feature point, we use a threshold value of d. If d exceeds one pixel size, the feature point is judged as dynamic and removed, and if d is smaller than a pixel size, it is judged as static and retained.

5. Instance-Level Semantic Map Construction

Figure 5 illustrates the framework of our scenario semantic map construction algorithm. The algorithm constructs the semantic map using keyframe images with dynamic feature points removed. Firstly, we generate a single-frame point cloud containing only static feature points from the keyframe images. Then, we stitch and filter the generated single-frame point clouds to obtain the scene point cloud map. Next, we use the semantic information and masks provided by SOLOv2 to extract the 3D semantic tags of the targets from the point clouds, establishing and updating an instance-level semantic tag library. Finally, we integrate the semantic information of the targets into the point cloud map to generate a 3D semantic point cloud map. To accommodate larger scenes and conserve storage space, we construct an octree semantic map.

An RGB-D camera captures both color and depth information for each pixel in the scene. By modeling the camera and using its intrinsic and extrinsic parameters, we can map the 2D pixels in the image to their corresponding 3D points in space, creating a point cloud. For a given frame, let (x, y) be the 2D coordinates of a pixel p, (X, Y, Z) be the 3D coordinates of the corresponding spatial point P, and s be the depth value of p. The transformation relationship between p and P can be expressed as follows:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x \\ y \\ s \end{bmatrix} \tag{5}$$

The coordinates of P are

$$\begin{cases} X = s(x - c_x)/f_x \\ Y = s(y - c_y)/f_y \\ Z = s \end{cases}, \tag{6}$$

where f_x and f_y are the focal length of the RGB-D camera, and c_x and c_y are the offsets of the image origin relative to the imaging point of the camera's optical center. By applying this transformation to the pixels in the key image frames, we can obtain the corresponding point cloud.

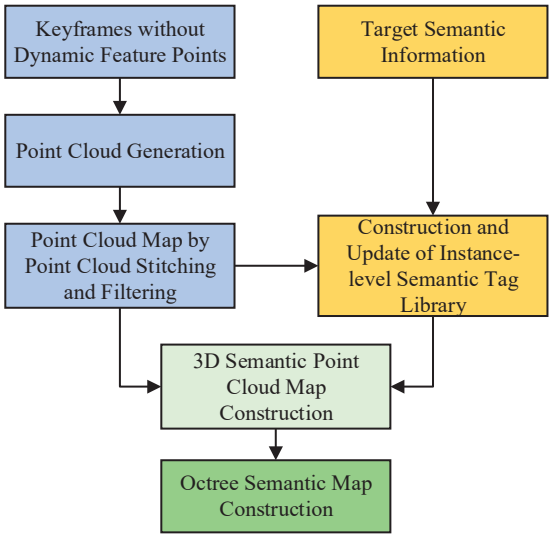


Figure 5. Framework of scenario semantic map construction algorithm.

To match and stitch point clouds, we utilize the PCL (Point Cloud Library) [43] and follow a three-step process. First, we find the point cloud that corresponds to a certain frame and match them. Second, we calculate the transformation matrix between the two point clouds. Finally, we transform the matched point clouds into the same coordinate system and stitch them together, resulting in a complete point cloud map of the scene.

The mathematical expression for point cloud stitching can be described as follows:

$$m = \sum_{i=0}^{n} T_i C_i, \tag{7}$$

where m is the local point cloud map obtained by generating and stitching the first n image frames. C_i represents the point cloud obtained from the i-th keyframe, and T_i represents the position and orientation of the camera corresponding to the i-th keyframe.

To remove outlier noise points from the point cloud map, a statistical filter is employed to filter the point cloud map to remove these outlier noise points. To address the issue of overlapping points obtained from different viewing angles while preserving the shape characteristics of the point cloud map, a voxel filter is used to remove these overlapping points.

Although we have generated a global 3D point cloud map of the scene through single-frame point cloud generation, point cloud stitching, and filtering, this point cloud map is simply geometry-based and does not incorporate the semantic information of targets. As a result, it cannot provide a deeper understanding of the scene for mobile robots. Therefore, we designed an algorithm for constructing and updating an instance-level Semantic Tag Library (STL), presented in Algorithm 1. Firstly, we extract and optimize the 3D point

cloud corresponding to each target to generate the corresponding 3D semantic tag. Then, we match and fuse the semantic tags corresponding to the 3D point clouds extracted from the same target in different perspectives. Finally, we construct and update the global static target semantic tag library of the scene.

Algorithm 1 Algorithm of instance-level STL.

Input: semantic information and 3D point cloud of targets
Output: instance-level STL

```
for each 3D point cloud do
  if an unprocessed point cloud exists then
    extract & optimize point cloud for semantic tag
    if semantic tag exists in tag library then
      calculate spatial consistency of point cloud
      if d_min < d_w then
        fuse & update STL
        go to step 18
      else
        insert the tag into STL
        go to step 18
      end
    else
      insert the tag into STL
      go to step 18
    end
  else
    save STL
  end
end for
```

Since the SOLOv2 instance segmentation algorithm can accurately segment the target area, the resulting target mask area is highly precise and contains only pixels corresponding to the target. This makes it possible to map the target semantic information obtained from the segmentation directly to the 3D point cloud, resulting in a 3D point cloud corresponding to each target. The detailed process involves locating the region of each segmented target instance using the 2D mask, recording the index of the corresponding point cloud for each pixel in the mask area that matches the semantic mask category, calculating the average depth of the point cloud in the target mask area, removing outlier points, and performing statistical and voxel filtering on the point cloud index corresponding to each target. Instance-level semantic tags are then generated based on the semantic information and corresponding 3D point clouds.

To update the target semantic tag library, the target semantic tags generated from the segmentation are compared with existing tags in the library. If a tag with the same category does not exist, it is added to the library. If it does exist, a spatial consistency calculation is performed on the 3D point cloud, and if the minimum Euclidean distance $d_{\min}$ between the centers of the point clouds is less than the average width d_{w} of the two candidate boxes, they are considered the same target, and the target semantic tag library is fused and updated. Otherwise, the target semantic tag is inserted into the library. The library updating process involves merging similar targets' point clouds and recalculating their center, maximum, and minimum point coordinates.

After constructing and updating the semantic tag library, the 3D point cloud map contains the instance-level semantic information of each target in the scenario. However, it also contains invalid information, such as textures on the ground and shadows in shadowed areas, which could overload the computing resources. Therefore, to achieve the localization and mapping of larger-scale scenes, a visual 3D octree semantic map is established by

performing point cloud semantic extraction on each keyframe and matching and fusing target point clouds generated from different observations at different positions.

Suppose a certain node in the map is denoted as n, its observed value is z, and the probability log value of the node from the beginning to the time t is $L(n|z_{1:t})$; then, the probability log value at the time $t+1$ is

$$L(n|z_{1:t+1}) = L(n|z_{1:t-1}) + L(n|z_t), \tag{8}$$

written in probabilistic form as

$$P(n|z_{1:t}) = \left[1 + \frac{1 - P(n|z_t)}{P(n|z_t)} \cdot \frac{1 - P(n|z_{1:t-1})}{P(n|z_{1:t-1})} \cdot \frac{P(n)}{1 - P(n)}\right]^{-1} \tag{9}$$

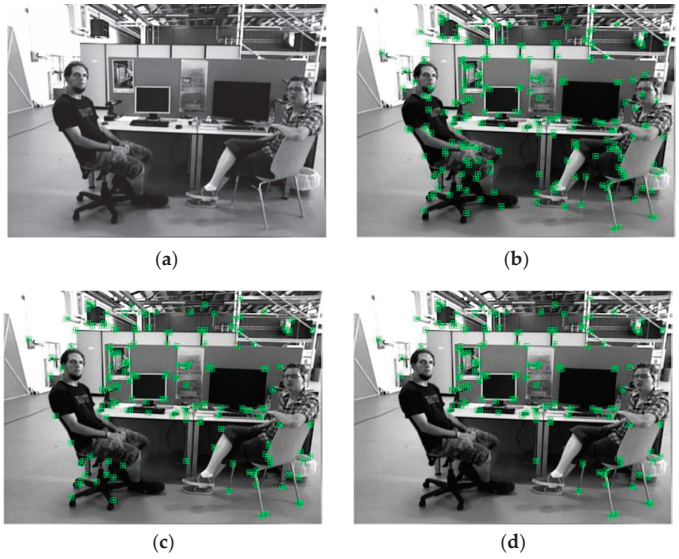
With the help of log probabilities, we can effectively combine and enhance the entire octree map using RGB-D data. When the depth information of each pixel is converted into point cloud data, the 3D point cloud representing the target will be contained within the limits of the corresponding octree sub-node. By increasing the occupancy probability of that node, we can obtain the occupancy information of the node. Furthermore, by assigning target semantic RGB color values to each node of the octree, we can create a highly visual octree semantic map.

6. Experiment and Discussion

6.1. Test of Dynamic Feature Removal Method

To analyze the effectiveness of the dynamic feature removal method, we selected the fr3/walking_xyz dataset from the TUM dataset [44] for testing. This dataset scenario is similar to our daily work environment, as shown in Figure 6a, with static objects such as tables and monitors, dynamic targets such as people, and potential dynamic targets such as chairs. Two monitors are static, two people are dynamic, and the chair on the left moves due to the movement of the person and is dynamic, while the chair on the right has not been moved and is static. Figure 6b presents the ORB feature extraction results, with feature points marked with green dots distributed throughout the scenario. Many features are extracted from objects with distinctive features such as people, chairs, and monitors, which contain numerous dynamic features, such as people. In Figure 6c, we show the results of removing dynamic features only with the SOLOv2 instance segmentation method. Although most of the ORB dynamic features on the two people are removed, a few feature points remain at the contact edge between the people and the chair, limited by the accuracy of the instance segmentation algorithm. It is difficult to perfectly segment features at a contact edge, and the SOLOv2 algorithm used in this paper, despite having high segmentation accuracy for object edges, still has some unavoidable errors. Additionally, the feature points on the two chairs are not removed since the real state of the potential dynamic target cannot be accurately distinguished only based on instance segmentation. Therefore, the two chairs are simply judged as static targets. Figure 6d illustrates the results of our tightly coupled method for removing dynamic features. Compared to Figure 6c, almost all the features of the two moving people are removed, indicating that our method is more effective at reducing the segmentation error of the instance segmentation algorithm. The features of the chair on the right are judged as static features and preserved, while the features of the chair on the left are judged to be dynamic and removed, indicating that our method accurately removes the features of all dynamic objects in the scenario. This is consistent with the motion state of each target in the test dataset. Table 1 lists the number of different types of feature points obtained by different methods.

Our findings confirm the effectiveness of our method in accurately eliminating dynamic features and minimizing segmentation errors in instance segmentation algorithms.



(**a**) (**b**)

(**c**) (**d**)

Figure 6. Comparison of dynamic feature removal method. (**a**) Original grayscale image. (**b**) ORB feature extraction. (**c**) Dynamic feature removal based on instance segmentation. (**d**) Dynamic feature removal based on tightly coupled method.

Table 1. Comparison of the number of different types of feature points.

Dynamic Feature Removal Method	Type of Feature Points		
	Static	Dynamic	Potentially Dynamic
ORB feature extraction	144	54	15
Instance segmentation	144	0	15
Tightly coupled method	144	0	0

6.2. Test of Semantic Map Construction Algorithm

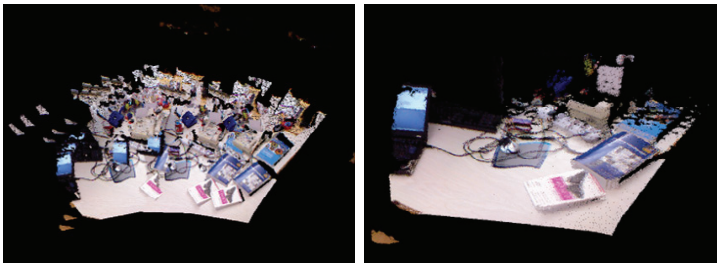
We utilized the partial sequence located in the fr1/room of the TUM dataset to conduct our local semantic mapping evaluation. This dataset provides us with RGB images, depth images, and the precise position and orientation of the camera. For our mapping test, we selected five consecutive frames of images. Figure 7a shows the RGB images of the selected frames, while Figure 7b displays the corresponding depth images.

(a) RGB image

(b) depth image

Figure 7. Selected image frame sequence diagram.

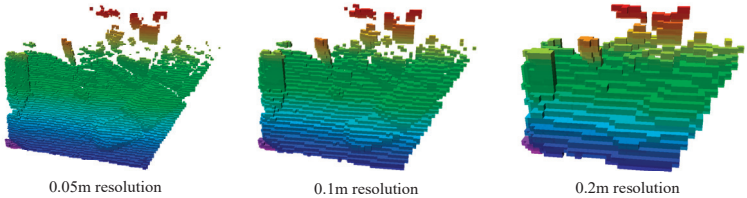
To create our point cloud map, we stitched and filtered all of the single-frame point clouds generated from extraction. The point cloud map before and after filtering is depicted in Figure 8, where Figure 8a,b illustrate the point cloud map before and after filtering, respectively. Prior to filtering, the point cloud map exhibited a significant amount of overlap between point clouds and contained numerous outlier noise points. However, after filtering, the quality of the point cloud map was significantly enhanced. Our experimental statistical analysis revealed that the numbers of point clouds generated before and after filtering were 1,118,657 and 634,787, respectively, representing a reduction of almost half of the total number of point clouds. This highlights how filtering can effectively improve mapping outcomes and greatly conserve computing resources.



(a) Before filtering (b) After filtering

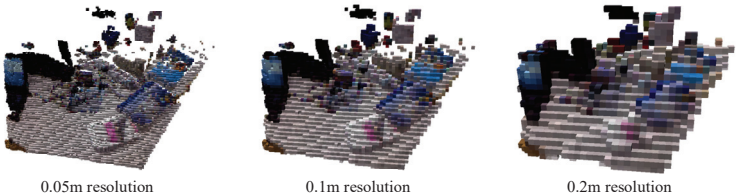
Figure 8. Comparison of point cloud map before and after filtering.

Figures 9 and 10 display the reconstruction results of the octree maps before and after integrating target semantic color information. In Figure 9, the octree map is annotated with a gradient color scheme without a specific pattern. In Figure 10, the octree map with the added target semantic color information contains visualized semantic information of the scenario's targets in Figure 8b, which significantly enhances the scenario reconstruction and produces a visually compelling result. Additionally, Figures 9 and 10 demonstrate the impact of the octree map at varying resolutions. In our testing, we used a default depth of 16 layers, with an edge length of each small square measuring 0.05 m. As the depth decreases by one layer, the leaf nodes of the octree move up one layer, and the edge length of each small square doubles.



0.05m resolution 0.1m resolution 0.2m resolution

Figure 9. Octree map before integrating target semantic color information.



0.05m resolution 0.1m resolution 0.2m resolution

Figure 10. Octree map after integrating target semantic color information.

Moreover, the file sizes of the point cloud map and the octree map are 10.2 megabytes and 217.8 kilobytes, respectively. This represents a significant reduction in storage space of nearly fifty times, highlighting the benefits of using an octree map for reconstructing larger scenarios.

6.3. Experimental Platform and Evaluation Index

To test the feasibility and effectiveness of our SLAM system in complex environments with dynamic objects, we conducted experiments in static, low-dynamic, and high-dynamic public datasets, as well as in real laboratory scenarios. In order to simplify subsequent discussions, we gave our improved SLAM system a name: TSG-SLAM.

An experimental platform for mobile robots was developed to meet the demands of complex environments. It includes a Mecanum wheel mobile robot, a Kinect V2 depth camera (Microsoft Inc., Redmond, WA, USA), a computer, and a vehicle-mounted power supply, among other components. This platform is depicted in Figure 11.

Figure 11. Experimental platform.

The software system is built on Ubuntu 16.04 and utilizes the ROS system for managing the entire system. The program is primarily written in the C++ language and utilizes various open-source libraries, including OpenCV for processing keyframe images, Eigen for matrix operations, Keras for instance segmentation, Ceres for solving least squares problems during optimization, g2o for graph optimization, PCL for generating point clouds, and octomap [45] for constructing octree maps.

To assess the localization accuracy of a SLAM system, the absolute trajectory error (ATE) and the relative pose error (RPE) are used as evaluation metrics to evaluate the motion trajectory estimation. ATE is employed to assess the overall accuracy of the SLAM system. The formula for calculating ATE is as follows:

$$\mathrm{ATE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \| \, trans(T_{g,i}^{-1} T_{e,i}) \, \|_2^2}, \tag{10}$$

where N is the number of frames, and $T_{g,i}$ and $T_{e,i}$ are the true position value and evaluated position value of the i-th frame.

The RPE metric is utilized to assess the local accuracy of the trajectory estimation and the position estimation drift of the SLAM system within a certain fixed time. Within a fixed time interval t, RPE can be obtained as follows:

$$\mathrm{RPE} = \sqrt{\frac{1}{N-\Delta t}\sum_{i=1}^{N-\Delta t} \| \, trans((T_{g,i}^{-1} T_{g,i+\Delta t})^{-1} (T_{e,i}^{-1} T_{e,i+\Delta t})) \, \|_2^2} \tag{11}$$

where Δt represents the number of frames within t.

6.4. Public Dataset Experiments

To test the system, static, low-dynamic, and high-dynamic scenarios were chosen from the TUM dataset. The selected static scenarios were fr1/desk and fr1/room. The fr1/desk sequence had a smaller camera movement range, capturing mainly indoor local scenarios focused on the table and items on it. The fr1/room sequence, on the other hand, had a larger camera movement range and included most indoor spatial scenarios. To better analyze the impact of dynamic objects on localization and mapping, dynamic datasets were selected with dynamic targets as the primary subject. The low-dynamic scenarios were fr3_sitting_static and fr3_sitting_xyz, which depicted two people sitting in chairs and talking, accompanied by small movements such as waving and turning their heads. The fr3_sitting_static camera had a small range of motion, while the fr3_sitting_xyz camera had a large range of motion around the dynamic subject in the x-y-z direction. The high-dynamic scenarios were fr3_walking_static and fr3_walking_xyz, depicting two people with fast and large-scale movements. The camera movements in fr3_walking_static and fr3_walking_xyz were similar to those in fr3_sitting_static and fr3_sitting_xyz, respectively.

To compare TSG-SLAM's performance with ORB-SLAM3, we conducted an analysis of ATE and RPE data, including the mean, median, root mean square error (RMSE), and standard deviation (STD). The RMSE measures the precision of the observed values, which reflects the accuracy of the system, while the STD measures the dispersion of the observed values, which reflects the robustness of the system. Moreover, we also calculated the improvement rate of TSG-SLAM's localization performance relative to ORB-SLAM3 by using the formula below:

$$\eta = \frac{\delta_1 - \delta_T}{\delta_1} \times 100\%, \tag{12}$$

where η is the improvement rate (IR) and δ_T and δ_1 are the error of TSG-SLAM and ORB-SLAM3, respectively.

Table 2 illustrates a comparison of ATE and RPE in static scenarios. As can be observed from the table, both systems have small errors in all aspects for the local scenario fr1/desk. In the global scenario fr1/room, ATE increases significantly but still within an acceptable range, and the errors of the two systems are very close, with some errors being lower than ORB-SLAM3. ORB-SLAM3 is presently one of the most mature visual SLAM algorithms known for its high localization accuracy in static scenarios. TSG-SLAM introduces a dynamic feature removal module based on ORB-SLAM3, and its effect is not significant in static scenarios. Thus, the localization accuracy of both systems in static scenarios is quite similar.

Table 2. Comparison of ATE and RPE in static scenarios.

Evaluation Index		fr1/Desk			fr1/Room		
		ORB-SLAM3	**TSG-SLAM**	**IR**	**ORB-SLAM3**	**TSG-SLAM**	**IR**
ATE (m)	Mean	0.0178	0.0160	10.11%	0.0579	0.0512	11.57%
	Median	0.0144	0.0132	8.33%	0.0468	0.0415	11.32%
	RMSE	0.0212	0.0191	9.91%	0.0660	0.0589	10.76%
	STD	0.0114	0.0105	7.90%	0.0318	0.0297	6.60%
RPE (m)	Mean	0.0144	0.0140	2.78%	0.0146	0.0149	−2.05%
	Median	0.0097	0.0102	−5.15%	0.0113	0.0117	−3.54%
	RMSE	0.0196	0.0192	2.04%	0.0189	0.0192	−1.59%
	STD	0.0134	0.0131	2.24%	0.0119	0.0123	−3.36%

Figures 12 and 13 show a comparison of estimated trajectories and true trajectories for the fr1/desk and fr1/room sequences, respectively. The estimated trajectory closely follows the true trajectory. Therefore, in static scenarios, TSG-SLAM does not have a significant advantage in localization performance, and both systems exhibit high localization accuracy.

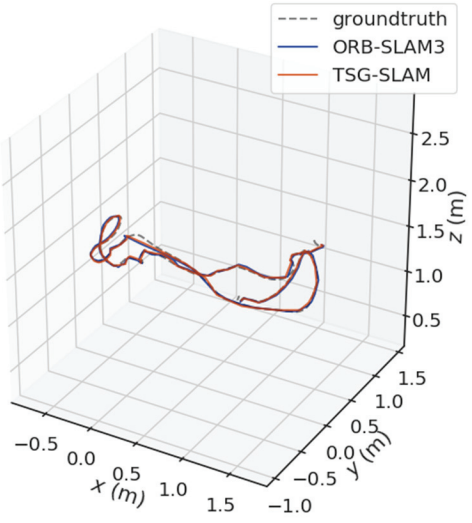


Figure 12. Estimated trajectory vs. true trajectory for fr1/desk.

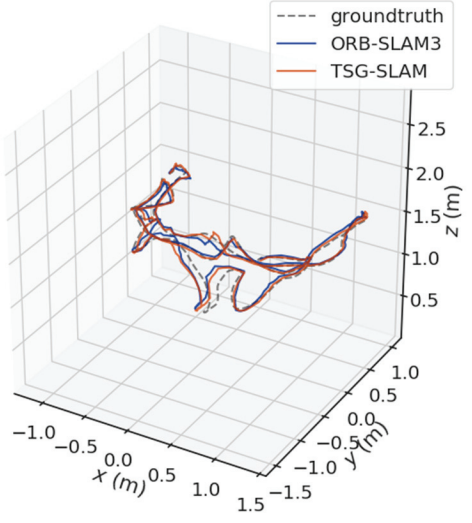


Figure 13. Estimated trajectory vs. true trajectory for fr1/room.

Table 3 compares ATE and RPE in low-dynamic scenarios. In the fr3/sitting_static sequence that focuses on dynamic targets, TSG-SLAM has smaller errors compared to ORB-SLAM3, with RMSE improvement rates of 45.45% and 34.05% for ATE and RPE, respectively. Similarly, in the fr3/sitting_xyz sequence with a larger field of view, the RMSE improvement rates also reach 39.2% and 20.71% for ATE and RPE, respectively. Figures 14 and 15 show a comparison of estimated and true trajectories for the fr3/sitting_static and fr3/sitting_xyz sequences, respectively. ORB-SLAM3 exhibits a certain deviation between the estimated and true trajectories, especially for the fr3/sitting_static sequence, while TSG-SLAM's estimated trajectory is much closer to the true trajectory. Therefore, in low-dynamic scenarios, TSG-SLAM has a definite advantage in localization, and its localization accuracy is significantly improved.

Table 3. Comparison of ATE and RPE in low dynamic scenarios.

Evaluation Index		fr3/sitting_static			fr3/sitting_xyz		
		ORB-SLAM3	TSG-SLAM	IR	ORB-SLAM3	TSG-SLAM	IR
ATE (m)	Mean	0.0143	0.0074	48.25%	0.0105	0.0064	39.05%
	Median	0.0133	0.0060	54.89%	0.0088	0.0046	47.73%
	RMSE	0.0154	0.0084	45.45%	0.0125	0.0076	39.2%
	STD	0.0058	0.0039	32.76%	0.0067	0.0049	26.86%
RPE (m)	Mean	0.0174	0.0105	39.66%	0.0161	0.0129	19.88%
	Median	0.0168	0.0087	48.21%	0.0125	0.0096	23.2%
	RMSE	0.0185	0.0122	34.05%	0.0198	0.0157	20.71%
	STD	0.0161	0.0112	30.43%	0.0116	0.009	22.41%

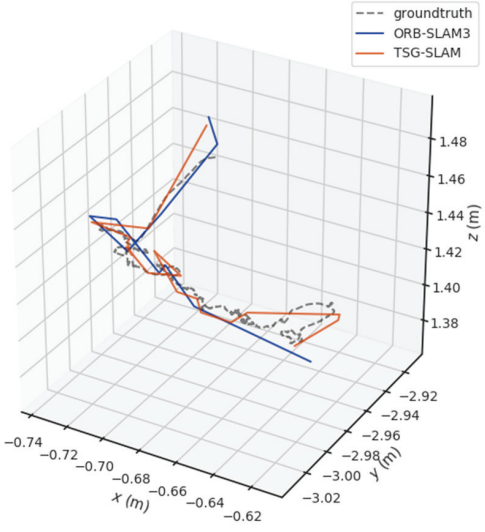


Figure 14. Estimated trajectory vs. true trajectory for fr3/sitting_static.

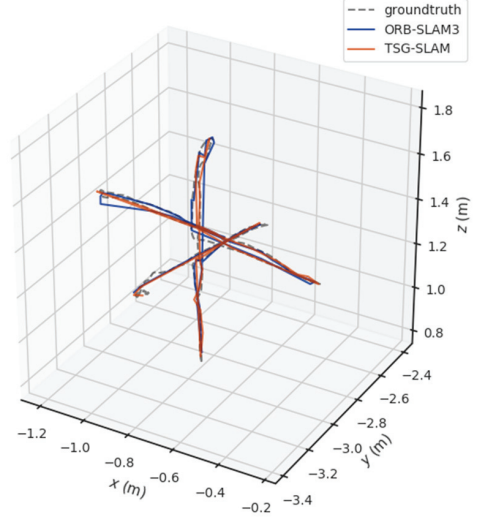


Figure 15. Estimated trajectory vs. true trajectory for fr3/sitting_xyz.

Table 4 compares ATE and RPE in high-dynamic scenarios. The errors of ORB-SLAM3 are significant, especially the RMSE of ATE, reaching 0.3832 m and 0.7123 m for the fr3/sitting_static and fr3/sitting_xyz sequences, respectively. In contrast, TSG-SLAM controls the errors well, with both the RMSE and STD improvement rates of ATE exceeding 96%. This indicates that TSG-SLAM has greatly improved global localization accuracy and stability in high dynamic scenarios, and the RMSE and STD improvement rates of RPE also exceed 55%. Figures 16 and 17 show a comparison of estimated and true trajectories for the fr3/sitting_static and fr3/sitting_xyz sequences, respectively. The estimated trajectory of ORB-SLAM3 exhibits significant deviation from the true trajectory, while the estimated trajectory of TSG-SLAM has some deviation from the true trajectory, but they are still relatively close overall. Therefore, in high-dynamic scenarios, ORB-SLAM3 is unable to function effectively, while TSG-SLAM can still function stably and has significantly improved localization accuracy.

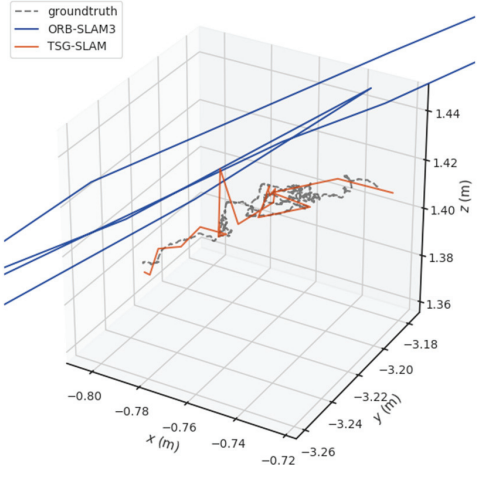


Figure 16. Estimated trajectory vs. true trajectory for fr3/walking_static.

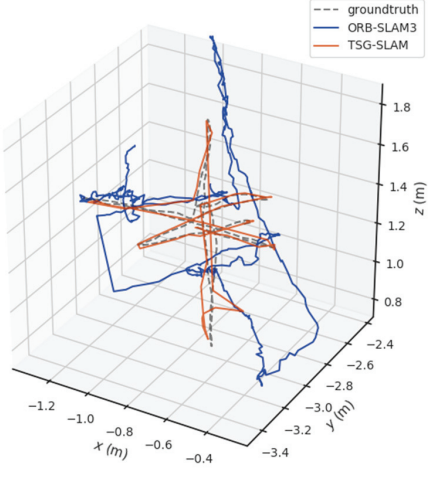


Figure 17. Estimated trajectory vs. true trajectory for fr3/walking_xyz.

Table 4. Comparison of ATE and RPE in high-dynamic scenarios.

Evaluation Index		fr3/walking_static			fr3/walking_xyz		
		ORB-SLAM3	**TSG-SLAM**	**IR**	**ORB-SLAM3**	**TSG-SLAM**	**IR**
ATE (m)	Mean	0.3697	0.0054	98.54%	0.5975	0.0228	96.18%
	Median	0.3534	0.0046	98.70%	0.5835	0.0171	97.07%
	RMSE	0.3832	0.0010	99.74%	0.7123	0.0121	98.30%
	STD	0.1009	0.0024	97.62%	0.3877	0.0121	96.87%
RPE (m)	Mean	0.0212	0.0074	65.09%	0.0311	0.0251	19.29%
	Median	0.0093	0.0049	47.31%	0.0207	0.0163	21.26%
	RMSE	0.0367	0.0094	74.39%	0.0850	0.0376	55.76%
	STD	0.0300	0.0059	80.33%	0.0790	0.0280	64.56%

To evaluate the superior localization performance of TSG-SLAM in complex environments with dynamic objects, we compared it with several dynamic SLAM systems that have shown good performance in recent years, such as DS-SLAM, DynaSLAM, MISD-SLAM, and RDS-SLAM. Since we used different computers for testing, we could not directly compare the error data obtained. Therefore, we used the relative accuracy improvement rates of these dynamic SLAM systems compared to ORB-SLAM3 as the evaluation standard for a performance comparison, specifically the RMSE and STD improvement rates of ATE.

The comparison results are presented in Table 5. In the low-dynamic scenarios of fr3/sitting_static, TSG-SLAM demonstrated a significant advantage compared to other dynamic SLAM systems. The ATE improvement rate was much higher than that of the other dynamic SLAM systems thanks to the high dynamic segmentation accuracy of the dynamic feature removal method proposed in this paper. The data for DynaSLAM are not provided in the relevant paper; therefore, no comparison could be made. In high-dynamic scenarios, all dynamic SLAM systems showed significant improvements compared to the ORB-SLAM3 system. Although TSG-SLAM had slightly lower improvement rates than some dynamic SLAM systems, it still had certain advantages overall.

Table 5. Comparison of ATE improvement rates of all SLAM system relative to ORB-SLAM3.

Dataset	Index	ATE Improvement Rate				
		DS-SLAM	**Dyna SLAM**	**MISD-SLAM**	**RDS-SLAM**	**TSG-SLAM**
fr3/walking_static	RMSE	97.76%	98.11%	63.31%	97.78%	99.74%
	STD	97.83%	97.89%	68.92%	97.37%	97.62%
fr3/walking_xyz	RMSE	97.30%	98.21%	95.54%	98.39%	98.30%
	STD	96.69%	98.23%	94.89%	98.52%	96.87%
fr3/sitting_static	RMSE	27.78%	-	11.94%	30%	45.45%
	STD	23.26%	-	24.23%	25.58%	32.76%

In summary, the TSG-SLAM system overcomes the challenges posed by moving targets in complex environments and demonstrates reliable performance in various dynamic environments with high localization accuracy and stability. It also performs comparably to other top-performing dynamic SLAM systems in certain low-dynamic scenarios and even outperforms them in terms of localization accuracy.

An experimental evaluation was conducted to assess the instance-level semantic mapping performance of TSG-SLAM in static, low-dynamic, and high-dynamic scenarios. Figure 18 depicts the results of scene semantic mapping on six datasets, including fr1/desk, fr1/room, fr3/sitting_static, fr3/sitting_xyz, fr3/walking_static, and fr3/walking_xyz. TSG-SLAM successfully constructs 3D geometric models of the objects in the scenarios and adds semantic tag color information for the objects, which enhances the map's visualization, such as the blue screen, red chair, and gray table.

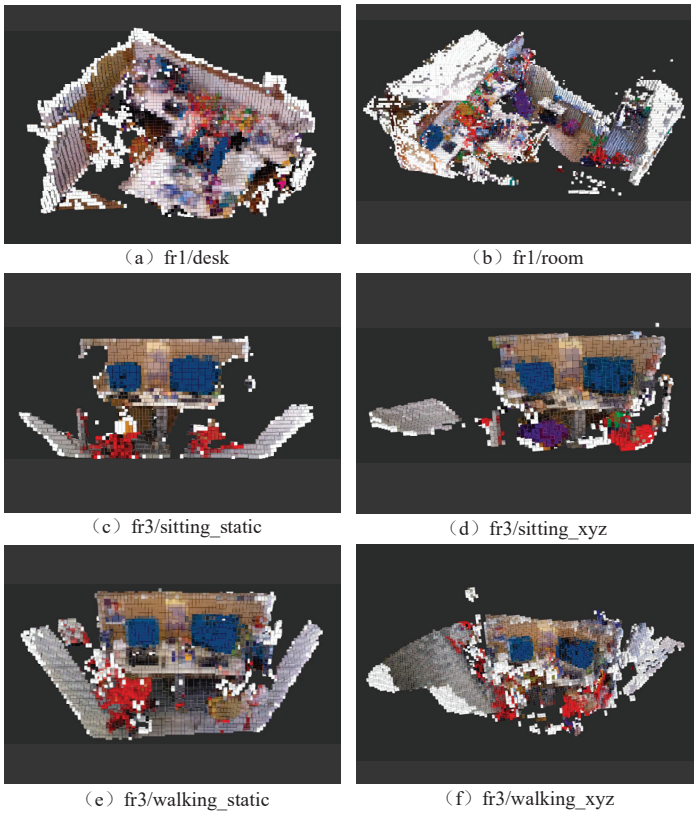


Figure 18. Octree semantic map construction results.

Figure 18a,b show the results of semantic map construction for the indoor static scenario with the desktop as the main object and the larger global static scenario, respectively. The scenarios and targets appearing in the sequence were accurately reconstructed, and semantic color information was added to the targets.

Figure 18c,d demonstrate the results of semantic map construction for low-dynamic scenarios. It can be observed that even two people only slightly moving their hands and heads were successfully segmented and most of the features on their bodies were removed, leaving only the static targets for reconstructing the scenario. Additionally, Figure 18c has a smaller reconstruction range than Figure 18d but with better results due to capturing more data from the local screen area caused by a smaller camera movement range. Meanwhile, the camera's view in Figure 18c is limited, and some scenarios blocked by people were not reconstructed and remain blank, while most of the scenarios blocked by moving people in Figure 18d were reconstructed.

Figure 18e,f depict the results of constructing semantic maps for high-dynamic scenarios. It can be seen that the features of the two moving persons were removed, and the parts of the scenario occluded by people were also reconstructed. The overall scenario reconstruction is relatively complete. The camera motion in Figure 18e is slower and has a smaller range, so the reconstruction scope is smaller, but the reconstruction effect is better.

Therefore, TSG-SLAM, with the help of its dynamic feature segmentation module, is capable of effectively handling the presence of dynamic objects in both static and dynamic scenarios, reconstructing scenarios accurately, acquiring the instance-level semantic information of objects, and building a static 3D semantic map with dynamic interference removed.

6.5. Real-World-Scenario Experiments

In order to verify the effectiveness of the TSG-SLAM system in real-world scenarios, experiments on the simultaneous localization and semantic mapping of mobile robots were conducted in an indoor laboratory. Due to the challenge of replicating identical trajectories for a mobile robot, an experiment was conducted using a mobile robot that was remotely controlled to move around in an indoor laboratory. A dataset was generated with the camera's viewpoint identical to that of the robot's movements which was used to evaluate the ORB-SLAM3 and TSG-SLAM systems. The evaluation was carried out on two different experimental scenarios, one static and the other dynamic, to assess the system's performance in simultaneous localization and semantic mapping. To simulate a dynamic environment, the experimenter moved freely around the scenario, capturing image sequences from various angles, as shown in Figure 19.

(a) static scenes

(b) dynamic scenes

Figure 19. Partial image sequence of real scenario.

In real-world scenarios, it is challenging to obtain an accurate camera motion trajectory. Therefore, the estimated trajectories of TSG-SLAM and ORB-SLAM3 were compared based on the dataset captured by the mobile robot in the experimental scenario. As the ground upon which the mobile robot moves is nearly horizontal, a 2D estimated trajectory plot in the x-y direction was created to facilitate the comparison of the estimated trajectories.

Figure 20 presents a comparison of the trajectory estimation between TSG-SLAM and ORB-SLAM3 in static scenarios. The use of Mecanum wheels with differential steering for the mobile robot can cause jitter in the camera when the steering angle is large, leading to more fluctuations in the estimated trajectory during sharp turns. Nevertheless, the estimated trajectories of TSG-SLAM and ORB-SLAM3 in real static scenarios are nearly identical, which is consistent with the comparison results of the estimated trajectories in public static datasets. As ORB-SLAM3 has good localization accuracy in static scenarios, this result indicates that TSG-SLAM also performs well in real static scenarios.

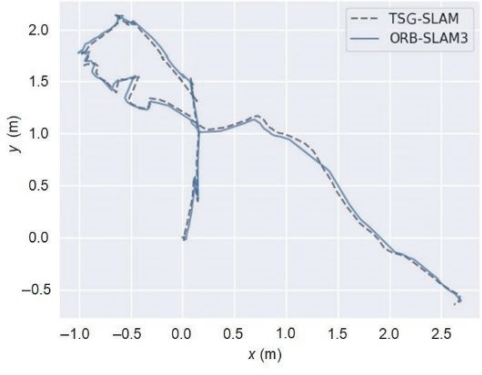


Figure 20. Comparison of estimated trajectory for real static scenarios.

Figure 21 illustrates a comparison of the trajectory estimation between TSG-SLAM and ORB-SLAM3 in dynamic scenarios. In the first half of the trajectory, where no dynamic targets are observed or are still far away from the camera, both methods produce almost identical trajectory estimations. However, in the middle section, where the camera approaches the dynamic target (marked with a red box in Figure 21), ORB-SLAM3 is significantly affected, resulting in substantial fluctuations in the estimated motion trajectory. On the other hand, TSG-SLAM, which processes the dynamic target, is less affected, leading to smaller fluctuations in the estimated trajectory.

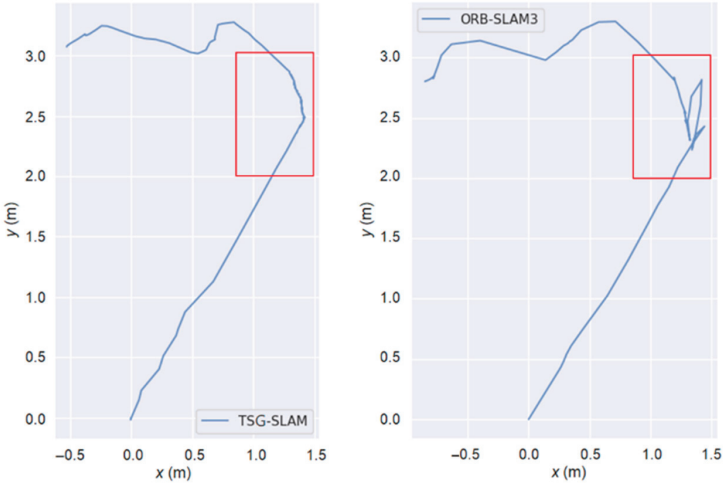
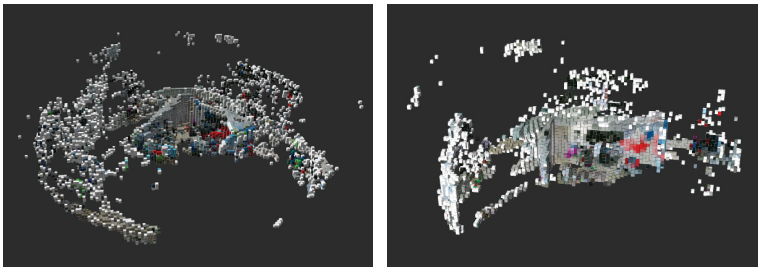


Figure 21. Comparison of estimated trajectory for real dynamic scenarios.

Figure 22 depicts a semantic octree map in a real scenario. Figure 22a,b display the map reconstruction results in static and dynamic scenarios, respectively. Due to the large scenario size and limited data collection, some details are still missing in the semantic map. Nevertheless, the overall effect is impressive, and the 3D geometric models of objects in the actual scenario were established well, with semantic tags and color information added, such as black displays and red tables, which produce a good visualization effect. Figure 22b indicates that TSG-SLAM only reconstructed static targets, while the moving experimenters were not reconstructed. This demonstrates that the dynamic feature module of TSG-SLAM successfully removed the features of dynamic targets, validating the effectiveness of the semantic mapping of TSG-SLAM in real-world dynamic environments.



（a）static scene　　（b）dynamic scene

Figure 22. Semantic map of real scenario.

7. Conclusions

This paper introduces TSG-SLAM, a simultaneous localization and semantic mapping method tailored for complex environments. The approach aims to address the impact of dynamic objects on mapping accuracy and the demand for semantic mapping in mobile robots. TSG-SLAM adds two threads to ORB-SLAM3's three-thread structure: dynamic feature removal and semantic map construction. The dynamic feature removal module tightly integrates the SOLOv2 instance segmentation algorithm with multi-view geometry techniques to detect and eliminate dynamic features, mitigating the influence of dynamic objects on visual SLAM systems. The semantic map construction module fuses target semantic information obtained by the instance segmentation algorithm with the 3D semantic point cloud, creating a 3D octree semantic map containing instance-level semantic information. Experimental results from the use of both public datasets and real-world scenarios demonstrate that TSG-SLAM can counteract the effects of moving objects on localization, exhibit excellent adaptability to dynamic environments, and ensure high localization accuracy and stability. The efficacy of the TSG-SLAM system's 3D semantic mapping is also validated, providing a theoretical foundation for mobile robots to execute high-level tasks, such as navigation and interaction in complex environments.

Future work is anticipated to focus on three key areas. Firstly, to mitigate the impact on system efficiency, the exploration of lightweight processing methods is proposed to improve segmentation speed and ensure high real-time performance. Secondly, to enhance localization and semantic mapping accuracy, the integration of depth information is suggested, addressing the limitations of the current 2D-based instance segmentation algorithm. Lastly, a more detailed analysis and testing will be conducted on the impact of the quantity and movement patterns of dynamic objects on the system.

Author Contributions: Conceptualization, Y.L. and P.C.; methodology, Y.L.; software, Y.L.; validation, Y.L. and Y.Z.; formal analysis, Y.L. and Y.Z.; investigation, Y.L.; resources, Y.L.; data curation, Y.L. and Y.Z.; writing—original draft preparation, Y.L. and Y.Z.; writing—review and editing, Y.Z. and P.C.; visualization, Y.L.; supervision, P.C.; project administration, P.C.; funding acquisition, P.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 62163014.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request. "COCO dataset" at https://cocodataset.org (accessed on 30 January 2023). "TUM dataset" at https://vision.in.tum.de/data/datasets/rgbd-dataset/download (accessed on 30 January 2023).

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

References

1. Davison, A.J.; Reid, I.D.; Molton, N.D.; Stasse, O. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1052–1067. [CrossRef] [PubMed]
2. Klein, G.; Murray, D. Parallel Tracking and Mapping for Small AR Workspaces. In Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, Nara, Japan, 13–16 November 2007; pp. 1–10.
3. Leutenegger, S.; Lynen, S.; Bosse, M.; Siegwart, R.; Furgale, P. Keyframe-Based Visual–Inertial Odometry Using Nonlinear Optimization. *Int. J. Robot. Res.* **2015**, *34*, 314–334. [CrossRef]
4. Mur-Artal, R.; Montiel, J.M.M.; Tardos, J.D. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *Trans. Rob.* **2015**, *31*, 1147–1163. [CrossRef]
5. Mur-Artal, R.; Tardós, J.D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [CrossRef]
6. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.; Tardós, J.D. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [CrossRef]

7. Mccormac, J.; Clark, R.; Bloesch, M.; Davison, A.; Leutenegger, S. Fusion++: Volumetric Object-Level SLAM. In Proceedings of the 2018 International Conference on 3D Vision (3DV), Verona, Italy, 5–8 September 2018; pp. 32–41.
8. Hoang, D.-C.; Lilienthal, A.J.; Stoyanov, T. Object-RPE: Dense 3D Reconstruction and Pose Estimation with Convolutional Neural Networks. *Robot. Auton. Syst.* **2020**, *133*, 103632. [CrossRef]
9. Hosseinzadeh, M.; Li, K.; Latif, Y.; Reid, I. Real-Time Monocular Object-Model Aware Sparse SLAM. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 7123–7129.
10. Oberlander, J.; Uhl, K.; Zollner, J.M.; Dillmann, R. A Region-Based SLAM Algorithm Capturing Metric, Topological, and Semantic Properties. In Proceedings of the 2008 IEEE International Conference on Robotics and Automation, Pasadena, CA, USA, 19–23 May 2008; pp. 1886–1891.
11. Kostavelis, I.; Gasteratos, A. Semantic Mapping for Mobile Robotics Tasks: A Survey. *Robot. Auton. Syst.* **2015**, *66*, 86–103. [CrossRef]
12. Luo, R.C.; Chiou, M. Hierarchical Semantic Mapping Using Convolutional Neural Networks for Intelligent Service Robotics. *IEEE Access* **2018**, *6*, 61287–61294. [CrossRef]
13. Lin, S.; Wang, J.; Xu, M.; Zhao, H.; Chen, Z. Topology Aware Object-Level Semantic Mapping Towards More Robust Loop Closure. *IEEE Robot. Autom. Lett.* **2021**, *6*, 7041–7048. [CrossRef]
14. Yang, B.; Jiang, T.; Wu, W.; Zhou, Y.; Dai, L. Automated Semantics and Topology Representation of Residential-Building Space Using Floor-Plan Raster Maps. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2022**, *15*, 7809–7825. [CrossRef]
15. Jin, J.; Jiang, X.; Yu, C.; Zhao, L.; Tang, Z. Dynamic Visual Simultaneous Localization and Mapping Based on Semantic Segmentation Module. *Appl. Intell.* **2023**, *53*, 19418–19432. [CrossRef]
16. Wang, K.; Yao, X.; Huang, Y.; Liu, M.; Lu, Y. Review of Visual SLAM in Dynamic Environment. *Robot* **2021**, *43*, 715–732. [CrossRef]
17. Yao, E.; Zhang, H.; Song, H.; Zhang, G. Fast and Robust Visual Odometry with a Low-Cost IMU in Dynamic Environments. *Ind. Robot.* **2019**, *46*, 882–894. [CrossRef]
18. Yang, D.; Bi, S.; Wang, W.; Yuan, C.; Wang, W.; Qi, X.; Cai, Y. DRE-SLAM: Dynamic RGB-D Encoder SLAM for a Differential-Drive Robot. *Remote Sens.* **2019**, *11*, 380. [CrossRef]
19. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
20. Sun, Y.; Liu, M.; Meng, M.Q.-H. Improving RGB-D SLAM in Dynamic Environments: A Motion Removal Approach. *Robot. Auton. Syst.* **2017**, *89*, 110–122. [CrossRef]
21. Sun, Y.; Liu, M.; Meng, M.Q.-H. Motion Removal for Reliable RGB-D SLAM in Dynamic Environments. *Robot. Auton. Syst.* **2018**, *108*, 115–128. [CrossRef]
22. Tan, W.; Liu, H.; Dong, Z.; Zhang, G.; Bao, H. Robust Monocular SLAM in Dynamic Environments. In Proceedings of the 2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Adelaide, SA, Australia, 1–4 October 2013; pp. 209–218.
23. Wang, Y.; Huang, S. Towards Dense Moving Object Segmentation Based Robust Dense RGB-D SLAM in Dynamic Scenarios. In Proceedings of the 2014 13th International Conference on Control Automation Robotics & Vision (ICARCV), Singapore, 10–12 December 2014; pp. 1841–1846.
24. Kerl, C.; Sturm, J.; Cremers, D. Dense Visual SLAM for RGB-D Cameras. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 2100–2106.
25. Yu, C.; Liu, Z.; Liu, X.-J.; Xie, F.; Yang, Y.; Wei, Q.; Fei, Q. DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1168–1174.
26. Bescos, B.; Fácil, J.M.; Civera, J.; Neira, J. DynaSLAM: Tracking, Mapping, and Inpainting in Dynamic Scenes. *IEEE Robot. Autom. Lett.* **2018**, *3*, 4076–4083. [CrossRef]
27. Bescos, B.; Campos, C.; Tardós, J.D.; Neira, J. DynaSLAM II: Tightly-Coupled Multi-Object Tracking and SLAM. *IEEE Robot. Autom. Lett.* **2021**, *6*, 5191–5198. [CrossRef]
28. You, Y.; Wei, P.; Cai, J.; Huang, W.; Kang, R.; Liu, H. MISD-SLAM: Multimodal Semantic SLAM for Dynamic Environments. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, e7600669. [CrossRef]
29. Liu, Y.; Miura, J. RDS-SLAM: Real-Time Dynamic SLAM Using Semantic Segmentation Methods. *IEEE Access* **2021**, *9*, 23772–23785. [CrossRef]
30. Zhao, Y.; Xiong, Z.; Zhou, S.; Peng, Z.; Campoy, P.; Zhang, L. KSF-SLAM: A Key Segmentation Frame Based Semantic SLAM in Dynamic Environments. *J. Intell. Robot. Syst.* **2022**, *105*, 3. [CrossRef]
31. Gonzalez, M.; Marchand, E.; Kacete, A.; Royan, J. TwistSLAM: Constrained SLAM in Dynamic Environment. *IEEE Robot. Autom. Lett.* **2022**, *7*, 6846–6853. [CrossRef]
32. Kuang, B.; Yuan, J.; Liu, Q. A Robust RGB-D SLAM Based on Multiple Geometric Features and Semantic Segmentation in Dynamic Environments. *Meas. Sci. Technol.* **2022**, *34*, 015402. [CrossRef]
33. Runz, M.; Buffier, M.; Agapito, L. MaskFusion: Real-Time Recognition, Tracking and Reconstruction of Multiple Moving Objects. In Proceedings of the 2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Munich, Germany, 16–20 October 2018; pp. 10–20.

34. Xu, B.; Li, W.; Tzoumanikas, D.; Bloesch, M.; Davison, A.; Leutenegger, S. MID-Fusion: Octree-Based Object-Level Multi-Instance Dynamic SLAM. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 5231–5237.
35. Li, A.; Wang, J.; Xu, M.; Chen, Z. DP-SLAM: A Visual SLAM with Moving Probability towards Dynamic Environments. *Inf. Sci.* **2021**, *556*, 128–142. [CrossRef]
36. Wu, W.; Guo, L.; Gao, H.; You, Z.; Liu, Y.; Chen, Z. YOLO-SLAM: A Semantic SLAM System towards Dynamic Environment with Geometric Constraint. *Neural Comput. Appl.* **2022**, *34*, 6011–6026. [CrossRef]
37. Li, J.; Zhang, R.; Liu, Y.; Zhang, Z.; Fan, R.; Liu, W. The Method of Static Semantic Map Construction Based on Instance Segmentation and Dynamic Point Elimination. *Electronics* **2021**, *10*, 1883. [CrossRef]
38. Xing, Z.; Zhu, X.; Dong, D. DE-SLAM: SLAM for Highly Dynamic Environment. *J. Field Robot.* **2022**, *39*, 528–542. [CrossRef]
39. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [CrossRef]
40. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2980–2988.
41. Wang, X.; Zhang, R.; Kong, T.; Li, L.; Shen, C. SOLOv2: Dynamic and Fast Instance Segmentation. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 6–12 December 2020.
42. Lin, T.-Y.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C.L.; Dollár, P. Microsoft COCO: Common Objects in Context. *arXiv* **2015**, arXiv:1405.0312v3.
43. Rusu, R.B.; Cousins, S. 3D Is Here: Point Cloud Library (PCL). In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1–4.
44. Sturm, J.; Engelhard, N.; Endres, F.; Burgard, W.; Cremers, D. A Benchmark for the Evaluation of RGB-D SLAM Systems. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 7–12 October 2012; pp. 573–580.
45. Hornung, A.; Wurm, K.M.; Bennewitz, M.; Stachniss, C.; Burgard, W. OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Auton. Robot.* **2013**, *34*, 189–206. [CrossRef]

sensors

MDPI

Article

Path Planning of a Mobile Robot for a Dynamic Indoor Environment Based on an SAC-LSTM Algorithm

Yongchao Zhang and Pengzhan Chen *

School of Intelligent Manufacturing, Taizhou University, Taizhou 318000, China; zycac@126.com
* Correspondence: cyxcpz@163.com

Abstract: This paper proposes an improved Soft Actor–Critic Long Short-Term Memory (SAC-LSTM) algorithm for fast path planning of mobile robots in dynamic environments. To achieve continuous motion and better decision making by incorporating historical and current states, a long short-term memory network (LSTM) with memory was integrated into the SAC algorithm. To mitigate the memory depreciation issue caused by resetting the LSTM's hidden states to zero during training, a burn-in training method was adopted to boost the performance. Moreover, a prioritized experience replay mechanism was implemented to enhance sampling efficiency and speed up convergence. Based on the SAC-LSTM framework, a motion model for the Turtlebot3 mobile robot was established by designing the state space, action space, reward function, and overall planning process. Three simulation experiments were conducted in obstacle-free, static obstacle, and dynamic obstacle environments using the ROS platform and Gazebo9 software. The results were compared with the SAC algorithm. In all scenarios, the SAC-LSTM algorithm demonstrated a faster convergence rate and a higher path planning success rate, registering a significant 10.5 percentage point improvement in the success rate of reaching the target point in the dynamic obstacle environment. Additionally, the time taken for path planning was shorter, and the planned paths were more concise.

Keywords: mobile robot; path planning; SAC-LSTM algorithm; burn-in mechanism; prioritized experience replay mechanism

Citation: Zhang, Y.; Chen, P. Path Planning of a Mobile Robot for a Dynamic Indoor Environment Based on an SAC-LSTM Algorithm. *Sensors* **2023**, *23*, 9802. https://doi.org/10.3390/s23249802

Academic Editor: Sergio Toral Marín

Received: 12 October 2023
Revised: 1 December 2023
Accepted: 9 December 2023
Published: 13 December 2023

1. Introduction

Mobile robot path planning is a crucial technique that enables robots to navigate through an environment while avoiding obstacles. This is achieved by planning a collision-free path [1] from the robot's current position to a desired destination based on environmental information obtained through sensors. Traditional path planning methods encompass a variety of algorithms with different principles and application scenarios. These methods can be classified into global and local path planning algorithms, including graph search methods, random sampling methods, bionic algorithms, artificial potential field methods, simulated annealing algorithms, neural network methods, and dynamic window methods.

Classical graph search algorithms include Dijkstra [2], A* [3], and D* [4]. The Dijkstra algorithm solves the shortest path problem from a single point to all other vertices in a directed graph using a greedy algorithm. Although it can obtain the optimal path, it has a high computational cost and low efficiency. The A* algorithm, based on the Dijkstra algorithm, improves efficiency by adding the heuristic information, but its effectiveness in complex environments is not guaranteed. Moreover, this algorithm is only suitable for static environments and it performs poorly in dynamic environments. The D* algorithm is an improved version of the A* algorithm that can be applied in dynamically changing scenarios.

Classical random sampling methods such as Probabilistic Roadmap (PRM) [5] and Rapidly Exploring Random Tree (RRT) [6] have been widely used in robot path planning and motion control fields. The Lazy PRM [7] algorithm is an improved version of the PRM

algorithm that enhances efficiency by reducing the number of calls to the local planner. Liu et al. [8] improved the RRT algorithm by using a goal-biased sampling strategy to determine the nodes and introduced an event-triggered step length extension based on the hyperbolic tangent function to improve node generation efficiency. Euclidean distance and angle constraints were used in the cost function of node connection optimization. Finally, the path was optimized further using path pruning and Bezier curve smoothing methods, leading to an improved convergence and accuracy.

Bionic algorithms are heuristic optimization algorithms based on the evolution and behavior of biological organisms in nature, mainly including genetic algorithms [9] and ant colony algorithms [10]. Liang et al. [11] integrated the ant colony algorithm and genetic algorithm to propose a hybrid path planning algorithm, which used a genetic algorithm to generate initial paths and then used an ant colony algorithm to optimize them, significantly improving the accuracy and efficiency of path planning.

The artificial potential field method [12] was first applied in the mobile robot path planning field in 1986. Zha et al. [13] improved the artificial potential field method by adding a distance factor between the target point and the vehicle in the repulsive force function, and a safety distance within the influence range of obstacles. The experimental results showed that when the vehicle was driving within the safety distance of obstacles, it would be subject to increased repulsive force, ensuring the safety of the vehicle during driving. Zhao et al. [14] proposed a multi-robot path planning method based on an improved artificial potential field and a fuzzy inference system, which overcame the problem of smooth path planning existing in traditional artificial potential field methods by using the incremental potential field calculation method.

Afifi et al. [15] proposed a vehicle path planning method based on a simulated annealing algorithm, which solved the vehicle path planning problem under time window constraints, using the simulated annealing algorithm for path optimization and search. Jun et al. [16] proposed a particle swarm optimization combined with simulated annealing (PSO-ICSA) to self-adaptively adjust the coefficients, enabling high-dimension objects to enhance the global convergence ability.

Zhang et al. [17] proposed an indoor mobile robot path planning method based on deep learning. They used deep learning models to extract features from sensor data to predict the robot's motion direction and speed, achieving good results in experiments. During robot path planning, when the environment changes, the robot re-executes the algorithm, significantly increasing the time to find the optimal path. Dimensionality reduction seems to be a solution to this problem. Ferreira et al. [18] proposed a path planning algorithm based on a deep learning encoder model. They built a CNN encoder that uses nonlinear correlations to reduce data dimensions, eliminating unnecessary information and accelerating the efficiency of finding the shortest path.

Bai et al. [19] combined a dynamic window algorithm with the A* algorithm to propose an unmanned aerial vehicle path planning method. Experiments proved that the improved algorithm significantly reduced both path planning length and time. Lee et al. [20] proposed a dynamic window approach based on finite distribution estimation. Experimental results showed that this method could achieve a reliable obstacle avoidance effect for mobile robots and exhibited good performance in multiple scenarios.

Most of the above methods heavily rely on environmental map information. When faced with unknown environments, these methods may not achieve ideal results. When mobile robots are in unknown environments, due to the lack of environmental cognition, they must have certain exploration and autonomous learning abilities to efficiently complete path planning tasks. Therefore, studying mobile robot path planning algorithms that rely on little or no map information of the environment and have autonomous learning abilities has become one of the current key research topics.

Recently, artificial intelligence technologies represented by deep learning and reinforcement learning have developed rapidly. Reinforcement learning algorithms do not rely on map information and can learn path planning strategies in unknown environments by

interacting with the environment through trial and error. However, reinforcement learning is prone to the problem of dimensionality disaster in the path planning process. Deep learning is an end-to-end model that can fit the mapping relationship between high-dimensional input and output data, and is suitable for dealing with high-dimensional data problems. Deep reinforcement learning combines the advantages of deep learning and reinforcement learning and has a huge advantage over other path planning methods when dealing with complex unknown environments. Nevertheless, path planning methods based on deep reinforcement learning still face problems such as sparse rewards, a slow learning rate, and difficult convergence in application process.

This paper focuses on the indoor mobile robot path planning problem in complex and unknown dynamic environments with several static and dynamic obstacles, using the Soft Actor–Critic (SAC) algorithm as the main method. The SAC algorithm, while powerful, exhibits certain limitations, namely (1) difficulty in processing complex or dynamic environmental information; (2) inadequacy in handling long-term dependencies in path planning; and (3) lack of predictive capability for future environmental states. To address these shortcomings, an improved SAC-LSTM algorithm is proposed. The main contributions of this paper are as follows. First, the LSTM network with memory capability is introduced into the SAC algorithm, allowing the agent to make more reasonable decisions by combining historical and current states and predict the dynamic changes in the environment, such as the future positions of moving obstacles. Second, the burn-in training mechanism is introduced to solve the problem of memory impairment caused by the hidden state being zeroed during the training process of the LSTM network, stabilizing the learning process, especially in the early stages of training. Third, by combining the prioritized experience replay mechanism, the problem of low sampling efficiency of the algorithm is solved, and the convergence speed of the algorithm is accelerated. Fourth, a complex dynamic test scenario is constructed for indoor mobile robots, featuring multiple stationary and moving obstacles of various sizes and shapes, as well as different motion trajectories, making the test scenario more realistic.

The rest of this paper is organized as follows: Section 2 provides an overview of related works regarding path planning methods in dynamic environments. Section 3 describes the proposed SAC-LSTM system framework and algorithms in detail. Section 4 presents the experimental results and performance analysis. Finally, Section 5 concludes the paper and discusses future research directions.

2. Related Work

In recent years, researchers have started to apply deep reinforcement learning (DRL) algorithms to the field of path planning to solve complex problems. In 2016, Tai et al. [21] first applied the DQN algorithm to indoor mobile robots, which could complete path planning tasks in indoor scenarios, but the algorithm had low generalization. Wang et al. [22] introduced an improved DQN algorithm combined with artificial potential field methods to design reward functions, improving the efficiency of mobile robot path planning. However, it could not achieve continuous action output for robots.

Lei et al. [23] proposed a path planning algorithm using a DDQN framework with environment information obtained through LiDAR. It designed a new reward function to address the instability issue during training, improving algorithm stability, but its application was limited to simple scenarios with no guarantee of efficiency in complex situations. Tai et al. [24] used an asynchronous deterministic policy gradient algorithm to build a mapless path planner with input from the mobile robot's LiDAR-scanned environment information. After a period of training, the mobile robot could successfully reach the designated target, but the planned path was relatively tortuous.

In [25], a deep reinforcement learning-based online path planning algorithm was proposed, successfully achieving path planning for drones in dynamic environments. Zhang et al. [26] combined the advantages of DRL and interactive RL algorithms and proposed a deep interactive reinforcement method for autonomous underwater vehicle (AUV)

path tracking, achieving path tracking in a Gazebo-simulated environment. In [27], the DDPG algorithm was extended to a parallel deep deterministic policy gradient algorithm (PDDPG) and applied to multi-robot mapless collaborative navigation tasks.

Wang et al. [28] proposed an end-to-end modular DRL architecture that decomposed navigation tasks in complex dynamic environments into local obstacle avoidance and global navigation subtasks, using DQN and dual-stream DQN algorithms to solve them, respectively. Experiments demonstrated that this modular architecture could efficiently complete navigation tasks. Gao et al. [29] introduced an incremental training mode to address the low training efficiency in DRL path planning. In [30], incorporating a curiosity mechanism into the A3C algorithm provided an additional reward for the exploration behavior of mobile robots, addressing the reward sparsity issue to some extent.

De Jesus et al. [31] proposed a mobile robot path planning algorithm based on SAC, achieving path planning in different scenarios built on ROS. However, the algorithm still faced the problem of sparse environmental rewards. Park, K.-W. et al. [32] employed the SAC algorithm to solve the path planning problem for multi-arm manipulators to avoid fixed and moving obstacles, and used the LSTM network to predict the position of the moving obstacles. The simulation and experimental results showed the optimal path and good prediction of obstacle position. Although the simulation and experimental scenario considered moving obstacles, there was only one moving obstacle, lacking verification for multiple obstacles. Additionally, the multi-arm manipulators used a camera in conjunction with the OpenCV vision algorithm to detect obstacles, which has limited effectiveness in detecting and predicting multiple moving obstacles.

In addition to LSTM, the metaheuristic-based recurrent neural network (RNN) [33] has also been applied to control mobile robotic systems. The metaheuristic-based RNNs often integrate optimization algorithms (such as genetic algorithms and particle swarm optimization) with neural network characteristics. The Beetle Antennae Olfactory Recurrent Neural Network (BAORNN) [34] is a metaheuristic-based control framework used for simultaneous tracking control and obstacle avoidance in redundant manipulators. A key feature of this framework is that it unifies tracking control and obstacle avoidance into a single constrained optimization problem, actively rewarding the optimizer to avoid obstacles by introducing a penalty term in the objective function. The distance calculation is based on the Gilbert–Johnson–Keerthi algorithm, which calculates the distance between the manipulator and obstacles by directly using their three-dimensional geometric shapes. In contrast, the RNN may offer more flexibility in solving specific control and optimization problems, but might not match the LSTM's proficiency in handling complex sequential data and long-term dependencies.

As seen from the above literature, DRL-based path planning methods have the advantages of not relying on map information and autonomous learning capabilities, making them highly suitable for path planning tasks in unknown environments. However, during application, there are still issues such as sparse rewards, slow learning rates, and difficulty in converging the algorithm. In particular, in dynamic or complex environments, the SAC algorithm may require more accurate environmental models for effective learning. This could be difficult to achieve in practice, particularly in environments with high uncertainty or rapid changes. The SAC algorithm might be insufficient in effectively dealing with highly dynamic or non-stationary environments, where environmental states and dynamics can change rapidly.

3. Path Planning System

3.1. Framework of Path Planning

The task of the mobile robot path planning algorithm is to plan an optimal path from the starting point to the destination, while minimizing the robot's movement cost, avoiding obstacles, and enabling the robot to reach the destination as quickly as possible. The deep reinforcement learning algorithm is designed to learn a strategy that enables the agent to maximize its reward through interactions with the environment. Applying this

algorithm to the mobile robot path planning task essentially transforms the path planning problem into a reinforcement learning problem. The strategy that allows the robot to quickly reach the target point is learned by having the mobile robot try different actions and interact with the environment. The process of interacting with the environment is a Markov Decision Process (MDP), which requires defining the state space, action space, and reward function. This article proposes the application of the SAC-LSTM algorithm to the mobile robot path planning task, as depicted in Figure 1. In this framework, the SAC-LSTM algorithm combines historical and current states to select an action and controls the mobile robot to execute the action in the environment. After interacting with the environment, the mobile robot receives a reward value and a state value. During the training process, important experience samples are prioritized using the prioritized experience replay, and the algorithm is trained using the burn-in mechanism to obtain an optimal strategy for quickly reaching the target point.

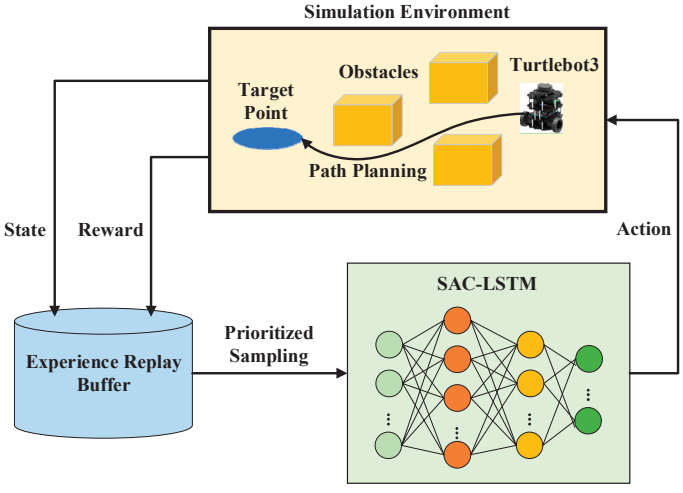


Figure 1. Path planning framework based on SAC-LSTM.

3.2. SAC-LSTM Algorithm

The Soft Actor–Critic (SAC) algorithm is an off-policy Actor–Critic algorithm based on the maximum entropy model framework. Off-policy learning improves sample efficiency and reduces training time by utilizing data generated from a behavioral policy to train the target policy. The Actor–Critic framework allows the algorithm to be applied to continuous state and action spaces, thus expanding the range of actions and states that can be selected. To prevent premature convergence and avoid local optima, the SAC algorithm uses a stochastic policy. Additionally, introducing entropy during the learning process improves the algorithm's ability to explore the environment and enhances its performance and robustness.

3.2.1. Maximum Entropy Principle

Entropy is a measure of uncertainty or randomness in a system, used in information theory. It is inversely proportional to system certainty and directly proportional to system randomness. Let x be a random variable with probability density function $P(X)$. The entropy $H(P)$ of X can be calculated according to

$$H(P) = \mathop{\mathbb{E}}_{X \sim P}[-\log P(X)]. \tag{1}$$

Classical reinforcement learning algorithms adopt a learning strategy that seeks to maximize the expected value of cumulative rewards. This objective is expressed in the following expression

$$\pi^* = \underset{\pi}{\operatorname{argmax}} Q(s,\ a) = \underset{\pi}{\operatorname{argmax}} \mathbb{E}\left[\sum_{t=0}^{\infty} R(s_t,\ a_t)\right], \tag{2}$$

where the action-value function $Q(s, a)$ represents the expected return obtained by taking a certain action a in a state s based on the policy function π. The reward value $R(s, a)$ is the expected value of the sum of immediate rewards for all possible actions at any given time t, denoted as $R(s, a) = E[R_{t+1} \mid S_t = s, A_t = a]$, in which s and a represent the state and action at time t, respectively.

The objective of the SAC algorithm in reinforcement learning is to maximize the entropy-regularized reward, which is the sum of cumulative reward and policy entropy. In other words, the SAC algorithm incorporates policy entropy in addition to maximizing the cumulative reward in classical reinforcement learning algorithms. The process of finding the optimal policy in the SAC algorithm is represented by Equation (3).

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \underset{\tau\sim\pi}{\mathbb{E}}\left[\sum_{t=0}^{\infty} \gamma^t \left(\underbrace{R(s_t, a_t, s_{t+1})}_{\text{reward}} + \underbrace{\alpha H(\pi(\cdot|s_t))}_{\text{entropy}} \right)\right] \tag{3}$$

In Equation (3), α is the entropy regularization coefficient, which adjusts the relative importance of reward and entropy. A higher value of α indicates a larger proportion of entropy, prompting the intelligent agent to explore the environment and employ diverse actions to achieve its objectives. Conversely, a lower value of α implies a reduced emphasis on entropy, causing the intelligent agent to rely more on existing actions to accomplish its objectives. A trajectory τ is a sequence of states and actions, where $\tau = (s_0, a_0, s_1, a_1, \ldots)$. $\gamma \in (0, 1)$ is the discount factor. Similarly, compared to classical reinforcement learning algorithms, the SAC algorithm adds entropy rewards to both the value function V^π and the action-value function Q^π. The specific definitions are shown in Equations (4) and (5),

$$V^\pi(s) = \underset{\tau\sim\pi}{\mathbb{E}}\left[\sum_{t=0}^{\infty} \gamma^t (R(s_t, a_t, s_{t+1}) + \alpha H(\pi(\cdot|s_t)))\middle| s_0 = s\right] \tag{4}$$

$$Q^\pi(s,a) = \underset{\tau\sim\pi}{\mathbb{E}}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) + \alpha\sum_{t=1}^{\infty} \gamma^t H(\pi(\cdot|s_t))\middle| s_0 = s, a_0 = a\right] \tag{5}$$

The relationship between $V^\pi(s)$ and $Q^\pi(s)$ is as follows:

$$V^\pi(s) = \underset{a\sim\pi}{\mathbb{E}}[Q^\pi(s,a)] + \alpha H(\pi(\cdot|s)) \tag{6}$$

and the Bellman equation for $Q^\pi(s)$ is

$$\begin{aligned} Q^\pi(s,a) &= \underset{\substack{s'\sim P\\ a'\sim\pi}}{\mathbb{E}} [R(s,a,s') + \gamma(Q^\pi(s',a') + \alpha H(\pi(\cdot|s')))] \\ &= \underset{s'\sim P}{\mathbb{E}}[R(s,a,s') + \gamma V^\pi(s')] \end{aligned} . \tag{7}$$

Based on the above steps, the policy network can iteratively update its network parameters using the iterative Bellman equation in order to obtain the optimal policy.

3.2.2. Updating Process of SAC Algorithm

This paper utilizes an entropy-weighted SAC algorithm with automatic adjustment of α. The algorithm consists of two Q networks (Critic networks) and one Actor network

(Policy network). The subsequent sections will present the updating processes of the Q networks and the Actor network.

1. Updating Process of the Q Networks

The Q networks are updated by sampling experiences $(s_t, a_t, r_t, s_{t+1}, d)$ from the experience replay buffer. The estimation of the state-action value for the Q networks is calculated by

$$Q^{\pi}(s_t, a_t) \approx r_t + \gamma(Q^{\pi}(s_{t+1}, \widetilde{a}_{t+1}) - \alpha \log \pi(\widetilde{a}_{t+1}|s_{t+1})),\ \widetilde{a}_{t+1} \sim \pi(\cdot|s_{t+1}) \tag{8}$$

where $\widetilde{a}_{t+1}$ is the prediction of the action a_{t+1} by the Actor network. The SAC algorithm employs the mean squared loss function as its loss. Define $D = \{\tau_i\}_{i=1,\ldots,N}$ as a set of trajectories, where each trajectory is obtained by letting the agent act in the environment using the policy π_θ. The loss function for the Q networks is defined as follows:

$$L(\phi_j, D) = \frac{1}{|D|} \sum_{(s_t, a_t, r_{t+1}, s_{t+1}, d) \in D} \left[\left(Q_{\phi_j}(s_t, a_t) - Q_{\text{target}}\ (r_t, s_{t+1}, d) \right)^2 \right],\ j = 1, 2 \tag{9}$$

where $|D| = N$ is the number of trajectories in D.

The SAC algorithm incorporates the clipped double-Q technique during the training of the Q networks. Consequently, $Q_{target}(r_t, s_{t+1}, d)$ takes the minimum Q-value between the two Q approximators. The specific definition is as follows:

$$Q_{target}(r_t, s_{t+1}, d) = r_t + \gamma(1-d)\left(\min_{j=1,2} Q_{\phi_{targ,j}}(s_{t+1}, \widetilde{a}_{t+1}) - \alpha \log \pi_\theta(\widetilde{a}_{t+1}|s_{t+1}) \right),\ \widetilde{a}_{t+1} \sim \pi_\theta(\cdot|s_{t+1}). \tag{10}$$

2. Updating Process of the Actor Network

The SAC algorithm utilizes a squashed Gaussian policy to select actions, which means that action samples are obtained according to

$$\widetilde{a}_\theta(s, \xi) = \tanh(\mu_\theta(s) + \sigma_\theta(s) \odot \xi),\ \xi \sim \mathcal{N}(0, I), \tag{11}$$

where θ represents the parameters of the Actor network π_θ, $\mu_\theta(s)$ and $\sigma_\theta(s)$ respectively correspond to the mean and standard deviation of the action distribution outputted by the Actor network π_θ, and ξ denotes a random noise that follows a normal distribution. The reparameterization technique is used to optimize the policy in order to rewrite the expectation over actions into an expectation over noise, as follows:

$$\mathop{\mathbb{E}}_{a \sim \pi_\theta}[Q^{\pi_\theta}(s, a) - \alpha \log \pi_\theta(a|s)] = \mathop{\mathbb{E}}_{\xi \sim \mathcal{N}}[Q^{\pi_\theta}(s, \widetilde{a}_\theta(s, \xi)) - \alpha \log \pi_\theta(\widetilde{a}_\theta(s, \xi)|s)]. \tag{12}$$

To obtain the policy loss, the final step is to substitute $Q^{\pi_\theta}(s, \widetilde{a}_\theta(s, \xi))$ with one of two function approximators. The policy of the Actor network π_θ is thus optimized according to

$$L(\pi_\theta, \mathcal{D}) = \max_\theta \mathop{\mathbb{E}}_{\substack{s \sim \mathcal{D} \\ \xi \sim \mathcal{N}}} \left[\min_{j=1,2} Q_{\phi_j}(s, \widetilde{a}_\theta(s, \xi)) - \alpha \log \pi_\theta(\widetilde{a}_\theta(s, \xi)|s) \right], \tag{13}$$

where $\min_{j=1,2} Q_{\phi_j}(s, \widetilde{a}_\theta(s, \xi))$ is the minimum of the two Q approximators.

3.2.3. LSTM Network

During the process of mobile robot path planning, individual states often fail to provide sufficient information to guide the robot in making optimal decisions. As a result, this paper proposes enhancements to the neural network of the SAC algorithm by incorporating LSTM (Long Short-Term Memory) neural networks [35]. By introducing LSTM networks, the algorithm gains the ability to retain and utilize past states alongside the current state, enabling it to make more informed and rational decisions.

Conventional fully connected neural networks are limited in their ability to effectively address time-dependent problems, as their outputs are solely determined by the input at the current time step. However, the Recurrent Neural Network (RNN) provides a solution to this challenge. By incorporating recurrent networks within their architecture, RNNs have demonstrated remarkable efficacy in handling time-dependent problems. Nevertheless, during the training process, RNNs are susceptible to issues such as gradient vanishing or exploding. LSTM networks introduce the gated mechanisms on the foundation of RNNs, allowing for selective information retention and addressing the limitations of traditional RNNs. The LSTM structure comprises three essential gates: the forget gate f, the input gate i, and the output gate o. The forget gate $\boldsymbol{f}$ determines how much information from the previous time step's memory cell C_{t-1} should be retained for the current time step's memory cell C_t. The input gate $\boldsymbol{i}$ regulates the amount of current time step's information to be stored in the candidate state $\widetilde{C}_t$. The output gate $\boldsymbol{o}$ controls the extent to which information from the current time step's memory cell C_t should be conveyed to the current hidden state Y_t. The diagram depicting the structure of the LSTM neural networks is illustrated in Figure 2.

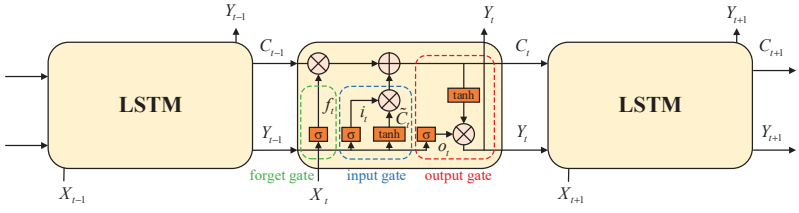


Figure 2. Structure diagram of LSTM neural networks.

In Figure 2, t refers to the current time. X_{t-1}, X_t, and X_{t+1} correspond to the inputs of the previous, current, and next time steps, respectively. Similarly, Y_{t-1}, Y_t, and Y_{t+1} denote the outputs of the previous, current, and next time steps, respectively. In addition, C_{t-1}, C_t, and C_{t+1} represent the memory cells at the previous, current, and next time steps, respectively. σ is the sigmoid function, defining output values between 0 and 1. The output values of the function tanh range from -1 to 1.

The workflow of LSTM is as follows. To begin with, the forget gate selectively discards information from the memory cell C_{t-1} of the previous time step. In order to obtain the forget coefficient, the previous hidden state information Y_{t-1} and the present input information X_t are both passed through the sigmoid function. The forget gate f_t is obtained by

$$f_t = \sigma\left(W_f \cdot [Y_{t-1}, X_t] + b_f\right) \tag{14}$$

where W_f and b_f represents the weight and bias of the layer network, respectively.

The second step involves generating the information required to update the current memory unit C_t. This process is divided into two parts. Firstly, the update value i_t is generated through the sigmoid layer of the input gate. Secondly, a new candidate value $\widetilde{C}_t$ is generated using the tanh layer. The specific calculations are defined by Equations (15) and (16).

$$i_t = \sigma(W_i \cdot [Y_{t-1}, X_t] + b_i) \tag{15}$$

$$\widetilde{C}_t = \tanh(W_C \cdot [Y_{t-1}, X_t] + b_C) \tag{16}$$

The current memory unit C_t at the current time step is defined by Equation (17). It is computed as the sum of two products: the product of the memory unit at the previous

time step C_{t-1} and the forget gate control signal f_t, and the product of the input gate value i_t and the candidate value $\widetilde{C}_t$.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \widetilde{C}_t \tag{17}$$

The final step involves determining the output of the LSTM model. Firstly, the control signal o_t of the output gate is obtained using the sigmoid function. Secondly, the current memory unit C_t at the current time step is scaled by applying the tanh function. Multiplying these two values yields the current output value Y_t. The specific calculations are defined by Equations (18) and (19).

$$o_t = \sigma(W_o \cdot [Y_{t-1}, X_t] + b_o) \tag{18}$$

$$Y_t = o_t \cdot \tanh(C_t) \tag{19}$$

Figure 3 illustrates the network structure of the SAC algorithm after incorporating LSTM. This diagram represents the final network architecture of the SAC-LSTM algorithm, as subsequent enhancements do not pertain to the network structure.

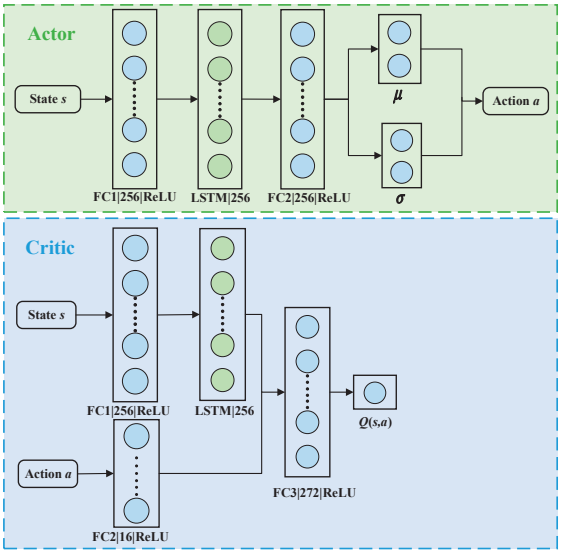


Figure 3. Network structure of SAC-LSTM.

The network architecture of the SAC-LSTM algorithm consists of two components: the Actor network and the Critic network. The Actor network begins with a fully connected layer, FC1, comprising 256 neurons. It takes the state feature vector $\boldsymbol{s}$ as input and utilizes the ReLU activation function to enhance feature extraction. The subsequent layer is a memory-capable LSTM layer with 256 neurons, enabling the algorithm to incorporate historical and current states for improved decision making. Following the LSTM layer, there is a fully connected layer, FC2, with 256 neurons that applies the ReLU activation function to process the LSTM layer's outputs. The final layer is another fully connected layer with four neurons, producing the mean μ and standard deviation σ, which are used to resample from a Gaussian distribution $N = (\mu, \sigma)$. The resulting action $\boldsymbol{a}$ is obtained by applying the tanh activation function.

The Critic network receives both the state vector $\boldsymbol{s}$ and the action vector $\boldsymbol{a}$ as inputs. The FC1 and LSTM layers mirror those of the Actor network. The FC2 layer, a fully connected layer with 16 neurons, extracts the features from $\boldsymbol{a}$, using the ReLU activation function. The FC3 layer takes the concatenated features from the LSTM and FC2 layers

as input, comprising 272 neurons and utilizing the ReLU activation function. Finally, the output layer consists of a single neuron, which outputs the *Q*-value for updating the Actor network.

3.2.4. Burn-In Mechanism

Updating LSTM networks requires a series of consecutive sequential samples. However, the high correlation among consecutive sequence samples can lead to increased variance in parameter updates. Therefore, the currently predominant approach is to employ the random order update method, as utilized in DRQN. This approach involves selecting a complete episode of experiences from the experience replay buffer and randomly choosing a fixed-length continuous sequence from that episode to train the algorithm. Prior to each training iteration, the LSTM's hidden state h_{int} is initialized to zero. Figure 4 provides a schematic representation of the random order update method, wherein the orange circles represent individual experience tuples $(s_t, a_t, r_{t+1}, s_{t+1}, d)$, the black boxes represent sequence experiences of length L, and DRL refers to a deep reinforcement learning algorithm incorporating an LSTM network. This updating method offers the advantage of simplicity and low complexity. However, resetting the LSTM network's hidden state to zero before training can lead to impaired memory within the LSTM network, subsequently impacting the algorithm's performance.

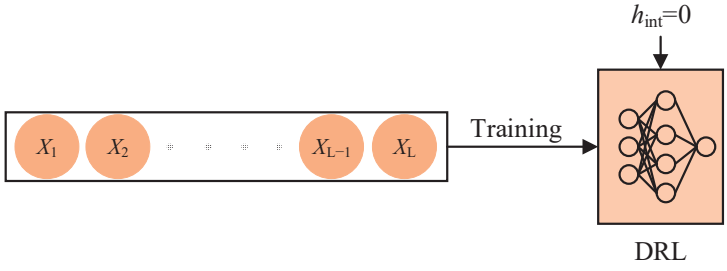


Figure 4. Random order update method.

Therefore, the burn-in mechanism utilized in the R2D2 algorithm [36] is introduced in this paper. The burn-in mechanism serves as a warm-up mechanism, initializing the LSTM network's hidden state h_{int} with a portion of historical data prior to training. Figure 5 depicts the conceptual diagram illustrating the application of the burn-in period in deep reinforcement learning algorithms. The black box represents a sequence of data, while the green circles represent the burn-in data comprising l_b items. The red circles represent the training data consisting of l_t items. When the sequence data are sampled by the DRL algorithm, the first l_b items are used to update the LSTM's hidden state h_t within the DRL framework. Subsequently, the remaining l_t items are utilized to train the DRL algorithm. By incorporating the burn-in mechanism, the hidden state of the LSTM network is updated before training, thereby circumventing the issue of impaired memory capacity caused by zeroing the LSTM network's hidden state prior to training. Consequently, this integration enhances the performance of the algorithm.

This study treats a fixed-length sequence $(s_t, a_t, r_{t+1}, s_{t+1}, d)$,..., $(s_{t+L}, a_{t+L}, r_{t+1+L}, s_{t+1+L}, d)$ of L as a single experience. As the experience tuples during the burn-in period are not utilized for network training, an approach is adopted to prevent experience wastage. Specifically, the storage format of experiences is designed in such a way that there is a duplication of half the experience tuple between two adjacent experiences. Figure 6 provides a visual representation of the specific storage format, with each circle representing an individual experience tuple $(s_t, a_t, r_{t+1}, s_{t+1}, d)$.

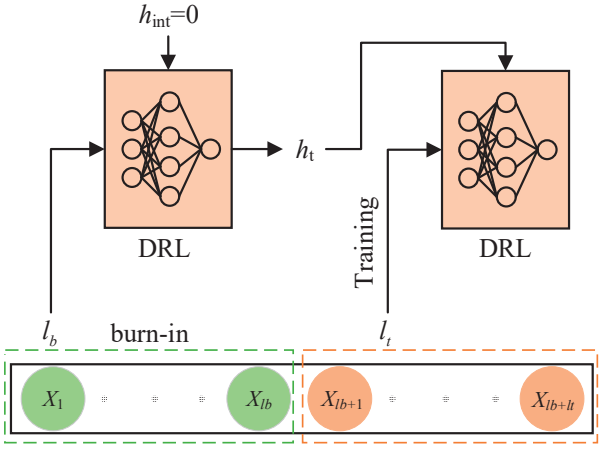


Figure 5. Burn-in period.

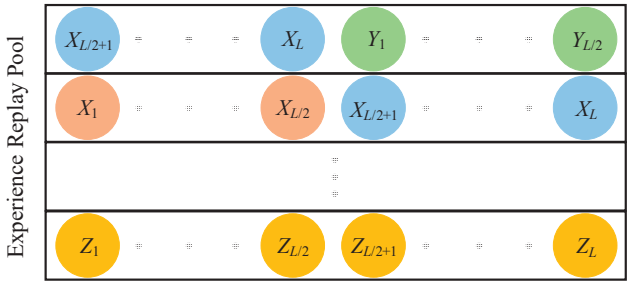


Figure 6. Experience storage format.

3.2.5. Prioritized Experience Replay

During the exploration of the environment, the agent accumulates training data, which is then stored in an experience replay buffer in the form of experience tuples. However, these samples tend to be sparse, and there exists a strong correlation among consecutively collected samples. Hence, in the context of deep reinforcement learning training, it is essential to employ random sampling of the samples within the experience replay buffer. This approach serves to enhance the efficiency of data utilization, disrupt the inter-sample correlation, and effectively mitigate the occurrence of overfitting in neural networks.

The SAC algorithm employs random sampling during training, which enhances the efficiency of data utilization and mitigates neural network overfitting to a certain extent. However, the assumption of equal importance for all experience samples in random sampling is not aligned with reality. In practice, different experience samples possess varying levels of significance. For instance, experiences with high success rates or frequent failures hold greater value for the algorithm's learning process, as they can expedite convergence. Consequently, this paper integrates the concept of prioritized experience replay with the SAC algorithm. This integration allows the agent to discern the importance of experience samples and prioritize frequent sampling of high-value samples, thus accelerating the convergence speed of the algorithm.

In the context of reinforcement learning, TD-error is commonly utilized to quantify the importance of samples, with a higher TD-error value indicating a greater degree of significance for the respective sample. By prioritizing the learning from samples with larger TD-errors, the algorithm can expedite the rate of learning. Specifically, in the prioritized experience replay DQN algorithm, the TD-error of each experience tuple $(s_t, a_t, r_t, s_{t+1}, d)$

is defined as the error δ_t between the current Q-value and the target Q-value, as illustrated in Equation (20), in which Q_{target} and Q respectively represent the target Q-network and the current Q-network.

$$\delta_t = r(s_t, a_t) + \gamma Q_{target}(s_{t+1}, \widetilde{a}_{t+1}) - Q(s_t, a_t) \tag{20}$$

Unlike the DQN algorithm, the SAC algorithm incorporates two Q-networks. This paper defines the absolute value $|\delta_t|$ of the TD-error for an experience tuple as the average of the absolute TD-errors from the two Q-networks, as precisely specified in Equation (21).

$$\left|\delta_t\right| = \frac{1}{2}\sum_{j=1}^{2}\left|Q_{\phi_j}(s_t, a_t) - Q_{\text{target}}\left(r_t, s_{t+1}, d\right)\right| \tag{21}$$

As the LSTM network is introduced, the storage format of experiences has transformed into the form depicted in Figure 6. Moreover, in conjunction with the burn-in mechanism, the TD-error of an experience sample $\{(s_t, a_t, r_{t+1}, s_{t+1}, d), \ldots, (s_{t+L}, a_{t+L}, r_{t+1+L}, s_{t+1+L}, d)\}$ is defined as the absolute TD-error of the subsequent l_t-term experience tuples. This specific definition is presented in Equation (22).

$$\delta = \frac{1}{l_t}\sum_{t=l_b+1}^{l_b+l_t}|\delta(s_t, a_t, r_t, s_{t+1}, d)|, \tag{22}$$

The sampling probability [37] for an experience sample is expressed by

$$P(i) = \frac{p_i^{\alpha_p}}{\sum\limits_k p_k^{\alpha_p}}, \alpha_p \in [0, 1]. \tag{23}$$

In Equation (23), the exponent α_p serves as the coefficient for regulating prioritization. When α_p= 0, the sampling method reverts to uniform sampling. $p_i > 0$ is the priority of transition i based on TD-error, employing a proportional prioritization approach defined in Equation (24).

$$p_i = |\delta_i| + \varepsilon \tag{24}$$

In Equation (24), ε is typically a small positive value that ensures the inclusion of experience samples with a TD-error of 0. However, prioritizing samples with larger TD-error values can disrupt the probability distribution of training samples. This approach may introduce bias and potentially hinder the convergence of the neural network. Therefore, it is necessary to incorporate importance sampling to adjust the learning rate of the samples. The specific definition is provided in Equation (25).

$$w_i = \left(\frac{1}{N} \cdot \frac{1}{P(i)}\right)^{\beta} \tag{25}$$

In Equation (25), N represents the capacity of the experience replay buffer, while β is a hyperparameter for error correction that ranges between 0 and 1. By distinguishing the importance of experiences using the aforementioned approach, it enhances the learning efficiency of the algorithm.

3.2.6. SAC-LSTM Algorithm Workflow

Based on the SAC algorithm, this paper introduces the SAC-LSTM algorithm by incorporating LSTM neural networks, burn-in, and prioritized experience replay. The workflow of the algorithm and its corresponding pseudocode are presented in Figure 7 and Algorithm 1, respectively. The agent interacts with the environment, generating experience tuples $(s_t, a_t, r_t, s_{t+1}, d)$ that are subsequently stored in the experience replay buffer. By employing prioritized experience replay, the algorithm effectively samples experiences.

The integration of the burn-in mechanism, as indicated by the green circle in Figure 7, enhances the efficiency of the training process.

Algorithm 1: SAC-LSTM algorithm pseudocode

randomly initialize the parameter θ of the actor network and the parameters ϕ_1, ϕ_2 of the Critic network, clear the experience replay buffer $\mathcal{D}$
initialize the target networks $\phi_{targ,1} \leftarrow \phi_1$, $\phi_{targ,2} \leftarrow \phi_2$, set the length of the burn-in data and the lengths l_b, l_t of the training data, set the capacity N of the experience replay buffer
for episode = 1 to M **do**
  initialize the observation s_1 and the hidden state h_1
  for t = 1 to T **do**
    obtain the observation s_t and select an action a_t using the current policy network
    perform action a_t, obtain next observation s_{t+1}, receive reward r_t
    determine whether the current state is a terminal state through the signal d
    store $(s_t, a_t, r_t, s_{t+1}, d)$ into $\mathcal{D}$
  end for
  assign priority $P_t = \max_{i<t} P_i$ to experience $[(s_t, a_t, r_t, s_{t+1}, d), \ldots, (s_{t+L}, a_{t+L}, r_{t+L}, s_{t+1+L}, d)]$
  sample N experiences from $\mathcal{D}$ based on their priority $P(j)$ and reset the hidden state to zero
  Calculate the importance weight w_i for each experience sample
  Scan the previous l_b experiences for each sample and obtain the initial hidden state h_t
  Calculate the target Q-function values $y_1^i, \ldots, y_{l_t}^i$ using the last l_t experiences:

$$y_t^i = r_t^i + \gamma(1-d)\left(\min_{j=1,2} Q_{\phi_{targ,j}}\left(s_{t+1}^i, \widetilde{a}_{t+1}\right) - \alpha \log \pi_\theta\left(\widetilde{a}_{t+1}\middle|s_{t+1}^i\right)\right), i = 1,2, \widetilde{a}_{t+1} \sim \pi_\theta\left(\cdot\middle|s_{t+1}^i\right)$$

  Update the priority $p_i \leftarrow |\delta_i|$ based on the TD-error
  Update Q network using gradient descent method with the following formula:

$$\nabla_{\phi_j} \frac{1}{|N \cdot l_t|} \sum_i w_i \sum_t \left[\left(Q_{\phi_j}(s_t^i, a_t^i) - y_t^i\right)^2\right], \; i = 1,2$$

  Update policy network using gradient descent method with the following formula:

$$\nabla_\theta \frac{1}{|N \cdot l_t|} \sum_i \sum_t \left(\min_{j=1,2} Q_{\phi_j}\left(s_t^i, \widetilde{a}_t\right) - \alpha \log \pi_\theta\left(\widetilde{a}_t\middle|s_t^i\right)\right), i = 1,2, \widetilde{a}_{t+1} \sim \pi_\theta\left(\cdot \mid s_t^i\right)$$

  Update target network $\phi_{\text{targ},j} \leftarrow \rho\phi_j + (1-\rho)\phi_{\text{targ},\,j} j = 1,2$
End for

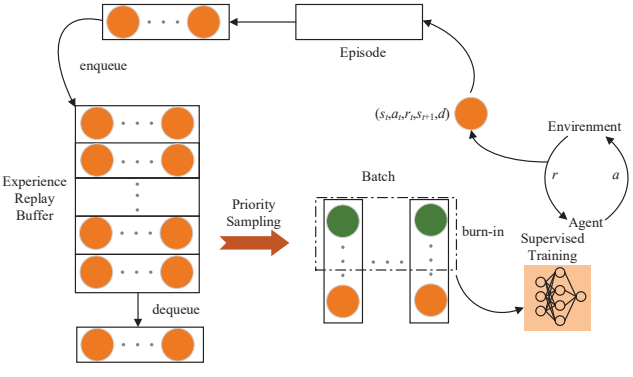


Figure 7. SAC-LSTM algorithm workflow.

3.3. Motion Model of Mobile Robot

Turtlebot3 is a cost-effective and open-source mobile robot that offers a simple yet powerful design. It boasts high expandability and the ability to easily integrate additional sensors as needed. Consequently, this paper selects Turtlebot3 as the platform for algorithm deployment. Operating on a differential drive system, this robot can execute turns, maintain

a constant velocity, and rotate in place by controlling the differential motion of its two rear wheels. The model structure is depicted in Figure 8.

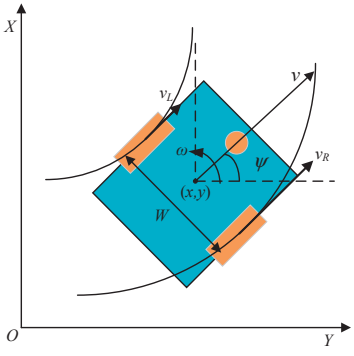


Figure 8. Motion model of the differential drive robot.

The pose of a mobile robot in the world coordinate system is represented by the coordinates $[x, y, \psi]^T$. Here, x and y correspond to the central point of the robot in the X- and Y-axes of the world coordinate system. The parameter ψ denotes the orientation of the robot, indicating the angle between its forward direction (aligned with the linear velocity v) and the X-axis of the world coordinate system. The distance between the two drive wheels of the robot is denoted as W. Assuming the left and right wheel velocities are v_L and v_R, respectively, the linear velocity v of the mobile robot can be determined using the following equation:

$$v = \frac{v_R + v_L}{2}. \tag{26}$$

The lateral angular velocity ω of the mobile robot is given by

$$\omega = \frac{v_R - v_L}{W}. \tag{27}$$

The instantaneous radius R of the mobile robot during motion is given by

$$R = \frac{v}{\omega} = \frac{W(v_R + v_l)}{2(v_R - v_l)}. \tag{28}$$

The motion equation in global coordinates is

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos\psi & 0 \\ \sin\psi & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{29}$$

3.4. Path Planning Algorithm

3.4.1. State Space and Action Space

The state represents the agent's perception of the environment and serves as the foundation for action selection. Designing a well-defined state space is crucial for the agent to learn an optimal strategy. In the context of path planning tasks performed by a mobile robot, the robot relies on sensor input to perceive the surrounding environment. By leveraging this information, the algorithm generates a collision-free path from the initial position to the goal. Consequently, the chosen state space in this paper revolves around two key aspects: sensor perception information and the position of the goal.

For our experimental setup, we employed the Turtlebot3 mobile robot equipped with a laser scanner for environment perception. The laser scanner provides a detection range of $[0°, 360°]$, resulting in a 360-dimensional data representation. To mitigate the issue of

high dimensionality without compromising the effectiveness of capturing environmental information, we adjusted the detection range of the laser scanner to $[-90°, 90°]$ and limited the detection distance to the range of [0.1 m, 3.5 m]. Specifically, the laser scanner's scan data is represented by 20 dimensions, as illustrated in Figure 9.

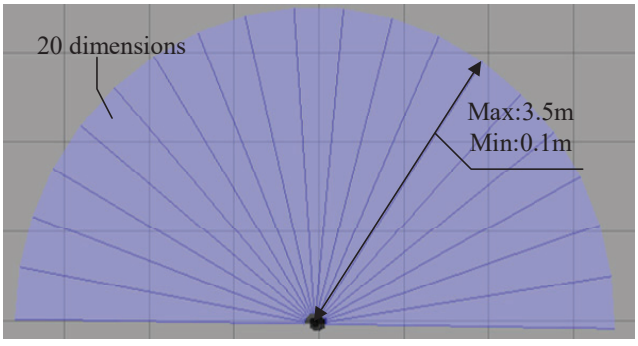


Figure 9. Detection range of the laser sensor on the mobile robot.

In order to expedite the attainment of the designated target point, it is imperative to provide guidance to the robot regarding its trajectory. Thus, this paper adopts the utilization of the heading angular deviation Rad^t_{diff} between the frontal orientation of the mobile robot and the target point and the distance between the robot and the target point as the fundamental states for this purpose. By employing the mobile robot's odometry, the coordinates $(x^t_{robot}, y^t_{robot})$ of the robot can be determined, along with the coordinates (x_{goal}, y_{goal}) of the target point as shown in Figure 10. Consequently, the distance dis_t between the robot and the target point can be computed.

$$dis_t = \sqrt{\left(x_{goal} - x^t_{robot}\right)^2 + \left(y_{goal} - y^t_{robot}\right)^2} \tag{30}$$

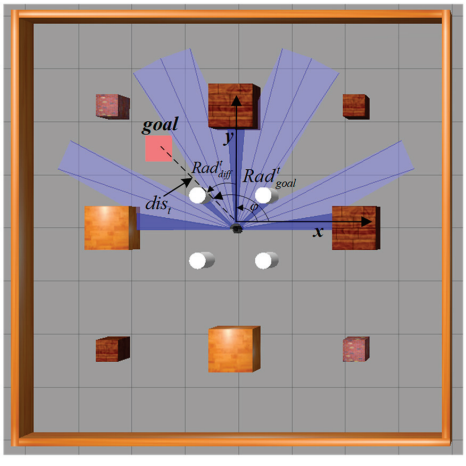


Figure 10. State of the mobile robot.

The heading angle yaw_t can be obtained from the odometer, but the quadratic data obtained cannot be used directly, and have to be converted into a Eulerian angle first.

$$yaw_t = euler\_from\_quaternion(orientation) \tag{31}$$

The angle Rad^t_{goal} between the mobile robot and the target point can be obtained by the inverse tangent function.

$$Rad^t_{goal} = \arctan\left(x_{goal} - x^t_{robot}, y_{goal} - y^t_{robot}\right) \tag{32}$$

The heading angular deviation is the difference between the angle Rad^t_{goal} and the orientation angle φ.

$$Rad^t_{diff} = Rad^t_{goal} - \varphi \tag{33}$$

In this paper, we enhance the state representation of a mobile robot by incorporating the linear velocity and angular velocity from the previous time step. This addition allows for a correspondence between the rewards provided by the environment and the actions performed by the mobile robot. Ultimately, the state space of the mobile robot is defined as a 20-dimensional vector comprising radar detection data, heading angular deviation, linear velocity, and angular velocity from the previous time step, as well as the distance between the target point and the mobile robot.

$$S_t = [scan_t, v_{t-1}, \omega_{t-1}, Rad^t_{diff}, dis_t] \tag{34}$$

Many previous deep reinforcement learning algorithms have used a discretized action space in which the linear and angular velocities of mobile robots were divided into several orders of magnitude. Although this approach is relatively simple, it ignores the fact that mobile robots output continuous actions. To make the simulation more realistic, this paper presents a network model that produces continuous angular and linear velocities. The linear velocity has a range of 0 to 0.22 m/s, while the angular velocity ranges from -2 to 2 rad/s, where positive angular velocity indicates clockwise rotation and negative angular velocity indicates counterclockwise rotation. The proposed approach is intended to bring the simulation closer to real-world scenarios.

3.4.2. Reword Function

The reward function plays a critical role in the success of deep reinforcement learning algorithms as it serves as the benchmark for evaluating agent performance. Similar to constraints used in traditional path planning tasks, the reward function guides agents by indicating which actions to avoid and which ones to pursue given the current state. In this paper, the proposed reward function is composed of two distinct components whose sum constitutes the overall reward:

$$R_{total} = R_a + R_b \tag{35}$$

The specific expressions of R_a and R_b are shown in Equations (36) and (37). d_o represents the distance threshold for reaching the target point, while $d_{\min}$ represents the threshold for avoiding collisions with obstacle. $d_{\max}$ is the safe distance threshold from the obstacle. Furthermore, d_t denotes the current distance between the moving robot and the target point, and d_{t-1} represents the previous time's distance. Additionally, d^t_{smin} is the minimum distance recorded from the radar scan at the current time. The reward coefficients are indicated as η_1, η_2, and η_3.

$$R_a = \begin{cases} r_{\text{a}} & if(d_t < d_o) \\ r_{\text{c}} & if\left(d^t_{\text{smin}} < d_{\min}\right) \\ \eta_1(d_{t-1} - d_t) + \eta_2(\pi - |\varphi|)\ otherwise \end{cases} \tag{36}$$

$$R_b = \begin{cases} \eta_3\left(d_{\text{smin}}^t - d_{\max}\right) if\left(d_{\max} > d_{\text{smin}}^t > d_{\min}\right) \\ 0 \qquad otherwise \end{cases} \tag{37}$$

This paper proposes a design approach where mobile robots reaching a designated target point or encountering obstacles are associated with sparse rewards, while dense rewards are assigned for other scenarios. This particular design methodology ensures algorithmic stability throughout the training process.

The sparse reward setting is relatively straightforward, involving the assignment of an immediate reward to the mobile robot upon reaching the target point or colliding with an obstacle. When the distance between the mobile robot and the target point is below a threshold value d_o, it is considered to have successfully reached the target point, resulting in a positive reward r_{a}. Conversely, if the distance between the mobile robot and the obstacle is below a threshold value $d_{\min}$, a collision is assumed, leading to a negative reward r_c.

The dense rewards comprise the distance reward, orientation angle reward, and safety reward. The distance reward, denoted as $\eta_1(d_{t-1} - d_t)$, is contingent upon the mobile robot's velocity while moving towards the target point, with higher rewards for faster velocities and lower rewards for slower velocities. The orientation angle reward, denoted as $\eta_2(\pi - |\varphi|)$, is determined by the orientation angle φ, where the reward is maximized at 0 when the mobile robot is directly facing the target point, and minimized at π when the mobile robot is facing away from the target point. The sum $\eta_1(d_{t-1} - d_t) + \eta_2(\pi - |\varphi|)$ of the distance reward and orientation angle reward aims to expedite the mobile robot's arrival at the target point. Additionally, the safety reward, represented by $\eta_3\left(d_{\text{smin}}^t - d_{\max}\right)$, imposes a negative reward when the distance between the mobile robot and the obstacle is below the designated threshold $d_{\max}$. Moreover, the negative reward increases as the distance decreases. The purpose of the safety reward is to maintain a safe distance from obstacles during the path planning process.

The pseudocode for the reward function is presented in Algorithm 2.

Algorithm 2: Reward function pseudocode

Input: initialized reward coefficients η_1, η_2, η_3, thresholds d_o,$d_{\max}$, action a_t
Output: reward r $d_{\min}$

```
for each step of reward do
    performs action a_t
    calculate dense reward r = η1(d_{t-1} − d_t) + η2(π − |φ|)
    if d_max > d^t_smin > d_min then
        r+ = η3(d^t_smin − d_max)
    else if d_t < d_o then
        r+ = r_a
    else if d^t_smin < d_min then
        r+ = r_c
    end
    end
    return reward r
end for
```

3.4.3. Path Planning Algorithm Workflow

The path planning algorithm, based on deep reinforcement learning, essentially involves learning the optimal strategy for the mobile robot to reach the target point quickly through interactive exploration with the environment. The algorithm pseudocode for path planning, utilizing SAC-LSTM, is depicted in Algorithm 3.

Algorithm 3: Path planning algorithm pseudocode

```
initialize system parameters
introduce LSTM network to optimize the network architecture
for N_t < N_tmax do
    initialize the robot's position
    get environmental information
    perform actions through the policy network
    obtain new state and reward
    store experiences in the experience buffer
    sample experiences based on their priority
     train the network using the burn-in mechanism
     if the robot reaches the target point then
         generate a new target point
         go to step 5
     else if the terminal state is not met or N_s < N_smax then
             go to step 5
         else
             go to step 3
         end
    end
end for
```

At the beginning of each episode, the mobile robot is positioned at a predefined starting point and subsequently navigates towards a randomly generated target destination. Throughout this process, the mobile robot relies on a laser radar to perceive its surrounding environment and utilizes the state information as input to the policy network of the SAC-LSTM algorithm, producing a corresponding robot action. Following interaction with the environment, the mobile robot transitions to the next state and receives a reward based on a predefined reward function. The acquired experiences are stored in the experience replay buffer, and significant experiences are extracted using a prioritized experience replay mechanism. The SAC-LSTM algorithm is trained using the burn-in strategy. Upon reaching the target point, a new target point is randomly generated in the environment, and the robot moves from its current location towards the newly selected target. The training episode terminates only when the robot collides with an obstacle or when the current step count N_s reaches the designated maximum value N_{smax}. The training process concludes when the number of training episodes N_t surpasses the predetermined maximum value N_{tmax}.

4. Experiments and Discussion

4.1. Simulation Platform

The experimental software environment for this paper included Ubuntu 18.04, CUDA 10.1, Pytorch 3.7, and ROS Kinetic. The hardware utilization comprised an AMD R7-5800H CPU and a GeForce GTX 3060 GPU with 6G of memory. The experiment employed the Turtlebot3 robot based on ROS, which obtains the environmental information around it with the help of laser radar. The 3D model of the mobile robot was loaded into the 'empty.world' in ROS, as depicted in Figure 11.

This paper utilizes Gazebo9 software to set up experimental environments. Three experimental environments, depicted in Figure 12, were built for testing the algorithm's effectiveness in mobile robot path planning in indoor environments. These environments are an obstacle-free environment, a static obstacle environment, and a dynamic obstacle environment; all three are square areas measuring 8 m in length. The obstacle-free environment has only four surrounding walls. The static obstacle environment features twelve stationary obstacles, consisting of four cylinders with a diameter of 0.3 m and a height of 0.5 m, four small cubes measuring 0.5 m in edge-length, and four large cubes measuring 0.8 m in edge-length, which are added to the obstacle-free environment. In contrast, the dynamic obstacle environment features moving obstacles. The four cylinders rotate counterclockwise at a speed of 0.5 rad/s around the origin of the coordinate system (the

center of the environment), as illustrated by the smaller white dashed circles in Figure 12. Concurrently, four large cubes move clockwise at the same speed, following the trajectory shown by the larger white dashed circles with arrows in Figure 12. Meanwhile, the four small cubes within this environment remain stationary.

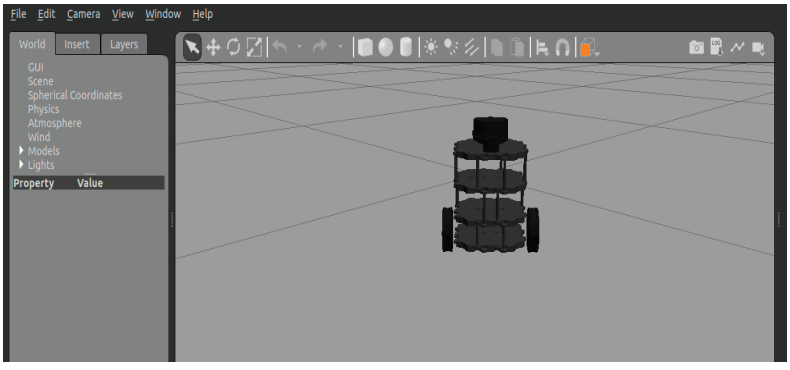
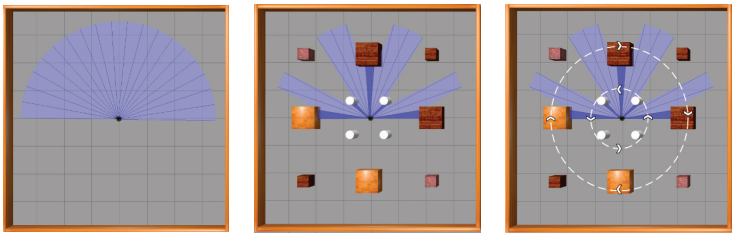


Figure 11. Turtlebot3 3D model.



(**a**) (**b**) (**c**)

Figure 12. Experimental environments: (**a**) obstacle-free environment; (**b**) static obstacle environment; (**c**) dynamic obstacle environment.

Deep reinforcement learning involves agents gathering training data by interacting with their real-world environments. In a 3D experimental environment created by Gazebo, data acquisition can become a time-consuming and computationally expensive process. Consequently, this paper employs a time acceleration technique to reduce the duration of training. By default, Gazebo's real-time update rate is 1000, and the max step size value is 0.001. When these settings are multiplied, the resulting ratio between the simulation time and the actual time is 1. However, in order to expedite simulations, the value for max step size is adjusted to 0.005, leading to a fivefold increase in simulation speed.

The specific experimental parameters are shown in Table 1.

Table 1. Experimental parameters of mobile robot.

Parameter	Value
Discount factor γ	0.99
Learning rate l_r	0.001
Priority regulating coefficient α_p	0.6
β	0.4
Experience buffer capacity (SAC)	20,000
Experience buffer capacity (SAC-LSTM)	5000
Batch size (SAC)	512

Table 1. *Cont.*

Parameter	Value
Batch size (SAC-LSTM)	32
Burn-in data length l_b	16
Training length	16
Maximum steps N_{smax}	500 l_t
Optimizer	Adam
Reward coefficient η_1	4
Reward coefficient η_2	0.1
Reward coefficient η_3	-3
Reward for reaching the target point r_{a}	100
Reward for collision with obstacle r_c	-50
Distance threshold for reaching the target point d_o	0.15
Minimum distance threshold from obstacle d_{min}	0.15
Safe distance threshold from obstacle d_{max}.	0.3

4.2. Computational Complexity

To quantify the computational complexity of SAC and SAC-LSTM algorithms, the contribution of each component to the overall computational load can be analyzed based on five key factors, as follows:

1. Computational Load of Neural Networks

Due to the larger batch size (512) and experience buffer capacity (20,000), the SAC algorithm requires processing more data in the network training steps, leading to an increased computational load. When it comes to the SAC-LSTM algorithm, the inclusion of LSTM layers introduces additional time-dependency computations. A smaller batch size (32) might reduce the computational load per training iteration, but the burn-in process and LSTM's time dependencies increase the computational load per batch.

2. Prioritized Experience Replay

Prioritized experience replay requires additional computations to maintain a priority queue and update it after each learning step, increasing the computational load, especially with a larger experience buffer.

3. Learning Parameters

The parameters, a learning rate of 0.001 and a discount factor of 0.99, mainly affect the convergence speed and stability of the algorithm, but have a relatively minor direct impact on computational load.

4. Reward Mechanism

The reward coefficients and thresholds influence the efficiency of learning, but have a limited direct impact on computational load.

5. Optimizer

The Adam optimizer is a computationally efficient optimizer but has a higher computational complexity compared to simpler ones such as SGD (Stochastic Gradient Descent).

Overall, the SAC-LSTM algorithm involves higher per-batch computational loads due to time-dependency processing with LSTM, smaller batch sizes, and burn-in processing. The SAC algorithm, although having larger data volumes per batch, might have a slightly lower per-batch computational load than SAC-LSTM, owing to the absence of complex time-series processing. When running both algorithms on the same computer platform, SAC-LSTM consumes approximately 10% more computation time per batch compared to SAC. This finding aligns closely with the qualitative analysis results mentioned above.

4.3. Experimental Results and Analysis

4.3.1. Obstacle-Free Experiment

In this paper, we chose the average reward as the evaluation metric for our algorithm. A higher value confirms better performance of the algorithm. The average reward is calculated from Equation (38). Within each episode, $T = \min(N_{s\max}, T)$ represents the number of steps taken by the agent, and r is the reward received for each step.

$$\text{reward} = \sum_{i=1}^{T} r/T \tag{38}$$

The robot's starting point is the origin, and the target point is generated at random. During a single episode, once the robot reaches the target point, the environment generates a new target point randomly. As a result, the robot may reach multiple target points within a single episode.

Two reinforcement learning algorithms, SAC and SAC-LSTM, were trained for 1000 episodes in the obstacle-free environment. Their average reward curves are illustrated in Figure 13. During the initial training phase, both algorithms demonstrated a decreasing trend in their average reward. This was attributed to the robot frequently circling in place or moving away from the target point, which was observed within the first 10 episodes. However, after that point, both algorithms exhibited a remarkable increase in their average reward. The SAC-LSTM algorithm reached its reward peak and convergence after 130 episodes, whereas the SAC algorithm achieved the same after 180 episodes. Notably, the average reward of the SAC-LSTM algorithm surpassed that of the SAC algorithm after convergence. This observation suggests that the SAC-LSTM algorithm enabled the robot to reach the target point more frequently during each episode's path planning process, resulting in better path planning performance. Although both algorithms accomplished the task in the obstacle-free environment, the SAC-LSTM algorithm converged faster and yielded a higher average reward after convergence.

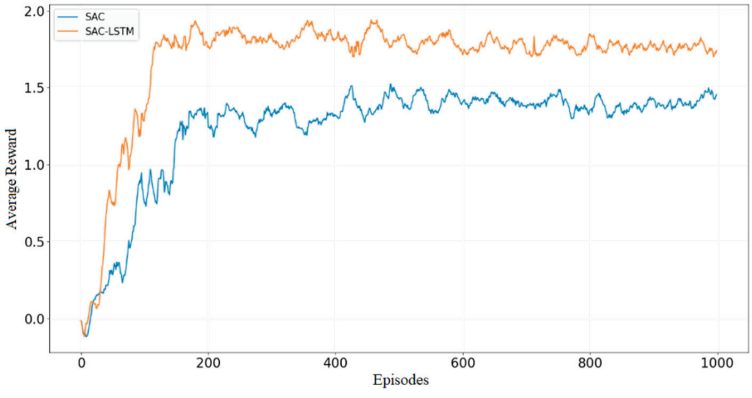


Figure 13. Average reward in the obstacle-free experiment.

To further evaluate the model's performance, the SAC and SAC-LSTM algorithms underwent 200 tests in the obstacle-free environment. For each test, the target points were randomly generated, and the results are presented in Table 2. Our evaluation reveals that the path planning success rate for the SAC algorithm was 97%. In contrast, the proposed SAC-LSTM algorithm yielded better performance, achieving a success rate of 100%.

Table 2. Obstacle-free experiment results.

Algorithm	Success Number	Success Rate
SAC	194	97%
SAC-LSTM	200	100%

Illustrated in Figure 14 is the motion process of the mobile robot towards the target point in the obstacle-free environment based on the SAC-LSTM algorithm. In the figure, the black dot represents the mobile robot, while the blue sector illustrates the range of the radar scan. The red square indicates the location of the target point. Notably, the mobile robot achieves optimal pathway and accurately reaches the target point from the starting point.

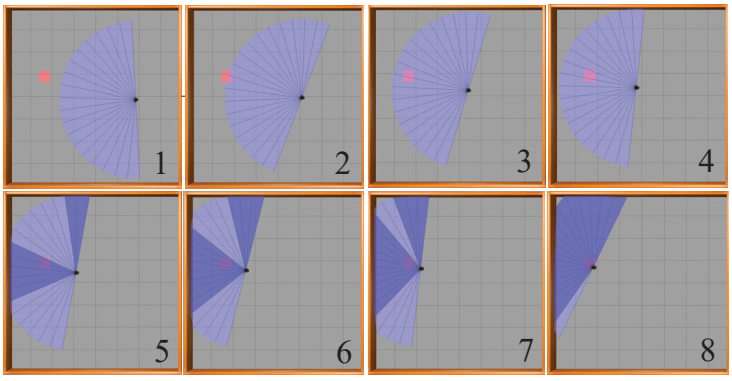


Figure 14. Movement process of the mobile robot in the obstacle-free environment.

4.3.2. Static Obstacle Experiment

The two algorithms were trained for 1000 episodes in the obstacle environment illustrated in Figure 12b. The plotted average reward curves of the two algorithms are depicted in Figure 15. Notably, the static obstacle environment is significantly more complex than the obstacle-free environment since collision with the surrounding obstacles is more frequent during robot exploration. As expected, greater exploration time was required. The SAC-LSTM algorithm began to converge from the 20th episode. Interestingly, its converging process had a higher average reward than the SAC algorithm, finishing the process at the 380th episode. On the other hand, the SAC algorithm had reduced fluctuations and began to converge from the 30th episode, while the converging process completed at the 500th episode. Impressively, the SAC-LSTM algorithm yielded a faster convergence rate and a higher average reward after convergence than the SAC algorithm.

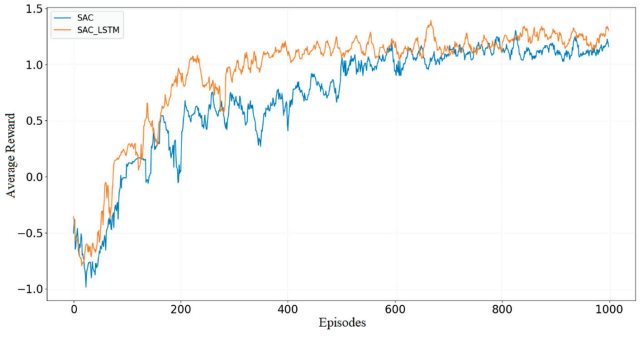


Figure 15. Average reward in static obstacle environment.

Two algorithms were tested 200 times in the static obstacle environment, where the target points were randomly generated. The test results are presented in Table 3, indicating that the SAC algorithm's success rate was only 88.5%, while the proposed SAC-LSTM algorithm achieved a success rate of 95.5%.

Table 3. Static obstacle experiment results.

Algorithm	Success Number	Success Rate
SAC	183	88.5%
SAC-LSTM	193	95.5%

Figure 16 illustrates the movement process of a mobile robot based on the SAC-LSTM algorithm, from its starting point to the target point in the static obstacle environment. As shown, the mobile robot avoided the obstacles in the environment and reached the target point via the optimal path.

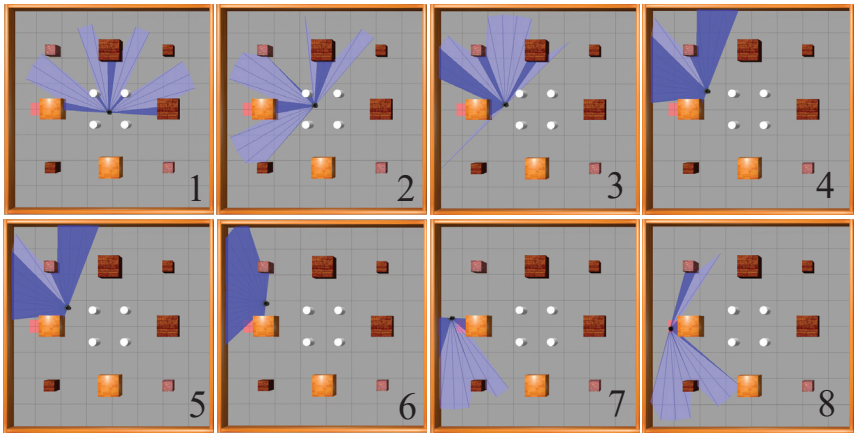


Figure 16. Movement process of the mobile robot in the static obstacle environment.

4.3.3. Dynamic Obstacle Experiment

The dynamic obstacle environment was constructed as shown in Figure 12c to simulate a realistic path planning scenario. The SAC and SAC-LSTM algorithms underwent 1000 episodes of training, with the training results presented in Figure 17. As shown in the figure, the difficulty of path planning increased due to the moving obstacles, resulting in significant fluctuations in the learning curves of both algorithms. The SAC-LSTM algorithm began to converge after the 75th episode when the fluctuations decreased, and it achieved convergence by the 510th episode. The SAC algorithm began to converge after the 95th episode and achieved convergence by the 710th episode. The proposed SAC-LSTM algorithm demonstrated a faster convergence rate than the SAC algorithm, with a slightly higher average reward after convergence.

A total of 200 tests were also conducted using both algorithms in the dynamic obstacle environment, with the target points generated randomly. Table 4 shows the success rates of both algorithms, with the SAC algorithm achieving a success rate of only 78.5%, while the SAC-LSTM algorithm achieved a success rate of 89%.

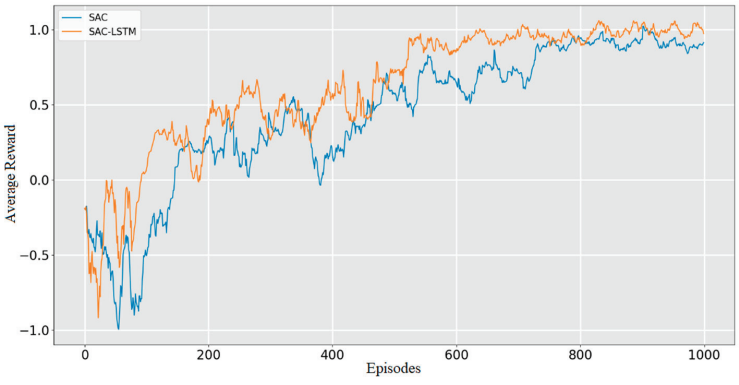


Figure 17. Average reward in the dynamic obstacle environment.

Table 4. Dynamic obstacle experiment results.

Algorithm	Success Number	Success Rate
SAC	157	78.5%
SAC-LSTM	178	89%

Figure 18 illustrates the movement process of a mobile robot driven by the SAC-LSTM algorithm, from its starting point to the target point in a dynamic obstacle environment. As shown, the mobile robot was able to avoid the moving obstacles and reach the target point via the optimal path.

As completing path planning tasks in dynamic environments is more challenging and requires higher performance from path planning algorithms, and as dynamic environments better reflect the actual work environment of mobile robots, in order to better test the performance of the two algorithms, an additional set of experiments was conducted on top of the initial experiment measuring success rate, which tested the path length, planning time, and number of times the robot reached the target point. Ten target points were randomly generated in the dynamic obstacle environment, all located near the moving obstacles at the periphery, so that each time the robot moved to a target point, interference from the moving obstacles was encountered, increasing the reliability of the experiment. The specific locations of the target points are illustrated in Figure 19.

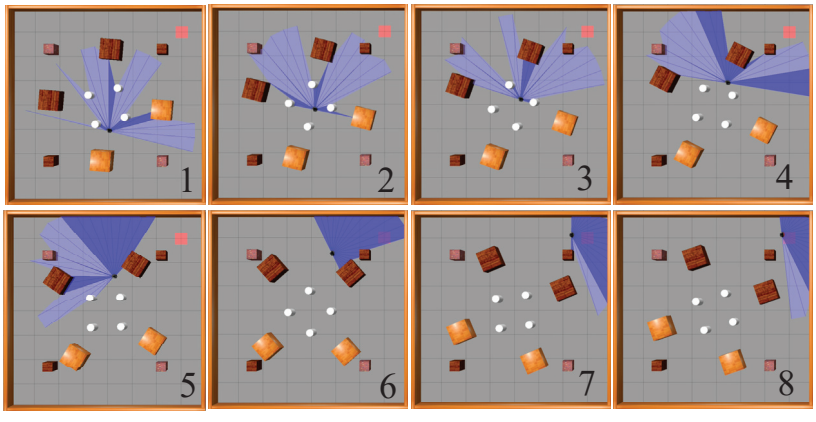


Figure 18. Movement process of the mobile robot in the dynamic obstacle environment.

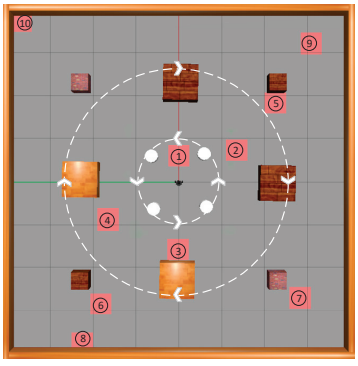


Figure 19. Locations of 10 target points.

The target point numbers in Figure 19 were sorted according to the Euclidean distance between each target point and the starting point of the mobile robot, which is set as the origin. Thirty experimental trials were conducted for each of the target points, with the path length, planning time, and success rates recorded. The path length and planning time were averaged from the successfully completed path planning tasks, and specific test results can be found in Tables 5 and 6. Among the 10 target points tested, the SAC-LSTM algorithm achieved both shorter average path lengths and planning times, as well as higher success rates, compared to the SAC algorithm. This indicates that the path planning performance of the SAC-LSTM algorithm is superior to that of the SAC algorithm, enabling the mobile robot to reach its designated target point in less time and with a shorter route. Notably, for the 10th target point, the SAC algorithm was unable to guide the mobile robot to its target point based solely on the current state information, whereas the SAC-LSTM algorithm, which incorporates the LSTM network, has memory capability to consider both historical and current states to make better decisions, and thus guided the robot to complete its path planning task.

Based on the results of the three simulation experiments, the trained mobile robot was able to successfully complete the path planning task in all three environments, and the improved SAC-LSTM algorithm demonstrated significant enhancements in both path planning success rate and convergence speed. In particular, in dynamic and complex scenarios, the SAC-LSTM algorithm exhibited shorter planning time, shorter planning paths, and a higher number of instances where the target point was reached.

Table 5. Test results of the SAC-LSTM algorithm.

Target Point Number	Target Point Location	Path Length (m)	Planning Time (s)	Success Number
1	(0.60, 0.00)	0.64	4.36	30
2	(0.72, −1.35)	1.61	8.89	30
3	(−1.59, 0.02)	1.76	10.47	29
4	(−0.89, 1.65)	1.90	10.98	30
5	(1.82, −2.24)	2.94	16.61	27
6	(−2.86, 1.83)	3.65	20.95	24
7	(−2.69, −2.83)	4.33	24.72	24
8	(−3.74, 2.24)	4.82	27.54	23
9	(3.21, −3.09)	4.85	27.42	25
10	(3.72, 3.68)	5.44	31.22	29

Table 6. Test results of the SAC algorithm.

Target Point Number	Target Point Location	Path Length (m)	Planning Time (s)	Success Number
1	(0.60, 0.00)	0.67	5.02	30
2	(0.72, −1.35)	1.75	9.65	30
3	(−1.59, 0.02)	1.89	11.28	24
4	(−0.89, 1.65)	2.03	11.87	27
5	(1.82, −2.24)	3.05	17.68	23
6	(−2.86, 1.83)	3.69	21.83	24
7	(−2.69, −2.83)	4.40	25.08	28
8	(−3.74, 2.24)	4.95	29.63	18
9	(3.21, −3.09)	4.95	27.8	25
10	(3.72, 3.68)	/	/	0

5. Conclusions

This paper presents the SAC-LSTM algorithm and develops a path planning algorithm framework for mobile robots, addressing the limitations of the SAC algorithm in path planning tasks. The proposed algorithm incorporates an LSTM network with memory capability, a burn-in mechanism, and a prioritized experience replay mechanism. By integrating historical and current states, the LSTM network enables more effective path planning decisions. The burn-in mechanism preheats the LSTM network's hidden state before training, addressing memory depreciation and enhancing the algorithm's performance. The prioritized experience replay mechanism accelerates algorithm convergence by emphasizing crucial experiences. A motion model for the Turtlebot3 mobile robot was established, and the state space, action space, reward function, and overall process of the SAC-LSTM algorithm were designed. To enhance the realism of the experimental scenarios, three environments were created, including obstacle-free, static obstacle, and dynamic obstacle scenarios. There are multiple stationary and moving obstacles of various sizes and shapes in the dynamic scenarios. The algorithm was subsequently trained and tested in these settings. The experimental results demonstrated that the SAC-LSTM algorithm outperformed the SAC algorithm in convergence speed and path planning success rate across all three scenarios with roughly the same computational cost. Furthermore, in an additional dynamic obstacle experiment, the SAC-LSTM algorithm exhibited shorter planning times, more efficient paths, and an increased number of instances where the target point was reached, indicating superior path planning performance.

Despite these advancements, certain limitations persist within this paper. The experiments relied solely on 2D lidar for environmental data, which may lead to inaccuracies when dealing with irregularly shaped obstacles. Future research could employ multi-sensor fusion to obtain more comprehensive environmental information. Additionally, the paper does not address the several sources of noise and uncertainty present in real-world environments, including sensor noise, environmental fluctuations, and uncertainty in obstacle movement. The algorithm needs sufficient robustness to handle these factors to perform well in practical settings. Moreover, the experiments were conducted exclusively in a simulated environment. To fully assess the effectiveness of the mobile robot in completing path planning tasks, it is essential to transfer the trained model to a real-world scenario.

Author Contributions: Conceptualization, Y.Z. and P.C.; methodology, Y.Z. and P.C.; software, Y.Z.; validation, Y.Z.; formal analysis, Y.Z.; investigation, Y.Z.; resources, Y.Z.; data curation, Y.Z.; writing—original draft preparation, Y.Z.; writing—review and editing, Y.Z. and P.C.; visualization, Y.Z.; supervision, P.C.; project administration, P.C.; funding acquisition, P.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 62163014.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

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

References

1. Liu, L.; Wang, X.; Yang, X.; Liu, H.; Li, J.; Wang, P. Path Planning Techniques for Mobile Robots: Review and Prospect. *Expert Syst. Appl.* **2023**, *227*, 120254. [CrossRef]
2. Dijkstra, E.W. A Note on Two Problems in Connexion with Graphs. In *Edsger Wybe Dijkstra: His Life, Work, and Legacy*; Association for Computing Machinery: New York, NY, USA, 2022; Volume 45, pp. 287–290. ISBN 978-1-4503-9773-5.
3. 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]
4. Stentz, A. Optimal and Efficient Path Planning for Partially-Known Environments. In Proceedings of the 1994 IEEE International Conference on Robotics and Automation, San Diego, CA, USA, 8–13 May 1994; Volume 4, pp. 3310–3317.
5. Kavraki, L.E.; Svestka, P.; Latombe, J.-C.; Overmars, M.H. Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. *IEEE Trans. Robot. Autom.* **1996**, *12*, 566–580. [CrossRef]
6. LaValle, S.M.; Kuffner, J.J. Randomized Kinodynamic Planning. *Int. J. Robot. Res.* **2001**, *20*, 378–400. [CrossRef]
7. Bohlin, R.; Kavraki, L.E. Path Planning Using Lazy PRM. In Proceedings of the 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065), San Francisco, CA, USA, 24–28 April 2000; Volume 1, pp. 521–528.
8. Liu, B.; Feng, W.; Li, T.; Hu, C.; Zhang, J. A Variable-Step RRT* Path Planning Algorithm for Quadrotors in Below-Canopy. *IEEE Access* **2020**, *8*, 62980–62989. [CrossRef]
9. Wang, C.; Soh, Y.C.; Wang, H.; Wang, H. A Hierarchical Genetic Algorithm for Path Planning in a Static Environment with Obstacles. In Proceedings of the IEEE CCECE2002. Canadian Conference on Electrical and Computer Engineering. Conference Proceedings (Cat. No. 02CH37373), Winnipeg, MB, Canada, 12–15 May 2002; Volume 3, pp. 1652–1657.
10. Niu, L.; Xiong, L. Optimisation and Application Research of Ant Colony Algorithm in Vehicle Routing Problem. *Int. J. Comput. Sci. Math.* **2021**, *13*, 177–193. [CrossRef]
11. Liang, Y.; Wang, L. Applying Genetic Algorithm and Ant Colony Optimization Algorithm into Marine Investigation Path Planning Model. *Soft Comput.* **2020**, *24*, 8199–8210. [CrossRef]
12. Khatib, O. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. In *Autonomous Robot Vehicles*; Cox, I.J., Wilfong, G.T., Eds.; Springer: New York, NY, USA, 1986; pp. 396–404. ISBN 978-1-4613-8999-6.
13. Zha, M.; Wang, Z.; Feng, J.; Cao, X. Unmanned Vehicle Route Planning Based on Improved Artificial Potential Field Method. *J. Phys. Conf. Ser.* **2020**, *1453*, 012059. [CrossRef]
14. Zhao, T.; Li, H.; Dian, S. Multi-Robot Path Planning Based on Improved Artificial Potential Field and Fuzzy Inference System. *J. Intell. Fuzzy Syst.* **2020**, *39*, 7621–7637. [CrossRef]
15. Afifi, S.; Dang, D.-C.; Moukrim, A. A Simulated Annealing Algorithm for the Vehicle Routing Problem with Time Windows and Synchronization Constraints. In *Learning and Intelligent Optimization*; Nicosia, G., Pardalos, P., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 259–265.
16. Jun, S.; Jian, L. An Improved Self-Adaptive Particle Swarm Optimization Algorithm with Simulated Annealing. In Proceedings of the 2009 Third International Symposium on Intelligent Information Technology Application, NanChang, China, 21–22 November 2009; Volume 3, pp. 396–399.
17. Zhang, L.; Zhang, Y.; Li, Y. Path Planning for Indoor Mobile Robot Based on Deep Learning. *Optik* **2020**, *219*, 165096. [CrossRef]
18. Ferreira, J.; Junior, A.A.F.; Galvão, Y.M.; Barros, P.; Murilo Maciel Fernandes, S.; Fernandes, B.J.T. Performance Improvement of Path Planning Algorithms with Deep Learning Encoder Model. In Proceedings of the 2020 Joint IEEE 10th International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob), Valparaiso, Chile, 26–30 October 2020; pp. 1–6.
19. Bai, X.; Jiang, H.; Cui, J.; Lu, K.; Chen, P.; Zhang, M. UAV Path Planning Based on Improved A* and DWA Algorithms. *Int. J. Aerosp. Eng.* **2021**, *2021*, e4511252. [CrossRef]
20. Lee, D.H.; Lee, S.S.; Ahn, C.K.; Shi, P.; Lim, C.-C. Finite Distribution Estimation-Based Dynamic Window Approach to Reliable Obstacle Avoidance of Mobile Robot. *IEEE Trans. Ind. Electron.* **2021**, *68*, 9998–10006. [CrossRef]
21. Tai, L.; Liu, M. Towards Cognitive Exploration through Deep Reinforcement Learning for Mobile Robots. *arXiv* **2016**, arXiv:1610.01733.
22. Wang, W.; Wu, Z.; Luo, H.; Zhang, B. Path Planning Method of Mobile Robot Using Improved Deep Reinforcement Learning. *J. Electr. Comput. Eng.* **2022**, *2022*, e5433988. [CrossRef]
23. Lei, X.; Zhang, Z.; Dong, P. Dynamic Path Planning of Unknown Environment Based on Deep Reinforcement Learning. *J. Robot.* **2018**, *2018*, e5781591. [CrossRef]
24. Tai, L.; Paolo, G.; Liu, M. Virtual-to-Real Deep Reinforcement Learning: Continuous Control of Mobile Robots for Mapless Navigation. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 31–36.

25. Gan, Z.; Li, B.; Neretin, E.S.; Dyachenko, S.A. UAV Maneuvering Target Tracking Based on Deep Reinforcement Learning. *J. Phys. Conf. Ser.* **2021**, *1958*, 012015. [CrossRef]
26. Zhang, Q.; Lin, J.; Sha, Q.; He, B.; Li, G. Deep Interactive Reinforcement Learning for Path Following of Autonomous Underwater Vehicle. *IEEE Access* **2020**, *8*, 24258–24268. [CrossRef]
27. Chen, W.; Zhou, S.; Pan, Z.; Zheng, H.; Liu, Y. Mapless Collaborative Navigation for a Multi-Robot System Based on the Deep Reinforcement Learning. *Appl. Sci.* **2019**, *9*, 4198. [CrossRef]
28. Wang, Y.; He, H.; Sun, C. Learning to Navigate through Complex Dynamic Environment with Modular Deep Reinforcement Learning. *IEEE Trans. Games* **2018**, *10*, 400–412. [CrossRef]
29. Gao, J.; Ye, W.; Guo, J.; Li, Z. Deep Reinforcement Learning for Indoor Mobile Robot Path Planning. *Sensors* **2020**, *20*, 5493. [CrossRef]
30. Shi, H.; Shi, L.; Xu, M.; Hwang, K.-S. End-to-End Navigation Strategy with Deep Reinforcement Learning for Mobile Robots. *IEEE Trans. Ind. Inform.* **2019**, *16*, 2393–2402. [CrossRef]
31. de Jesus, J.C.; Kich, V.A.; Kolling, A.H.; Grando, R.B.; Cuadros, M.A.d.S.L.; Gamarra, D.F.T. Soft Actor-Critic for Navigation of Mobile Robots. *J. Intell. Robot. Syst.* **2021**, *102*, 31. [CrossRef]
32. Park, K.-W.; Kim, M.; Kim, J.-S.; Park, J.-H. Path Planning for Multi-Arm Manipulators Using Soft Actor-Critic Algorithm with Position Prediction of Moving Obstacles via LSTM. *Appl. Sci.* **2022**, *12*, 9837. [CrossRef]
33. Khan, A.H.; Li, S.; Chen, D.; Liao, L. Tracking Control of Redundant Mobile Manipulator: An RNN Based Metaheuristic Approach. *Neurocomputing* **2020**, *400*, 272–284. [CrossRef]
34. Khan, A.H.; Li, S.; Luo, X. Obstacle Avoidance and Tracking Control of Redundant Robotic Manipulator: An RNN-Based Metaheuristic Approach. *IEEE Trans. Ind. Inform.* **2020**, *16*, 4670–4680. [CrossRef]
35. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
36. Kapturowski, S.; Ostrovski, G.; Quan, J.; Munos, R.; Dabney, W. Recurrent Experience Replay in Distributed Reinforcement Learning. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
37. Schaul, T.; Quan, J.; Antonoglou, I.; Silver, D. Prioritized Experience Replay. *arXiv* **2016**, arXiv:1511.05952.

sensors

MDPI

Article

A Software Platform for Quadruped Robots with Advanced Manipulation Capabilities

Jae-Bong Yi, Shady Nasrat, Min-seong Jo and Seung-Joon Yi *

Department of Electrical Engineering, Pusan National University, Busan 46241, Republic of Korea; niteofhunter@pusan.ac.kr (J.-B.Y.); shadyloai@pusan.ac.kr (S.N.); jominseong@pusan.ac.kr (M.-s.J.)
* Correspondence: seungjoon.yi@pusan.ac.kr; Tel.: +82-51-510-7917

Abstract: Recently, a diverse range of robots with various functionalities have become a part of our daily lives. However, these robots either lack an arm or have less capable arms, mainly used for gestures. Another characteristic of the robots is that they are wheeled-type robots, restricting their operation to even surfaces. Several software platforms proposed in prior research have often focused on quadrupedal robots equipped with manipulators. However, many of these platforms lacked a comprehensive system combining perception, navigation, locomotion, and manipulation. This research introduces a software framework for clearing household objects with a quadrupedal robot. The proposed software framework utilizes the perception of the robot's environment through sensor inputs and organizes household objects to their designated locations. The proposed framework was verified by experiments within a simulation environment resembling the conditions of the RoboCup@Home 2021-virtual competition involving variations in objects and poses, where outcomes demonstrate promising performance.

Keywords: quadruped robot; organize objects; mobile manipulation

Citation: Yi, J.-B.; Nasrat, S.; Jo, M.-s.; Yi, S.-J. A Software Platform for Quadruped Robots with Advanced Manipulation Capabilities. *Sensors* **2023**, *23*, 8247. https://doi.org/10.3390/s23198247

Academic Editors: Yuanlong Xie, Shiqi Zheng, Zhaozheng Hu and Shuting Wang

Received: 12 September 2023
Revised: 28 September 2023
Accepted: 2 October 2023
Published: 4 October 2023
Corrected: 13 December 2024

1. Introduction

Robots have been developed from performing repetitive tasks solely in industrial settings to becoming a part of our daily lives, thanks to advancements in software, sensors, and processors. Notably, recent breakthroughs in machine learning have enabled robots to adeptly perceive their surroundings and engage in natural language communication with humans [1]. Consequently, we now encounter robots operating in diverse environments such as city halls [2], museums [3–5], airports [6], and restaurants [7–9]. These robots offer interactive and intelligent assistance without relying on specific infrastructures as well as mere repetitive tasks.

However, most robots adopted in ordinary spaces have wheeled locomotion, which presents challenges when encountering obstacles like stairs or thresholds. Moreover, the design of manipulators is often characterized by limited capabilities, primarily encompassing basic gestures and actions.

Lately, studies have been conducted on home service robots designed as mobile manipulators to create practical automated mobile manipulation systems for home environments [1,10–12]. However, these investigations only focus on wheeled-type robots equipped with manipulators. Several studies proposed frameworks to conduct grasping tasks with a manipulator mounted on quadruped robot [13,14]. However, many of these frameworks did not include comprehensive tasks combining perception, navigation, locomotion, and manipulation.

In this context, we introduce the software framework to enable a quadruped robot to organize household objects to appropriate space in a domestic environment. Unlike previous works [13,14] that perform just simple mobile manipulation with quadruped robots, our research presents the method for delivering practical services through the utilization of quadruped robots.

Compared to other platforms that performed similar tasks, a quadruped robot used for this framework should also equipped with cameras, LiDAR, IMU, and a manipulator. Figure 1 shows the feature comparison of the robot model to apply to the framework and to Human Support Robots (HSRs) [15], which performed similar tasks in [1,10,11].

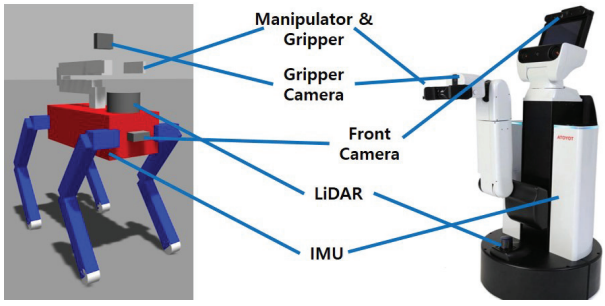


Figure 1. Feature comparison with a robot model for the framework and HSR.

The subsequent sections of this paper are structured as follows: Section 3 describes the overall system of this framework. In Section 4, the method for object detection and point cloud generation is detailed, involving a combination of machine learning techniques including YOLOv7 [16], K-Nearest Neighbor (KNN) [17], and Random Sample Consensus (RANSAC) [18], and the estimation of a grasp pose from a point cloud, accomplished through its conversion into a height map, is presented. Moving to Section 5, navigation strategies outlining how the robot plans its route to the designated area considering the positions of detected objects and LiDAR data are presented, and grasping methods, depending on the situation, are addressed. The mathematical analysis of manipulation and locomotion using Model Predictive Control (MPC) [19–22] are presented in Section 6. Section 7 showcases the experimental outcomes conducted within a simulation environment resembling RoboCup@Home 2021—virtual. Lastly, the paper concludes by discussing future works in Sections 8 and 9.

2. Related Work

Various service robots deployed in public places were presented in [4,6]. In [4], the "Lindsey" robot, stationed at the Lincoln Museum, successfully operated autonomously as a guide, providing informative tours to visitors. Despite its practical utility, the platform's lack of physical interaction capabilities limited its scope. A similar case is presented in [6], where the "Spencer" robot facilitated passenger assistance and guidance at Dutch KLM airports. However, this robot also lacks a manipulator for physical engagement. Addressing this limitation, refs. [1,10] introduced a modular software framework for home service robots equipped with manipulators. This comprehensive framework encompassed navigation, visual perception, manipulation, human–robot interaction, and natural language processing. The framework incorporated deep-learning-based perception packages, such as YOLOv3 and OpenPose, to perceive surroundings and combine them with manipulation or navigation tasks using ROS middleware. Depending on its detected data, the framework showed various manipulation skills implemented in robot competitions. This framework showcased promising results in RoboCup@Home 2020 and World Robot Summit 2020 Partner Robot Challenge (WRS 2020 PRC) league using the Toyota HSR [15]. It is worth noting that this system is primarily applicable to wheeled-type mobile manipulators.

Several studies have been conducted regarding quadruped robots equipped with manipulators. In [13], researchers detailed strategies that control a quadruped robot with a Whole Body Control (WBC) framework for arm-mounted quadruped robots. In this work, the author proposed two control modes: manipulation mode and loco-manipulation mode. In the manipulation mode, the author used Hierarchical Quadratic Programming (HQP) [23] to control the arm, legs, and base subject to the whole rigid-body dynamics of

the robot. In the loco-manipulation mode, the author controlled the arm with PD control, while the HQP controller controlled the base and the legs. It showed stable gaiting in complex terrains with an arm-mounted quadruped robot. However, this approach did not incorporate image or LiDAR data.

In [14], a comparable system was introduced. The framework was experimented with the quadruped robots equipped with a five Degree of Freedom (DOF) manipulator, front camera, and 3D LiDAR. Using Yolov5, it successfully detected an object to grasp in a 3D position. Additionally, it presented human following with LiDAR data. However, it had limitations in addressing the manipulation of complex-shaped objects like bowls and exclusively concentrated on object manipulation on the ground. Moreover, comprehensive experimental testing of the system's capabilities was lacking.

In [24], researchers outlined a methodology for grasping complex-shaped objects utilizing an anthropomorphic robotic hand affixed to a seven-DOF arm through imitation learning. By combining 3D reconstruction, KNN, and image recognition using a Residual Neural Network (ResNet), the author realized an imitation learning framework that learns how to grasp complex objects from humans. However, this system required a diverse dataset for learning to grasp, encompassing RGB images, point clouds, and trajectories.

Certain studies have explored methods to grasp detected objects without requiring additional learning [25,26]. In [25], researchers introduced a grasp pose estimation technique based on 3D point cloud analysis, employing Principal Component Analysis (PCA) [27] and RANSAC [18]. While this approach showed promising performance by focusing solely on point cloud contour lines, it was limited in its applicability to objects with complex shapes. Another study, outlined in [26], utilized a virtual gripper with a C-shape configuration to determine the grasp pose. This approach accommodates complex-shaped objects; however, due to the inherently random nature of the deduced grasping orientation, it demands a high-DOF manipulator to secure the object effectively

3. System Overview

Figure 2 shows the simulation model used in this work and the schematics of the framework designed to perform tidy-up tasks that require perception and mobile manipulation with a quadruped robot. To execute the main functions, the model has the form of a quadruped robot equipped with a front camera, gripper camera, LiDAR, and a low-DOF manipulator. The framework is combined with multiple modules interconnected through ROS [28] messages, which are divided into three blocks: perception, behavior control, and joint control. The approximate role of each block is as follows, and Table 1 shows the dimensions of the robot model used in the experiment.

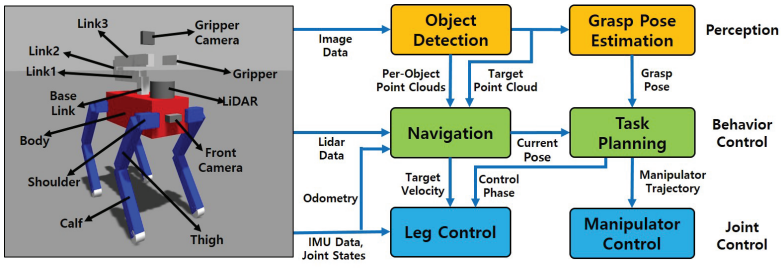


Figure 2. Overview of the system.

Table 1. Dimensions of the robot.

Body Length	Body Width	Thigh Length	Calf Length
0.419 m	0.2505 m	0.22 m	0.22 m

3.1. Perception

The perception block is the initial step in our research workflow, encompassing the object detection and grasp pose estimation modules. In the object detection module, we employ a machine learning-based algorithm to process image data, generating point clouds for each detected object. Subsequently, from these point clouds, we select the target point cloud for grasping and derive the grasp pose for the respective object in the grasp pose estimation module.

3.2. Behavior Control

The prior information required for joint control is derived in behavior control. By combining LiDAR data and odometry with per-object point cloud and target object, which is derived in the object detection module, the navigation module can generate the target velocity of Center of Mass (COM) and current pose on the map. The current pose and grasp pose are used to derive the control phase, which decides the robot's control state (e.g., walking or standing) and manipulation trajectory. The task planning module generates the manipulator trajectory when the current grasp pose is appropriate.

3.3. Joint Control

The joint control block performs actual roles in moving the robot. The leg control module employs the MPC-based method for precise and stable control. This method requires IMU data, joint states (e.g., position and velocity), and odometry. This module is rooted in [20], and we customize it to suit this research. On the other hand, the manipulator control module utilizes position control using the numerical solution of the inverse kinematics.

4. Perception

In order to detect each object in 3D space, we employ a combination of machine learning approaches, including Yolov7, KNN, and RANSAC. Initially, we select the target object from among the detected objects using these methods. Subsequently, we estimate the grasp pose of the chosen object by converting the point cloud into a height map.

4.1. Per-Object Point Cloud Generation

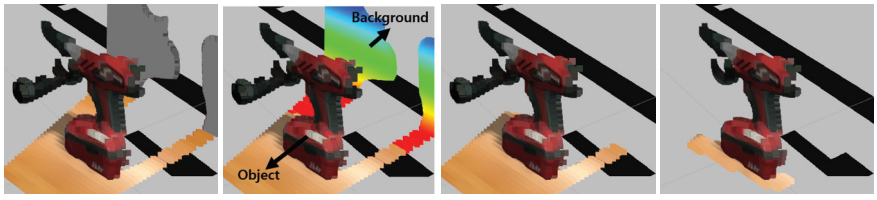
To obtain 3D information about the objects in determining which object to grasp and estimating its grasp pose, generating point clouds for each object emerges as a preliminary step. This endeavor follows the real-time detection of objects from 2D images. As illustrated in Figure 3a, we employ YOLOv7 [16], using Deep Learning methodologies to detect objects and outline their bounding boxes within RGB images in real-time. After object detection, we segment the corresponding positions in the depth image. By projecting this segmented data into 3D space via the intrinsic K matrix [29], point clouds for each object are derived, as shown in Figure 3b.

(**a**) (**b**)

Figure 3. Object detection: (**a**) Detecting objects with YOLOv7 and (**b**) Point clouds per object.

4.2. Filtering Outliers

Figure 4a reveals the presence of outliers causing distortions. To address this issue, we deploy two machine learning techniques: KNN [17] and RANSAC [18]. Initially, showcased in Figure 4b, we partition the point cloud into object and background segments using KNN [17]. However, distinguishing between the object and background through KNN [17] alone is challenging. To address this, considering the closer proximity of the object's centroid to the robot's camera, we exclude the background by eliminating the portion with a more distant centroid. This strategy yields a model devoid of background, showcased in Figure 4c. Subsequently, by leveraging plane-fitting RANSAC [18], the remaining outliers are filtered, aiding in the acquisition of plane normal vectors utilized for predicting grasp directions. The culmination of these steps yields refined 3D models free from outliers, as depicted in Figure 4d.

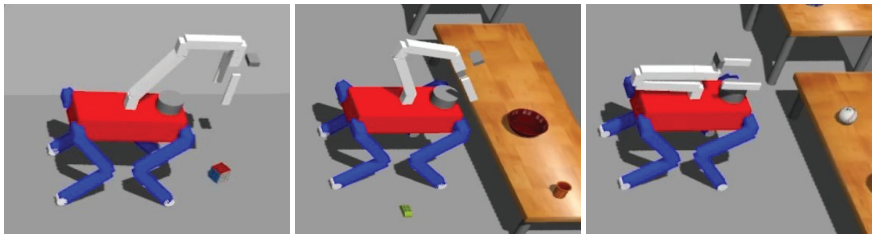


(**a**) (**b**) (**c**) (**d**)

Figure 4. Process of filtering outliers: (**a**) Original, (**b**) Part division, (**c**) Background removal, and (**d**) Filtering outliers.

4.3. Probing Direction Decision

Depending on the object's state, such as on the floor or the table, and properties, the robot should determine its probing direction to estimate a grasp pose of the object in the easy-to-grasp posture. The robot employs a gripper-mounted camera to probe the object vertically when the object is positioned on the floor, as exemplified in Figure 5a. However, for objects on the table, the probing direction requires prediction. This prediction entails adopting the posture depicted in Figure 5b for object assessment. Within this configuration, the robot employs the plane-fitting RANSAC [18] to compute the normal vector of the object's point cloud. If the z-coordinate of this normal vector surpasses a predetermined threshold, the robot concludes that vertical probing is ideal and proceeds to generate a corresponding height map. Conversely, horizontal probing is deemed more suitable if the z-coordinate falls below the threshold. In this case, the robot repositions its manipulator to the configuration shown in Figure 5c and adjusts its position to a grasp-appropriate point. Subsequently, with the manipulator reoriented, the robot employs its body-mounted front camera to create a height map for the object's horizontal probing assessment.



(**a**) (**b**) (**c**)

Figure 5. Setting probing posture in probing area: (**a**) Vertical-floor, (**b**) Vertical-table, and (**c**) Horizontal.

4.4. Height Map Creation

The construction of a height map, derived from the point cloud data, is executed through distinct coordinate configurations according to the robot's chosen probing direction. In instances where the robot decides on horizontal probing, the x and y coordinates of the height map are extracted from the point cloud's y and z coordinates, respectively. Conversely, for vertical probing, the x and y coordinates of the height map are derived from the point cloud's x and y coordinates, correspondingly. The height values assigned to the height map are drawn from the z-coordinates of the point cloud when probing vertically, while horizontal probing utilizes the x-coordinates for height value determination. A detailed explanation is given in Algorithm 1, and Figure 6 shows the result.

Algorithm 1 Height map creation

```
x_coords ← removeDuplicate(target_point_cloud.x)
y_coords ← removeDuplicate(target_point_cloud.y)
z_coords ← removeDuplicate(target_point_cloud.z)
y_unit ← y_coords.size / HEIGHT_MAP_SIZE
if probing_pose is horizontal then
    x_unit ← z_coords.size / HEIGHT_MAP_SIZE
    creteria_height ← max(target_point_cloud.x)
else
    x_unit ← x_coords.size / HEIGHT_MAP_SIZE
    creteria_height ← min(target_point_cloud.z)
end if
for i ← 1 to target_point_cloud.size do
    height_map_x ← rank(target_point_cloud.y[i], y_coords)
    if probing_pose is horizontal then
        point_height ← creteria_height – target_point_cloud.x[i]
        height_map_y ← rank(target_point_cloud.z[i], z_coords)
    else
        point_height ← target_point_cloud.z[i] – creteria_height
        height_map_y ← rank(target_point_cloud.x[i], x_coords)
    end if
    for j ← 1 to HEIGHT_MAP_SIZE do
        if height_map_x ≥ x_unit ∗ (j – 1) and height_map_x ≤ x_unit ∗ j then
            for k ← 1 to HEIGHT_MAP_SIZE do
                if height_map_y ≥ y_unit ∗ (k – 1) and height_map_y ≤ y_unit ∗ k then
                    height_map_num(j, k) ← height_map_num(j, k) + 1
                    height_map_sum(j, k) ← height_map_sum(j, k) + point_height
                end if
            end for
        end if
    end for
end for
height_map ← height_map_sum / height_map_num
```

4.5. Grasp Pose Prediction

The height map derived in Section 4.4 is used to predict grasp pose. From this map, we select grasp candidates and select grasp pose among them, considering contact pose inclination and distance to the center of the height map. Subsequently, we convert the grasp pose, initially represented in the height map, into the 3D space.

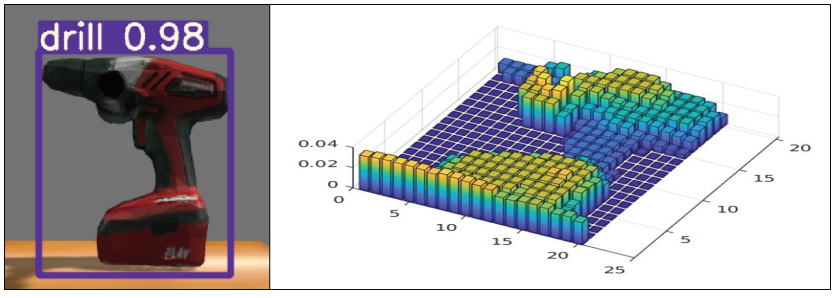


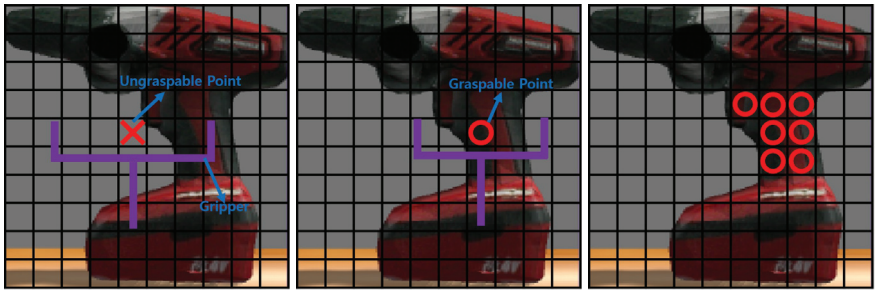
Figure 6. Height map.

4.5.1. Selecting Grasp Candidates

The primary step in the prediction process involves the selection of grasp candidates extracted from the height map. As illustrated in Figure 7, this procedure requires transforming the gripper to fit the height map and subsequently evaluating each element in conjunction with the gripper configuration to ascertain its graspability. A point is considered a graspable candidate when the height of the coordinate situated at the center of the gripper exceeds the height of the locations where the gripper's tips are positioned by a predefined margin. However, within this evaluation, if the slopes present within the gripper's region exhibit a gradient lower than a specified threshold, the coordinate is classified as ungraspable. Comprehensive details of this operational phase are presented in Algorithm 2.

Algorithm 2 Selecting grasp candidates

```
for i ← 1 to HEIGHT_MAP_SIZE do
    for j ← 1 to HEIGHT_MAP_SIZE do
        left_tip_pos ← i + gripper_width_half_height_map
        right_tip_pos ← i – gripper_width_half_height_map
        left_tip_diff ← height_map(i, j) – height_map(left_tip_pos, j)
        right_tip_diff ← height_map(i, j) – height_map(right_tip_pos, j)
        if left_tip_diff or right_tip_diff > GRASPABLE_HEIGHT then
            grasp_pos_found ← false
            for k ← right_tip_pos to i – 1 do
                right_height_slope ← height_map(k + 1, j) – height_map(k, j)
                if right_height_slope ≥ GRASPABLE_HEIGHT_VAR then
                    for l ← i + 1 to left_tip_pos do
                        left_height_slope ← height_map(l – 1, j) – height_map(l, j)
                        if right_height_slope ≥ GRASPABLE_HEIGHT_VAR then
                            grasp_pos_candidates.add({i, j})
                            right_tip_contact_poses.add({k, j})
                            left_tip_contact_poses.add({l, j})
                            grasp_pos_found ← true
                            break
                        end if
                    end for
                end if
                if grasp_pos_found then
                    break
                end if
            end for
        end if
    end for
end for
```



(**a**) (**b**) (**c**)

Figure 7. Selecting grasp candidates: (**a**) Ungraspable position, (**b**) Graspable position, and (**c**) Grasp candidates.

4.5.2. Getting Contact Pose Inclination

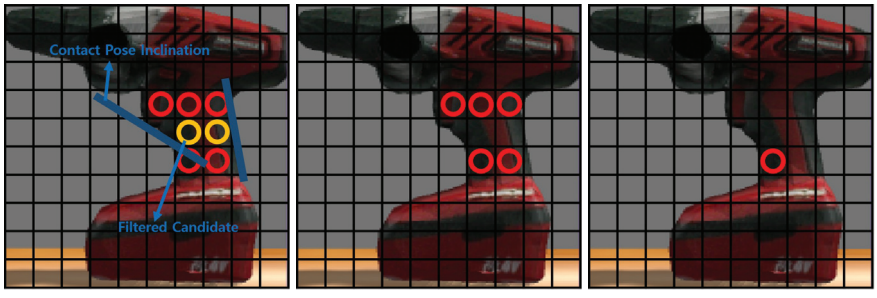
While candidates might meet the criteria outlined in Section 4.5.1, addressing potential slipping issues arising from unaccounted contact pose inclinations is essential. To address this concern, we engage neighboring contact coordinates around the present contact coordinate to ascertain inclinations. This involves determining slopes based on contact coordinates adjacent to the existing contact coordinate, thus enabling the derivation of contact pose inclinations. Algorithm 3 details the precise steps.

Algorithm 3 Getting contact pose inclination

```
for i ← 1 to HEIGHT_MAP_SIZE do
    for j ← 1 to HEIGHT_MAP_SIZE do
        vicinity_pose_x.empty()
        vicinity_pose_y.empty()
        if tip_contacted_pos.is_exist({i, j}) is true then
            for k ← −VICINE_RANGE to VICINITY_RANGE do
                for l ← −VICINE_RANGE to VICINITY_RANGE do
                    if tip_contacted_pos.is_exist({i + k, j + l}) is true then
                        vicinity_pose_x.add(i + k)
                        vicinity_pose_y.add(j + l)
                    end if
                end for
            end for
            vicinity_pose_x_diff ← vicinity_pose_x.max − vicinity_pose_x.min
            vicinity_pose_y_diff ← vicinity_pose_y.max − vicinity_pose_y.min
            tip_pose_inclination.add(atan2(vicinity_pose_y_diff, vicinity_pose_x_diff))
        end if
    end for
end for
```

4.5.3. Selecting Grasp Pose in Height Map

Following determining the contact pose inclinations, depicted in Figure 8, a subsequent filtering process is implemented to address candidates within low inclination regions. From the remaining candidates, the one closest to the center of the height map is selected as the prime candidate. In cases where multiple candidates share the same distance to the center, the selection prioritizes the candidate within the narrowest area.



(**a**) (**b**) (**c**)

Figure 8. Selecting grasp pose: (**a**) Getting contact inclination, (**b**) Filtering candidates, and (**c**) Selecting grasp pose.

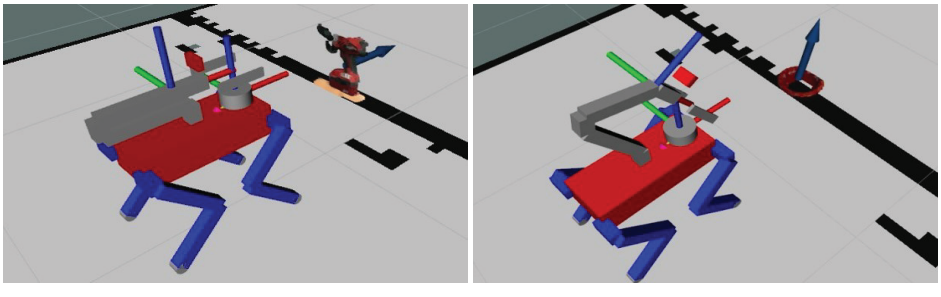
4.5.4. Grasp Pose Transition

Concluding the prediction process, the final step is translating the grasp pose determined within the height map to a comprehensive 3D pose, accomplished through Algorithm 4. This process effectively reverses the steps undertaken in Algorithm 1, utilizing derived variables such as *x_coords*, *y_coords*, and *z_coords* from the earlier algorithm.

The outcome of Algorithm 4 is showcased in Figure 9, where the position of the arrow symbolizes the grasp pose. At the same time, its orientation represents the derived grasp direction, facilitated by utilizing the normal vector from the plane-fitting RANSAC [18]. This step finalizes the prediction procedure, ensuring accurate grasp pose representation in three-dimensional space.

Algorithm 4 Grasp pose transition

if *probing_pose* is *horizontal* **then**
    $grasp\_pos\_3d.x \leftarrow creteria\_height - height\_map(grasp\_pos\_2d.x, grasp\_pos\_2d.y)$
else
    $grasp\_pos\_3d.z \leftarrow height\_map(grasp\_pos\_2d.x, grasp\_pos\_2d.y) + creteria\_height$
end if
$grasp\_pos\_idx\_horizontal \leftarrow grasp\_pos\_2d.y * y\_unit$
$grasp\_pos\_idx\_vertical \leftarrow grasp\_pos\_2d.x * x\_unit$
$grasp\_pos\_3d.y \leftarrow y\_coords[grasp\_pos\_idx\_horizontal]$
if *probing_pose* is *horizontal* **then**
    $grasp\_pos\_3d.z \leftarrow z\_coords[grasp\_pos\_idx\_vertical]$
else
    $grasp\_pos\_3d.x \leftarrow x\_coords[grasp\_pos\_idx\_vertical]$
end if



(**a**) (**b**)

Figure 9. Grasp pose in 3D point cloud: (**a**) Drill and (**b**) Bowl.

4.6. Rotation of an Object on The Floor

When dealing with objects situated on the floor, achieving an optimal grasp is facilitated when the object's orientation aligns with the gripper's wrist angle. To achieve this alignment, we leverage the line-fitting RANSAC [18] when probing objects on the floor. As illustrated in Figure 10, the direction of the object's normal vector corresponds to the desired gripper angle. To accommodate this alignment, we perform a z-axis rotation of the object's point cloud according to the normal vector, preceding the grasp pose prediction step. The final execution involves the robot grasping the object using the gripper positioned in alignment with the object's orientation, thus optimizing the grasping process for objects located on the floor.

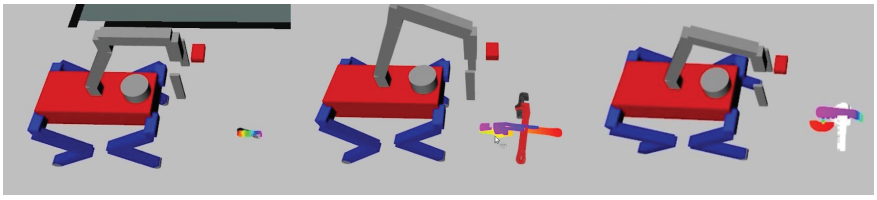


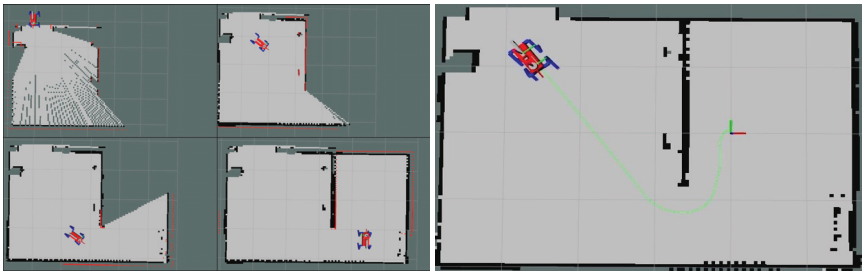
Figure 10. Rotation of an object on the floor.

5. Behavior Control

Before engaging in robot control at the joint level, managing and directing the robot's behavior is essential. Based on the detection information discussed in Section 4, the robot performs Navigation and decides grasping form.

5.1. SLAM

In preparation for organizing objects, the robot initiates its process by determining its position and comprehending its immediate environment. This initial phase involves the creation of a spatial map using ROS's SLAM package known as gmapping.Through manual guidance within the designated area, the robot creates a map using LiDAR data and odometry, as illustrated in Figure 11a. As the locomotion algorithm used in this framework generates less staggering in gaiting, additional compensations are not required.



(**a**) (**b**)

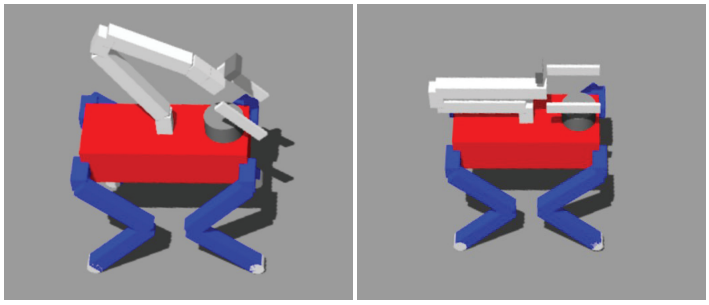
Figure 11. SLAM and navigation: (**a**) SLAM and (**b**) Navigation.

5.2. Navigation

For precise navigation to predetermined positions within the map established in Section 5.1, the robot's movement is facilitated by utilizing the ROS package associated with navigation, known as `amcl`. While this package has proven effective, it is employed primarily for transporting the robot to designated search or deposit zones due to limitations in accurately approaching goal poses. Figure 11b exemplifies the process involving this package.

5.3. Approaching

The robot's initial task involves identifying graspable objects. By employing a camera attached to the gripper, the robot scans the floor while maintaining the posture displayed in Figure 12a. Upon detecting object centroids within its body frame of reference, the robot adjusts its movement toward the nearest object. Yet, when the proximity to this object falls below a defined threshold, *TARGETING_DIST*, it is categorized as a *target_object*. Subsequently, the robot repositions itself to a *probing_area*, which ensures accessibility by the gripper, as illustrated in Figure 5. Conversely, the robot reconfigures its manipulator to resemble the stance depicted in Figure 12b in scenarios where no floor objects are detected. This alternative posture is employed for surveying objects on a table using the front camera, following a procedure analogous to that used for the floor. Algorithm 5 offers a comprehensive breakdown of this operational phase.



(**a**) (**b**)

Figure 12. Searching postures: (**a**) Searching floor object posture and (**b**) Searching table object posture.

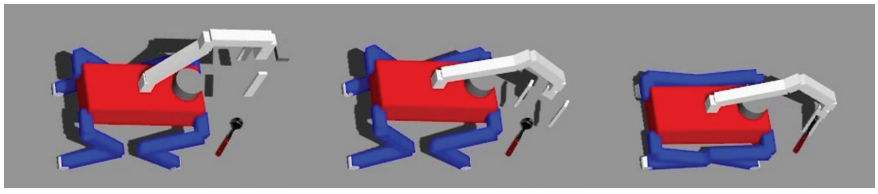
Algorithm 5 Approaching

```
searching_mode.manipulator ← floor_searching
searching_mode.camera ← gripper_camera
if isObjectExist(searching_state) is false then
    searching_mode.manipulator ← table_searching
    searching_mode.camera ← front_camera
    probing_area_type ← FLOOR_PROBING_AREA
else
    probing_area_type ← TABLE_PROBING_AREA
end if
while target_object.centroid not in probing_area do
    object_clouds ← searchObjects(searching_state)
    if target_object_id is null then
        closest_object ← getClosestObject(object_clouds)
        traceObject(closest_object.centroid)
        if closest_object.dist ≤ TARGETING_DIST then
            target_object_id ← closest_object.object_id
        end if
    else
        target_object ← getTragetObject(object_clouds, target_object_id)
        moveToProbingArea(target_object.centroid, probing_area_type)
    end if
end while
```

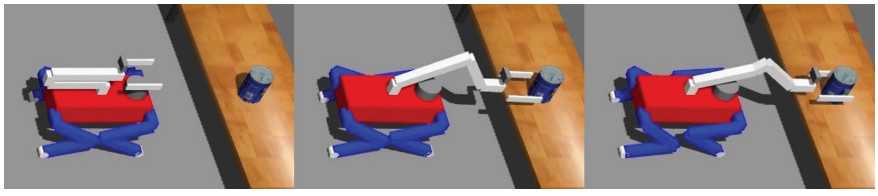
5.4. Grasping an Object

The act of grasping is executed with variations contingent upon the object's specific situation, as demonstrated in Figure 13. Despite these variations, the fundamental grasping

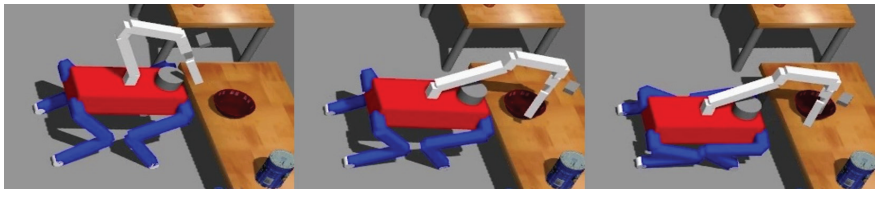
process can be categorized into three distinct stages: probing, transitioning to a grasp-ready pose, and actual grasping.



(**a**)



(**b**)



(**c**)

Figure 13. Grasping an object differently depending on the situation: (**a**) Grasping an object on the floor, (**b**) Grasping an object on the table horizontally, and (**c**) Grasping an object on the table vertically.

Upon reaching an area conducive to grasping, the robot initiates the probing stage, which adapts according to the specific scenario. After probing, the robot adjusts the positioning of its COM and manipulator to align the grasp pose within the object with the gripper's front. In the ensuing stage, when the grasp direction is vertical, the robot adjusts its COM along the z-axis by flexing its knee, effectively facilitating object grasping. Conversely, for horizontal grasp directions, the robot shifts its manipulator along the x-axis to the object's location, preventing any potential collision between the gripper and the object. However, when the object's distance exceeds the manipulator's operational range, the robot compensates by moving its COM along the x-axis. Algorithm 6 comprehensively describes this process.

Algorithm 6 Grasping an object

```
if grasp_ready_phase then
    moveCOM(–grasp_pos_3d.y)
    if grasp_orientation is vertical then
        gripper_pose.x ← grasp_pose_3d.x
        gripper_pose.z ← grasp_pose_3d.z + GRASP_READY_Z
        if centoid_grasp_pose_diff_dist ≥ ROLL_THRESHOLD then
            gripper_roll ← atan2(centoid_grasp_pose_diff.y, centoid_grasp_pose_diff.x)
        else
            gripper_roll ← line_fitting_RANSAC(filtered_point_cloud).normal
        end if
```

Algorithm 6 *Cont.*

```
    else
        gripper_pose.x ← GRASP_READY_X
        gripper_pose.z ← grasp_pose_3d.z
    end if
else if grasp_phase then
    if grasp_orientation is vertical then
        grapper_pos.z ← grasp_pos_3d.z + COM_MOVE_Z
        com_pose.z −= COM_MOVE_Z
    else
        if grasp_pos_3d.x ≤ MANIPULATOR_X_LIM then
            grapper_pos.x ← grasp_pos_3d.x
        else
            grapper_pos.x ← MANIPULATOR_X_LIM
            com_pose.x += (grasp_pos_3d.x − MANIPULATOR_X_LIM)
        end if
    end if
end if
```

6. Joint Control

To facilitate the execution of desired robot behaviors, control at the joint level should be performed. The robot's motion control comprises two essential components: Manipulator Control and Leg Control. The robot controls its manipulator with position control with a numerical solution of inverse kinematics and controls its leg with the MPC-based method. Table 2 shows the dimensions of the manipulator.

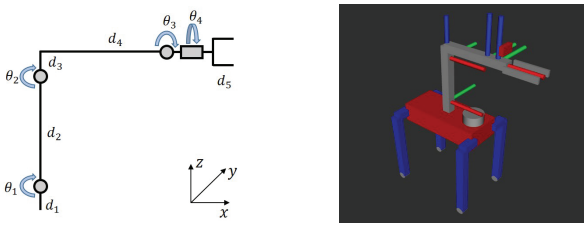
Table 2. Dimensions of the manipulator.

d_1	d_2	d_3	d_4	d_5
0.06 m	0.25 m	0.06 m	0.21 m	0.1 m

6.1. Manipulator Control

To ensure minimal impact on the robot's gaiting, the manipulator integrated onto the quadruped robot is designed to be lightweight. Achieving this objective involves employing a low-DOF manipulator, effectively reducing the weight of its actuators. As depicted in Figure 14, a four-DOF manipulator configuration has been adopted for object grasping. Utilizing these parameters, we can define $T_{tr}(x, y, z)$ as a translational transform and R_x, R_y, and R_z as rotational transforms around the x, y, and z axes, respectively. Consequently, the solution for the arm's forward kinematics can be deduced as follows:

$$\mathbf{T_{manipulator}} = T_{tr}(0,0,d_1)R_y(\theta_1)T_{tr}(0,0,d_2)R_y(\theta_2)T_{tr}(d_4,0,d_3)R_y(\theta_3)R_x(\theta_4)T_{tr}(d_5,0,0) \quad (1)$$



(**a**) (**b**)

Figure 14. Manipulator configuration and model: (**a**) Arm configuration and (**b**) Robot model.

Inverse Kinematics

To obtain the inverse kinematics solution for a target transform $\mathbf{T}_{\mathbf{manipulator}}$, we first calculate the relative wrist position $(x', 0, z')$.

$$\mathbf{T}'_{\mathbf{manipulator}} = T_{tr}(0, 0, -d_1)\mathbf{T}_{\mathbf{manipulator}}T_{tr}(-d_5, 0, 0) \tag{2}$$

$$\begin{bmatrix} x' & 0 & z' & 1 \end{bmatrix}^T = \mathbf{T}'_{\mathbf{manipulator}}\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T \tag{3}$$

Using the components in Figure 15, we can obtain θ_1 and θ_2 with following equation.

$$d_6 = \sqrt{x'^2 + z'^2} \tag{4}$$

$$d_7 = \sqrt{d_3^2 + d_4^2} \tag{5}$$

$$\theta_1 = \frac{\pi}{2} - a\cos\left(\frac{d_1^2 + d_6^2 - d_7^2}{2d_1d_6}\right) - a\tan 2(z', x') \tag{6}$$

$$\theta_2 = \pi - a\cos\left(\frac{d_1^2 + d_7^2 - d_6^2}{2d_1d_7}\right) - a\tan 2(d_4, d_3) \tag{7}$$

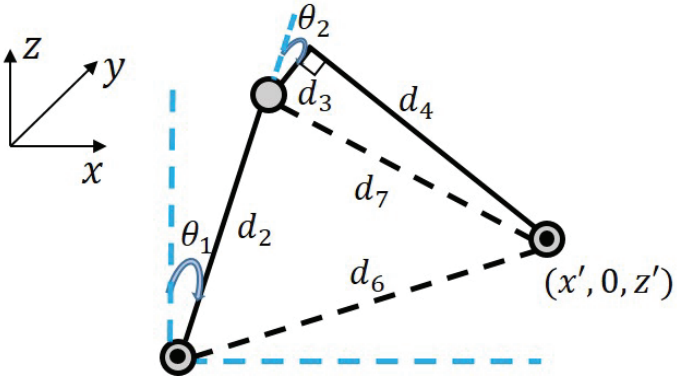


Figure 15. Arm configuration without the gripper and base.

As all joints without θ_4 are moved in the y-axis, the gripper's roll is the same with θ_4. Now that we know θ_1, θ_2, and θ_4, we can obtain θ_3 with following equation:

$$R_y(\theta_3) = T_{tr}(-d_4, 0, -d_3)R_y(-\theta_2)T_{tr}(0, 0, -d_2)R_y(-\theta_1)\mathbf{T}'_{\mathbf{manipulator}}R_x(-\theta_4) \tag{8}$$

$$\theta_3 = a\cos(R_y(\theta_3)_{11}) \tag{9}$$

6.2. Leg Control

In Leg Control, we adopted a framework rooted in MPC, introduced in [20]. Considering its current state and desired pose, this framework derives the appropriate Ground Reaction Force (GRF) with MPC. To adopt this framework as the Leg Control module, we adjusted several components to fit this work.

6.2.1. COM Controller

As the Navigation module returns the target velocity, we adjust the desired pose with this value. When the target velocity is returned, the desired pose is adjusted as follows.

$$\boldsymbol{p}_{target}(t + \Delta t) = \boldsymbol{p}_{target}(t) + \boldsymbol{v}_{target}\Delta t \tag{10}$$

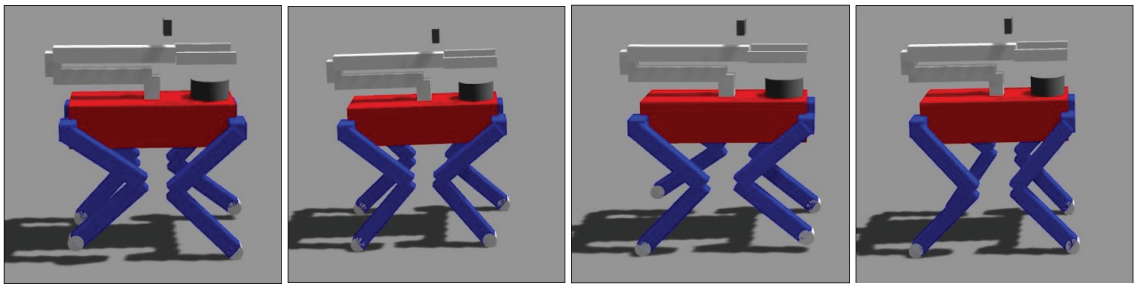
$$\Omega_{target} = \begin{bmatrix} 0 & -\omega_{target\_x} & -\omega_{target\_y} & -\omega_{target\_z} \\ \omega_{target\_x} & 0 & \omega_{target\_z} & -\omega_{target\_z} \\ \omega_{target\_y} & -\omega_{target\_z} & 0 & \omega_{target\_x} \\ \omega_{target\_z} & \omega_{target\_y} & -\omega_{target\_x} & 0 \end{bmatrix} \tag{11}$$

$$\boldsymbol{q}_{target}(t + \Delta t) = (\boldsymbol{I} + \frac{1}{2}\Omega_{target}\Delta t)\boldsymbol{q}_{target}(t) \tag{12}$$

Subsequently, we set *VEL_LIMIT* in FSM the same as an absolute value of the target velocity.

6.2.2. Gait

In navigation, we adopt trotting as a gaiting form. As shown in Figure 16, the trotting phase is divided into four phases: swing_FLRR, stance_FLRR, swing_FRRL, and stance_FRRL. The desired GRF in swing phases (swing_FLRR and swing_FRRL) is half of the body mass, and the desired GRF in stance phases (stance_FLRR and stance_FRRL) is a quarter of the body mass. Since the leg mass is less than 10% of the robot's total mass, the legs' inertia effect could be neglected.



(**a**) (**b**) (**c**) (**d**)

Figure 16. Gaiting with four phases: (**a**) swing_FLRR, (**b**) stance_FLRR, (**c**) swing_FRRL, and (**d**) stance_FRRL.

6.2.3. Parameters

The parameters used in Leg Control should be adjusted to fit the environment of this work. Table 3 shows the parameters used in this work. In this table, $\boldsymbol{R_u}$, $\boldsymbol{Q}$, N_{hor}, T_{sw}, T_{st}, and T_{pred} represent input weight matrix, pose weight matrix, expectation horizon, swing foot time, stance foot time, and prediction time, respectively.

Table 3. Parameters.

Parameter	Value
$\boldsymbol{R_u}$	[0.1 0 0; 0 0.1 0; 0 0 0.1]
$\boldsymbol{Q_p}$	[100,000 0 0; 0 150,000 0; 0 0 100,000]
$\boldsymbol{Q_{\dot{p}}}$	[100 0 0; 0 100 0; 0 0 100]
$\boldsymbol{Q_R}$	[5000 0 0; 0 5000 0; 0 0 5000]
$\boldsymbol{Q_\omega}$	[2 0 0; 0 4 0; 0 0 3]
N_{hor}	6
T_{sw}	0.3
T_{st}	0.1
T_{pred}	0.03

Note: T_{sw}, T_{st}, and T_{pred} have the unit [s].

6.2.4. Finite State Machine

This segment defines the robot's reference state utilized in the MPC framework. Under normal circumstances, we presume that the robot's initial posture and velocity mirror the current state, aiming to refine its velocity by controlling acceleration to achieve the intended pose. Nevertheless, when the current velocity's trajectory diverges from the desired pose, we adapt the planned velocity to align with the desired pose's direction, bypassing the current state's influence. Algorithm 7 provides an intricate breakdown of this process, incorporating parameters such as *ADJUST_VEL_THRESHOLD*, *FEED_BACK_VEL*, and *PLANNED_ACC*, all of which are denoted as positive numerical values.

Algorithm 7 Finite state machine

now_planned_pose ← *current_pose*
now_planned_vel ← *current_vel*
for $i \leftarrow 1$ to *EXPECTATION_HORIZON* **do**
    pose_diff ← *desired_pose* – *now_planned_pose*
    if *pose_diff.abs* < *ADJUST_VEL_THRESHOLD* **then**
        now_planned_vel ← 0
        now_planned_pose ← *desired_pose*
    else
        if *pose_diff* ∗ *now_planned_vel* < 0 **then**
            if *now_planned_vel* < 0 **then**
                now_planned_vel ← *FEED_BACK_VEL*
            else
                now_planned_vel ← –*FEED_BACK_VEL*
            end if
        else
            if $\frac{now\_planned\_vel^2}{2ACCEL} <$ *pose_diff.abs* **then**
                now_planned_vel –= *PLANNED_ACC* ∗ T_{pred} ∗ *pose_diff.sign*
            else if *now_planned_vel.abs* ≤ *VEL_LIMIT* **then**
                now_planned_vel += *PLANNED_ACC* ∗ T_{pred} ∗ *pose_diff.sign*
            end if
        end if
        now_planned_pose += *now_planned_vel* ∗ T_{pred}
    end if
    planned_vel[i] ← *now_planned_vel*
    planned_pose[i] ← *now_planned_pose*
end for

6.2.5. Swing Foot Trajectory

Given the dynamic nature of the desired foot placement position, varying based on the specific leg position (front, rear, left, or right), we employ Algorithm 8 to establish the swing foot trajectory.

Algorithm 8 Swing foot trajectory

TIME_CONST ← $\frac{T_{sw}}{2}$ + *EXTRA_DIST*
if is_front **then**
    desired_swing_foot_pos.x ← *BODY_LENGTH_HALF*
    foot_pos_des.x ← (*now_lin_vel.x* – *HIP_JOINT_Y* ∗ ω_z) ∗ *TIME_CONST*
    foot_pos_des.y ← (*now_lin_vel.y* + *SHOULDER_LENGTH* ∗ ω_z) ∗ *TIME_CONST*
else
    desired_swing_foot_pos.x ← –*BODY_LENGTH_HALF*
    foot_pos_des.x ← (*now_lin_vel.x* + *HIP_JOINT_Y* ∗ ω_z) ∗ *TIME_CONST*
    foot_pos_des.y ← (*now_lin_vel.y* – *SHOULDER_LENGTH* ∗ ω_z) ∗ *TIME_CONST*

Algorithm 8 *Cont.*

end if
if is_left **then**
    $desired\_swing\_foot\_pos.y \leftarrow HIP\_JOINT\_Y$
else
    $desired\_swing\_foot\_pos.y \leftarrow -HIP\_JOINT\_Y$
end if
$xy\_pos\_traj\_elem \leftarrow 0.5(1 - cos(\frac{\pi}{T_{sw}}t))$
$desired\_swing\_foot\_pos.x \mathrel{+}= foot\_pos\_des.x * xy\_pos\_traj\_elem$
$desired\_swing\_foot\_pos.y \mathrel{+}= foot\_pos\_des.y * xy\_pos\_traj\_elem$
$desired\_swing\_foot\_pos.z = TROT\_FOOT\_HEIGHT * sin(\frac{\pi}{T_{sw}}t)$

7. Experiment

The experimental evaluation was conducted using Gazebo [30], an open-source 3D robotic simulator integrated within the ROS framework [28]. However, we found a slipping problem when grasping an object with the gripper, so we adopted `gazebo_grasp_plugin` to solve it. The experimental setup, as illustrated in Figure 17, closely emulated the configuration resembling the Robocup@Home 2021-virtual league environment. As is customary in the competition, shown in Figure 18, this experiment encompassed object classification tasks. The system's speed and accuracy were assessed by placing objects in predefined positions. To run the simulation and YOLOv7 simultaneously, we used the desktop equipped with an AMD Ryzen 7 5800X, 32 GB of RAM, and an NVIDIA GeForce RTX 3090.

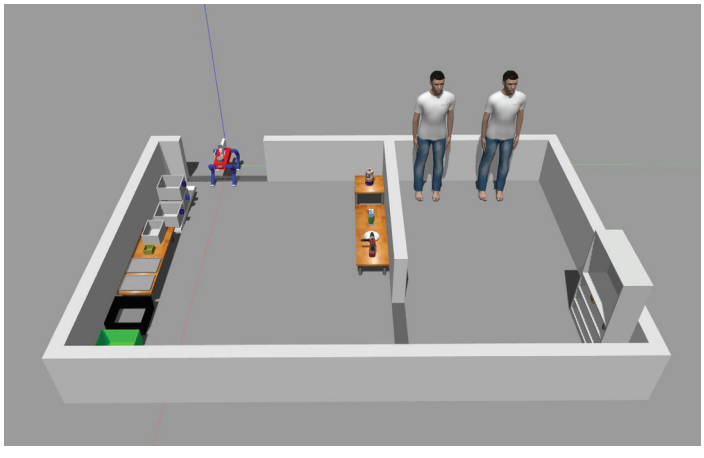
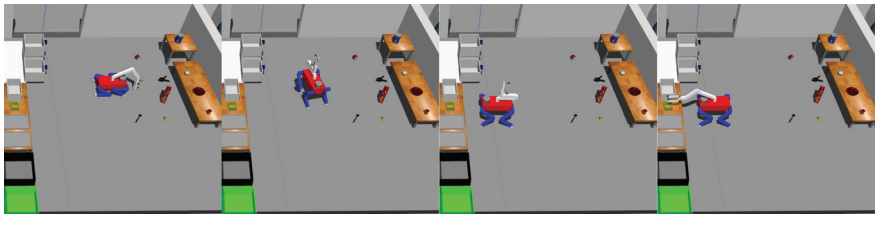
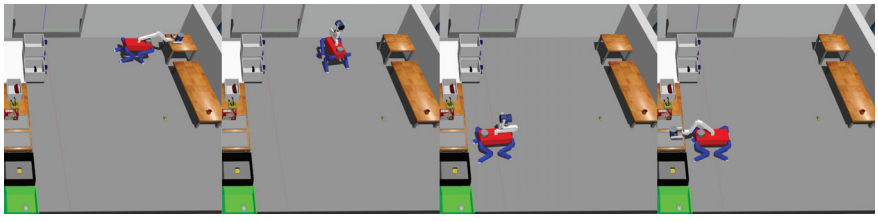


Figure 17. Simulation model and environment.



(**a**)

Figure 18. *Cont.*



(**b**)

Figure 18. Object classification tasks: (**a**) Classifying an object on the floor and (**b**) Classifying an object on the table.

7.1. Objects

A set of eighteen distinct objects was utilized during the experimental phase, as illustrated in Figure 19. These objects include a baseball, bowl, brick, clamp, coffee can, Rubik's cube, cup, driver, Lego, marker, padlock, cracker, spoon, sugar, tennis ball, tomato soup, and toy gun. These objects are drawn from the YCB object dataset, an official selection used in the Robocup@Home. Throughout the experiment, the objects' configurations were modified within the environment.

Figure 19. Objects used in the experiment.

7.2. Tasks

The experiment was structured around two main tasks: opening drawers and classifying objects. As depicted in Figure 20, the drawers were positioned in three distinct configurations: left, right, and top. At the outset of the experiment, the initial step involved manipulating the manipulator to open these drawers. Once the drawer-opening task was completed, the robot proceeded to the object classification phase.

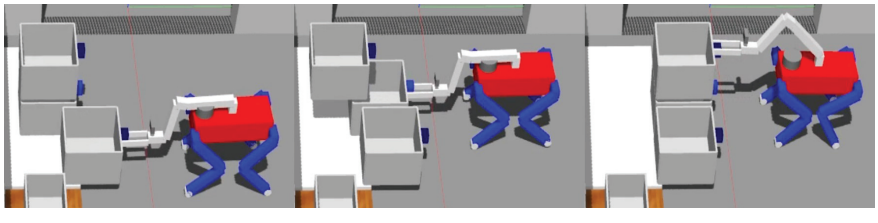
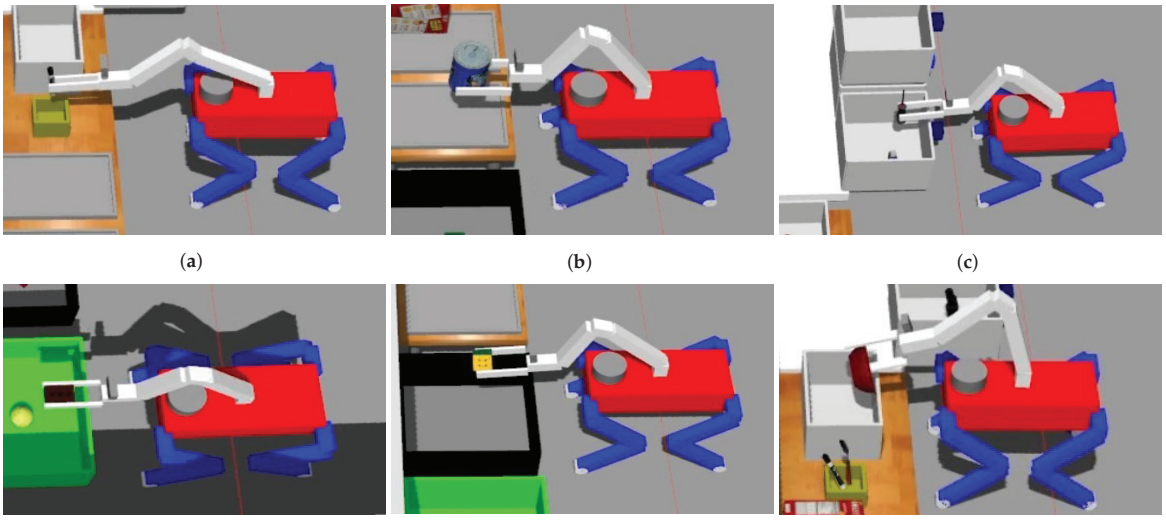


Figure 20. Opening drawer.

Object classification was executed through the deposition of objects into designated areas corresponding to specific categories. These categories encompassed orientation-based items (e.g., marker, spoon), food items (e.g., coffee can, sugar, tomato soup, and cracker), tools (e.g., driver, clamp, and padlock), shape items (e.g., baseball, tennis ball, cup, and brick), task items (e.g., Rubik's cube, toy gun, and Lego), and kitchen items (e.g., bowl, mug). The classification process was facilitated by arranging objects in the appropriate area corresponding to their respective categories. This classification was performed within an area divided into six distinct sections, as illustrated in Figure 21: pencil case (orientation-based items), tray (foods), drawer (tools), green box (shape items), black box (task items), and small box (kitchen items).

For a comprehensive visual representation of the experiments, all corresponding video recordings can be accessed at the link https://www.youtube.com/playlist?list=PLB1pUAsYGpRGpUhJ0qVN3_Y0EIwi5TMcz, (accessed on 1 October 2023).



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 21. Object classification sections: (**a**) Pencil case, (**b**) Tray, (**c**) Drawer, (**d**) Green box, (**e**) Black box, and (**f**) Small box.

7.3. Results

The experiment was repeated across five distinct environments, each involving ten objects from the selection shown in Figure 19. The outcomes of these trials are summarized in Table 4, providing details such as the number of successful attempts, the number of failures, the success rate, and the duration taken for each test. However, as the opening of the drawers succeeded in all experiments, it is not described in the table. Notably, the average success rate across all trials amounted to 96%. Moreover, the success rates for all environments consistently exceeded 80%, underscoring the system's robust performance across varying contexts. Comparatively, our system exhibited longer task execution times in certain scenarios, such as turning in place, walking sideways, and setting a grasp-ready pose, when compared to a wheeled robot. In particular, our platform required more than 3 min to complete the tasks with six fewer objects than [10], which utilized a wheeled robot (HSR) for a similar experiment.

Table 4. Results.

Trial	Objects	Success Num	Failure Num	Success Rate	Time
1	Rubik's cube, clamp, marker, cracker, spoon, Lego, coffee can, baseball, bowl, cup	10	0	100%	19:07
2	tennis ball, bowl, driver, padlock, spoon, sugar, tomato soup, toy gun, mug, brick	8	2	80%	18:55
3	tennis ball, mug, driver, toy gun, padlock, coffee can, Rubik's cube, tomato soup, bowl, brick	10	0	100%	20:48
4	tennis ball, bowl, padlock, Lego, marker, cracker, brick, mug, tomato soup, Rubik's cube	10	0	100%	19:59
5	marker, mug, tomato soup, Rubik's cube, clamp, driver, cup, bowl, coffee can, baseball	10	0	100%	18:10
Average		9.6	0.4	96%	19:24

8. Discussion

In this work, we proposed a mobile manipulation framework to organize household objects with quadruped robots. As described in Section 7, the model used in the experiment shows high stability and accuracy in locomotion and manipulation. Additionally, when estimating an object's grasp pose by combining machine learning algorithms, it selected an appropriate grasp pose in real time for the objects used in the experiment, even for challenging objects such as bowls, toy guns, and cups. However, when comparing its spending time with previous experiments using wheeled robots [10], the system's operational speed is relatively time-consuming.

Although this work successively reaches the goal of developing a framework to organize household objects with a quadruped robot, it is essential to acknowledge that these accomplishments were obtained exclusively within a simulation environment without tasks executed on uneven terrains, a significant advantage inherent to quadruped robots. Furthermore, the experiment required only mobile manipulation skills without Human–Robot Interaction (HRI) required for the robots used in ordinary places.

To address these limitations, in the following works, we plan to conduct experiments in real-world environments using physical hardware, including scenarios with uneven terrains. Simultaneously, we will committed to optimizing the framework to reduce task completion times and expanding its capabilities to include HRI functionalities.

9. Conclusions

This study introduces a comprehensive robotic framework that effectively performs household tasks through the integration of a quadruped robot equipped with perception, navigation, manipulation, and body control. The system's reliability is underscored by a successful experiment that attests to its high accuracy. However, this research was confined to simulation-based experiments, and task execution times were relatively extended.

In our future work, we plan to transition from simulation-based experiments to real-world experimentation employing an actual quadruped robot PADWQ [31], shown in Figure 22, developed at the Pusan National University. Additionally, we will also expand its functionality, such as Natural Language Processing (NLP), pose estimation, human tracking, etc., to encompass a wider array of general-purpose tasks, including HRI. Furthermore, our ongoing efforts will focus on scenarios that involve various terrains, including stairs or thresholds, while concurrently working to reduce task completion times.

Figure 22. Physical hardware platform for future work.

Author Contributions: Conceptualization, J.-B.Y. and S.-J.Y.; methodology, J.-B.Y.; software, J.-B.Y.; validation, J.-B.Y.; formal analysis, J.-B.Y.; investigation, J.-B.Y.; resources, S.-J.Y.; data curation, J.-B.Y.; writing—original draft preparation, J.-B.Y.; writing—review and editing, S.N., M.-s.J. and S.-J.Y.; visualization, J.-B.Y.; supervision, S.-J.Y.; project administration, S.-J.Y.; funding acquisition, S.-J.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by a 2-Year Research Grant of Pusan National University.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are unavailable due to privacy.

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

Abbreviations

The following abbreviations are used in this manuscript:

DOF	Degree of Freedom
KNN	K-Nearest Neighbor
RANSAC	Random Sample Consensus
HSR	Human Support Robot
HQP	Hierarchical Quadratic Programming
WBC	Whole Body Control
PCA	Principal Component Analysis
COM	Center of Mass
MPC	Model Predictive Control
GRF	Ground Reaction Force
NLP	Natural Language Processing
HRI	Human–Robot Interaction

References

1. Yi, J.B.; Kang, T.; Song, D.; Yi, S.J. Unified Software Platform for Intelligent Home Service Robots. *Appl. Sci.* **2020**, *10*, 5874. [CrossRef]
2. Hansen, S.T.; Hansen, K.D. *Public Relation Robots—An Overview*; Association for Computing Machinery: New York, NY, USA, 2020; pp. 284–286. [CrossRef]
3. Daczo, L.D.; Kalova, L.; Bonita, K.L.F.; Lopez, M.D.; Rehm, M. *Interaction Initiation with a Museum Guide Robot—From the Lab into the Field*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 438–447. [CrossRef]
4. Duchetto, F.D.; Baxter, P.; Hanheide, M. Lindsey the Tour Guide Robot—Usage Patterns in a Museum Long-Term Deployment. In Proceedings of the 2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), New Delhi, India, 14–18 October 2019; pp. 1–8. [CrossRef]
5. Okano, S.; Matsuhira, N.; Shimokawara, E.; Yamaguchi, T.; Narita, M. Employing Robots in a Museum Environment: Design and Implementation of Collaborative Robot Network. In Proceedings of the 2019 16th International Conference on Ubiquitous Robots (UR), Jeju, Republic of Korea, 24–27 June 2019; pp. 224–227. [CrossRef]
6. Triebel, R.; Arras, K.; Alami, R.; Beyer, L.; Breuers, S.; Chatila, R.; Chetouani, M.; Cremers, D.; Evers, V.; Fiore, M.; et al. *SPENCER: A Socially Aware Service Robot for Passenger Guidance and Help in Busy Airports*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 113, pp. 607–622. [CrossRef]
7. Langedijk, R.M.; Odabasi, C.; Fischer, K.; Graf, B. Studying Drink-Serving Service Robots in the Real World. In Proceedings of the 2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), Naples, Italy, 31 August–4 September 2020; pp. 788–793. [CrossRef]
8. Naik, L.; Palinko, O.; Kollakidou, A.; Bodenhagen, L.; Krüger, N. An interactive drink serving social robot: Initial System Implementation. In Proceedings of the IROS 2020 Workshop: Robots for Health and Elderly Care: An Honest Discourse on the Gap Between Research and Practical Application, Virtual, 29 October 2020.
9. Chen, C.S.; Lin, C.J.; Lai, C.C. Non-Contact Service Robot Development in Fast-Food Restaurants. *IEEE Access* **2022**, *10*, 31466–31479. [CrossRef]
10. Kang, T.; Song, D.; Yi, J.B.; Kim, J.; Lee, C.Y.; Yoo, Y.; Kim, M.; Jo, H.J.; Zhang, B.T.; Song, J.; et al. Team Tidyboy at the WRS 2020: A modular software framework for home service robots. *Adv. Robot.* **2022**, *36*, 836–849. [CrossRef]
11. Yi, J.B.; Yi, S.J. Mobile Manipulation for the HSR Intelligent Home Service Robot. In Proceedings of the 2019 16th International Conference on Ubiquitous Robots (UR), Jeju, Republic of Korea, 24–27 June 2019; pp. 169–173. [CrossRef]
12. Kang, T.; Kim, J.; Song, D.; Kim, T.; Yi, S.J. Design and Control of a Service Robot with Modular Cargo Spaces. In Proceedings of the 2021 18th International Conference on Ubiquitous Robots (UR), Gangneung, Republic of Korea, 12–14 July 2021; pp. 595–600. [CrossRef]
13. Xin, G.; Zeng, F.; Qin, K. Loco-Manipulation Control for Arm-Mounted Quadruped Robots: Dynamic and Kinematic Strategies. *Machines* **2022**, *10*, 719. [CrossRef]
14. Guo, J.; Chai, H.; Li, Y.; Zhang, Q.; Wang, Z.; Zhang, J.; Zhang, Q.; Zhao, H. Research on the Autonomous System of the Quadruped Robot with a Manipulator to Realize Leader-following, Object Recognition, Navigation and Operation. *IET Cyber-Syst. Robot.* **2022**, *4*, 376–388. [CrossRef]
15. Yamamoto, T.; Terada, K.; Ochiai, A.; Saito, F.; Asahara, Y.; Murase, K. Development of Human Support Robot as the research platform of a domestic mobile manipulator. *ROBOMECH J.* **2019**, *6*, 1–15. [CrossRef]
16. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv* **2022**, arXiv:2207.02696.
17. Jin, X.; Han, J. K-Means Clustering. In *Encyclopedia of Machine Learning*; Sammut, C., Webb, G.I., Eds.; Springer: Boston, MA, USA, 2010; pp. 563–564. [CrossRef]
18. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
19. García, C.E.; Prett, D.M.; Morari, M. Model predictive control: Theory and practice—A survey. *Automatica* **1989**, *25*, 335–348. [CrossRef]
20. Ding, Y.; Pandala, A.; Li, C.; Shin, Y.H.; Park, H.W. Representation-Free Model Predictive Control for Dynamic Motions in Quadrupeds. *IEEE Trans. Robot.* **2020**, *37*, 1154–1171. [CrossRef]
21. Di Carlo, J.; Wensing, P.M.; Katz, B.; Bledt, G.; Kim, S. Dynamic Locomotion in the MIT Cheetah 3 Through Convex Model-Predictive Control. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Detroit, MI, USA, 1–5 October 2018; pp. 1–9. [CrossRef]
22. Ding, Y.; Pandala, A.; Park, H.W. Real-time Model Predictive Control for Versatile Dynamic Motions in Quadrupedal Robots. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 8484–8490. [CrossRef]
23. Escande, A.; Mansard, N.; Wieber, P.B. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *Int. J. Robot. Res.* **2014**, *33*, 1006–1028. [CrossRef]
24. Yi, J.B.; Kim, J.; Kang, T.; Song, D.; Park, J.; Yi, S.J. Anthropomorphic Grasping of Complex-Shaped Objects Using Imitation Learning. *Appl. Sci.* **2022**, *12*, 12861. [CrossRef]

25. Lei, Q.; Chen, G.Y.; Wisse, M. Fast grasping of unknown objects using principal component analysis. *AIP Adv.* **2017**, *7*, 095126. [CrossRef]
26. Lei, Q.; Chen, G.Y.; Meijer, J.; Wisse, M. A novel algorithm for fast grasping of unknown objects using C-shape configuration. *AIP Adv.* **2018**, *8*, 025006. [CrossRef]
27. Mishra, S.; Sarkar, U.; Taraphder, S.; Datta, S.; Swain, D.; Saikhom, R.; Panda, S.; Laishram, M. Principal Component Analysis. *Int. J. Livest. Res.* **2017**, *7*, 60–78. [CrossRef]
28. Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A. ROS: An open-source Robot Operating System. In Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan, 12–17 May 2009; Volume 3.
29. Lin, K.Y.; Tseng, Y.H.; Chiang, K.W. Interpretation and Transformation of Intrinsic Camera Parameters Used in Photogrammetry and Computer Vision. *Sensors* **2022**, *22*, 9602. [CrossRef] [PubMed]
30. 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 (IROS), Sendai, Japan, 28 September–2 October 2004; Volume 3, pp. 2149–2154. [CrossRef]
31. Kim, J.; Kang, T.; Song, D.; Yi, S.J. PAWDQ: A 3D Printed, Open Source, Low Cost Dynamic Quadruped. In Proceedings of the 2021 18th International Conference on Ubiquitous Robots (UR), Gangneung, Republic of Korea, 12–14 July 2021; pp. 217–222. [CrossRef]

sensors

MDPI

Article

Design and Implementation of an Integrated Control System for Omnidirectional Mobile Robots in Industrial Logistics

Ahmed Neaz, Sunyeop Lee and Kanghyun Nam *

School of Mechanical Engineering, Yeungnam University, 280 Daehak-Ro, Gyeongsan 38541, Republic of Korea
* Correspondence: khnam@yu.ac.kr; Tel.: +82-53-810-2455

Abstract: The integration of intelligent robots in industrial production processes has the potential to significantly enhance efficiency and reduce human adversity. However, for such robots to effectively operate within human environments, it is critical that they possess an adequate understanding of their surroundings and are able to navigate through narrow aisles while avoiding both stationary and moving obstacles. In this research study, an omnidirectional automotive mobile robot has been designed for the purpose of performing industrial logistics tasks within heavy traffic and dynamic environments. A control system has been developed, which incorporates both high-level and low-level algorithms, and a graphical interface has been introduced for each control system. A highly efficient micro-controller, namely myRIO, has been utilized as the low-level computer to control the motors with an appropriate level of accuracy and robustness. Additionally, a Raspberry Pi 4, in conjunction with a remote PC, has been utilized for high-level decision making, such as mapping the experimental environment, path planning, and localization, through the utilization of multiple Lidar sensors, IMU, and odometry data generated by wheel encoders. In terms of software programming, LabVIEW has been employed for the low-level computer, and the Robot Operating System (ROS) has been utilized for the design of the higher-level software architecture. The proposed techniques discussed in this paper provide a solution for the development of medium- and large-category omnidirectional mobile robots with autonomous navigation and mapping capabilities.

Keywords: ROS and LabVIEW interaction 1; autonomous robot 2; navigation with ROS 3; control design with ROS 4; SLAM 5; navigation with ROS; omnidirectional mobile robot 6; integrated control system 7; industrial logistic robots 8

Citation: Neaz, A.; Lee, S.; Nam, K. Design and Implementation of an Integrated Control System for Omnidirectional Mobile Robots in Industrial Logistics. *Sensors* **2023**, *23*, 3184. https://doi.org/10.3390/s23063184

Academic Editors: Yuanlong Xie, Shiqi Zheng and Zhaozheng Hu

Received: 28 February 2023
Revised: 14 March 2023
Accepted: 14 March 2023
Published: 16 March 2023

1. Introduction

The COVID-19 pandemic has presented the global community with a unique challenge, and the scientific community has been working diligently to protect human health and maintain societal and industrial progress. The field of robotics has played a crucial role in this context. The utilization of different types of robots has been a highly researched topic in the wake of the pandemic. In fact, a survey [1] conducted in 2020 found that over 3500 papers were published on the topic of robots in contagion scenarios. Furthermore, the most significant research keywords, based on 280 publications, were mapped, with "autonomous robot" being among the top keywords. During the pandemic, the world has witnessed the successful deployment of robotic nurses [2] in Hong Kong, delivery robots in the United States, and working robots in Japan and Korea. Additionally, a study published in 2020 [3] indicates that since the onset of the COVID-19 pandemic, consumers are willing to pay an extra 61.28% for robot delivery.

The widespread adoption of robots has broadened the spectrum of human–robot collaboration, leading to an improvement in task accuracy and proximity to human employees. Among the various types of robots, mobile robots have gained significant attention for both industrial and logistic uses. The incorporation of autonomous robots in large-scale

factories and logistics centers has become a common practice for reducing the strain on human labor.

For many years, autonomous guided vehicles (AGVs) [4] have dominated the robot industry due to their efficiency in handling manufacturing processes and logistics tasks, such as picking, packing, and palletizing, along pre-defined pathways. However, their inflexibility in adjusting to route changes and limited ability to collaborate with other systems or human operators has led to the development of a more advanced technology: autonomous mobile robots (AMRs) [5]. These robots have the capability of decision-making and autonomous navigation, without being restricted to a pre-defined path.

Another crucial consideration in terms of the integration of robots into human environments is the requirement for a proper understanding of the surrounding environment to avoid obstacles and unexpected encounters with humans or other objects. In fast-growing industrial environments with high traffic and narrow hallways surrounded by various objects and people, omnidirectional mobile robots (OMRs) [6] may be a superior solution due to their ability to move in any direction. However, their overlooked lower-level control design may not be effective in handling continuously changing loads. Thus, advanced control design, even for the lower-level control, is necessary to ensure the effectiveness of OMRs in heavy logistics duties.

In this research study, a design for a mobile robot has been proposed, featuring four Mecanum wheels driven by a bridge motor driver and controlled by a myRIO microprocessor. The rotation speed of these wheels allows for control over the forward, backward, and sideways movements, as well as the turning, of the robot. This research focused on studying different research and ideas from different projects and putting those puzzles together to create an improved and better-performing autonomous mobile robot.

The study aimed to develop a closed-loop feedback control system that incorporated both feedforward and Disturbance Observer (Dob) [7] with a graphical interface. The upper computer software was designed to enable remote control and monitoring of the robot, as well as to provide a user-friendly human–computer interaction.

Automatic navigation and mapping were performed using the Robot Operating System (ROS), which provided a Navigation Stack or Automatic Navigation System. This 2D or 3D [8] method integrates information from odometry, sensor data, and a goal pose to produce safe velocity commands. The Navigation Stack can generate the shortest path and avoid obstacles, even if those obstacles are not predetermined in the map data.

In order to build a map of the environment, Simultaneous Localization and Mapping (SLAM) was utilized. The G-Mapping [9] Package was employed for the robot, utilizing multiple LiDAR and odometry data and employing graph-based optimization to generate a highly accurate representation of the environment.

2. Designing Hardware Architecture

The design and construction of an autonomous robot involves a holistic consideration of both its mechanical and electrical components. This integrated approach is critical in ensuring that the robot functions optimally and efficiently in fulfilling its intended tasks. The developed robot was named “Motion Bot” and its mechanical and electrical components are thoroughly described in the subsequent sections of this paper. The comprehensive analysis of the mechanical and electrical components plays a critical role in illuminating the intricacies and interdependencies of the various elements that comprise the autonomous robot’s architecture.

2.1. Mechanical Components Design

The autonomous robot is designed with a lightweight aluminum body suitable for indoor environments. The design of the robot’s body was created using computer-aided design (CAD) software, which was utilized to perform simulations to calculate the load-bearing capacity of the robot. Upon successful design, the chassis was manufactured

using a computer numerical control (CNC) machine. Figure 1 depicts the actual physical appearance of the robot.

(a) (b)

Figure 1. Appearance of Motion Bot (Experimental Robot) (**a**) without robotic arm; (**b**) with robotic arm.

Mobile robots equipped with non-holonomic systems possess the ability to move in a variety of directions regarding their current positions and orientations. This feature, known as omnidirectionality, is highly sought after in the field of mobile robotics. Several types of omnidirectional wheels exist, each with their own distinct advantages and disadvantages. The most common types of omnidirectional drives are the Kiwi and Holonomic systems [10], which require a precise arrangement to achieve omnidirectional motion. However, these wheels are not suitable for climbing ramps and have a lower capacity (approximately 50%) [11] for multi-directional movement. In contrast, Mecanum wheels, invented by Bengt Ilon, are highly efficient for both forward and reverse movements, as well as lateral movements. The orientation of Mecanum wheels can be arranged in a conventional manner, with lateral motion achieved through wheel velocity control.

In the current research, "Motion Bot" was equipped with four Mecanum wheels with a 100 cm diameter each, with twelve internal rollers at a 45-degree angle with the Y axis of the wheel. The wheels were connected to the main body frame via a suspension mechanism that provides surface contact conformity and reduces vibrations on the robot body.

Figure 2 presents a visual representation of the kinematic vector direction of the chassis, which incorporates the Mecanum wheel and its internal rollers. The procedure for determining the kinematics [12] of the system involves first calculating the inverse kinematics, and then calculating the pseudo-inverse [13]. This was achieved by utilizing a Cartesian coordinate system, which facilitated the analysis of vectors and other relevant variables. The list of variables and their definitions are listed in Table 1 also list of all symbols used in this article is expressed in Appendix A section.

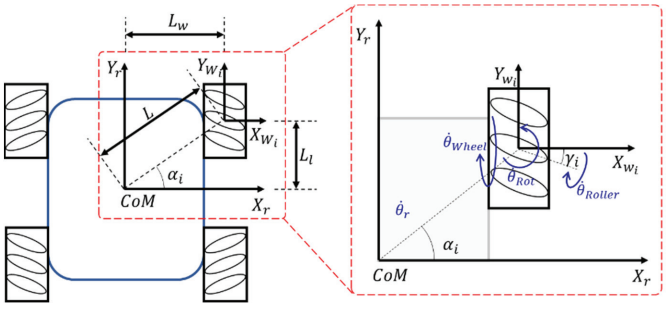


Figure 2. Direction of speed vector on Robot and Mecanum wheel.

Table 1. Robot's kinetic model variable and definition.

Variable	Definition
$\dot{Y}_r$	Instantaneous longitudinal velocity of the robot
$\dot{X}_r$	Instantaneous lateral velocity of the robot
$\dot{\theta}_r$	Angular velocity of the robot
$\dot{Y}_{Wi}$	Instantaneous longitudinal velocity of the i wheel
$\dot{X}_{Wi}$	Instantaneous lateral velocity of the i wheel
$\dot{\theta}_{Wheel}$	Angular velocity of the wheel along X_{Wi} axis (pitch axis)
$\dot{\theta}_{Rot}$	Angular velocity of the wheel along Z_{Wi} axis (yaw axis)
$\dot{\theta}_{Roller}$	Angular velocity of the Roller when it contacts the ground
γ_i	Rotation angle between the i wheel frame and the roller frame
α_i	Angle between robot main frame and the i wheel frame
L_w	Distance between robot coordinate and i wheel along x-axis
L_l	Distance between robot coordinate and i wheel along y-axis
R	Wheel radius
r	Roller radius
CoM	Center of mass of the robot

To derive the kinematic equation, first, the relation between wheel velocity and the vehicle velocity was studied:

$$\dot{X}_{Wi} = \dot{X}_r + \dot{\theta}_r \cdot L \cdot \cos\left(\frac{\pi}{2} + \alpha_I\right) \tag{1}$$

$$\dot{Y}_{Wi} = \dot{Y}_r + \dot{\theta}_r \cdot L \cdot \sin\left(\frac{\pi}{2} + \alpha_i\right) \tag{2}$$

$$\dot{\theta}_{Wi} = \dot{\theta}_r \tag{3}$$

Additionally, the relation between wheel velocity and the roller velocity was found:

$$\dot{X}_{Wi} = r \cdot \dot{\theta}_{Roller} \cdot \cos\left(\frac{\pi}{2} + \gamma_i\right) \tag{4}$$

$$\dot{Y}_{Wi} = r \cdot \dot{\theta}_{Roller} \cdot \sin\left(\frac{\pi}{2} + \gamma_i\right) + R \cdot \dot{\theta}_{Wheel} \tag{5}$$

$$\dot{\theta}_{Wi} = \dot{\theta}_{Rot} \tag{6}$$

Now arranging Equations (1)–(3) in martrix form it can be written:

$$\begin{bmatrix} \dot{X}_{Wi} \\ \dot{Y}_{Wi} \\ \dot{\theta}_{Wi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & L \cdot \cos(\frac{\pi}{2} + \alpha_i) \\ 0 & 1 & L \cdot \sin(\frac{\pi}{2} + \alpha_i) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\theta}_r \end{bmatrix} \tag{7}$$

and arranging Equations (4)–(6) in matrix form:

$$\begin{bmatrix} \dot{X}_{Wi} \\ \dot{Y}_{Wi} \\ \dot{\theta}_{Wi} \end{bmatrix} = \begin{bmatrix} 0 & r \cdot \cos(\frac{\pi}{2} + \gamma_i) & 0 \\ R & r \cdot \sin(\frac{\pi}{2} + \gamma_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\theta}_{Wheel} \\ \dot{\theta}_{Roller} \\ \dot{\theta}_{Rot} \end{bmatrix} \tag{8}$$

From Equations (7) and (8) it can be written:

$$\begin{bmatrix} \dot{\theta}_{Wheel_i} \\ \dot{\theta}_{Roller_i} \\ \dot{\theta}_{Rot_i} \end{bmatrix} = \begin{bmatrix} \frac{1}{R \cdot \tan(\gamma_i)} & \frac{1}{R} & \frac{L}{R}(\cos(\alpha_i) - \sin(\alpha_i) \cdot \cot(\gamma_i)) \\ -\frac{1}{r \cdot \sin(\gamma_i)} & 0 & L \cdot \frac{\sin(\alpha_i)}{r \cdot \sin(\gamma_i)} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\theta}_r \end{bmatrix} \tag{9}$$

Within the context of Equation (9), the angular velocity of the roller is not a focal point of consideration as the wheels are securely attached to the motor, thereby eliminating any potential for rotational velocity in the yaw direction. Hence, through considering the angular velocity of the wheel, the conclusion can be:

$$\dot{\theta}_{\text{Wheel}_i} = \frac{1}{R \cdot \tan(\gamma_i)} \cdot \dot{X}_r + \frac{1}{R} \cdot \dot{Y}_r + \frac{L}{R}(\cos(\alpha_i) - \sin(\alpha_i) \cdot \cot(\gamma_i)) \cdot \dot{\theta}_r \tag{10}$$

Table 2 is listed with the wheel and roller angular parameters for each wheel of the experimental robot.

Table 2. Wheel and roller angular parameter and their values.

Symbol	Wheel_1	Wheel_2	Wheel_3	Wheel_4
α_i	$\frac{\pi}{6}$	$\frac{5\pi}{6}$	$\frac{7\pi}{6}$	$\frac{11\pi}{6}$
γ_i	$-\frac{\pi}{4}$	$\frac{\pi}{4}$	$-\frac{\pi}{4}$	$\frac{\pi}{4}$

(Note: $L \cdot \cos(\alpha_i) = L_w$ // $L \cdot \sin(\alpha_i) = L_l$).

Substituting the I and I_i value in Equation (10), we can rewrite the equation as it is written below:

$$\begin{bmatrix} \dot{\theta}_{\text{Wheel}_1} \\ \dot{\theta}_{\text{Wheel}_2} \\ \dot{\theta}_{\text{Wheel}_3} \\ \dot{\theta}_{\text{Wheel}_4} \end{bmatrix} = \frac{1}{R} \begin{bmatrix} -1 & 1 & (L_l + L_w)((\cos(\alpha_1) - \sin(\alpha_1)) \\ 1 & 1 & -(L_l + L_w)((\cos(\alpha_2) - \sin(\alpha_2)) \\ -1 & 1 & -(L_l + L_w)((\cos(\alpha_3) - \sin(\alpha_3)) \\ 1 & 1 & (L_l + L_w)((\cos(\alpha_4) - \sin(\alpha_4)) \end{bmatrix} \begin{bmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\theta}_r \end{bmatrix} \tag{11}$$

Equation (11) is the inverse kinematics of the system, and to find the forward kinematics, the pseudo-inverse process of Equation (11) must be processed, and then the equation will be:

$$\begin{bmatrix} \dot{X}_r \\ \dot{Y}_r \\ \dot{\theta}_r \end{bmatrix} = \frac{R}{4} * \begin{bmatrix} -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 \\ \frac{1}{(L_l+L_w)((\cos(\alpha_1)-\sin(\alpha_1))} & \frac{-1}{(L_l+L_w)((\cos(\alpha_2)-\sin(\alpha_2))} & \frac{-1}{(L_l+L_w)((\cos(\alpha_3)-\sin(\alpha_3))} & \frac{1}{(L_l+L_w)((\cos(\alpha_4)-\sin(\alpha_4))} \end{bmatrix} * \begin{bmatrix} \dot{\theta}_{\text{Wheel}_1} \\ \dot{\theta}_{\text{Wheel}_2} \\ \dot{\theta}_{\text{Wheel}_3} \\ \dot{\theta}_{\text{Wheel}_4} \end{bmatrix} \tag{12}$$

2.2. Hardware Connection and Configuration

For the experimental robot divide, the electrical components were divided into three classes. The first one is the decision-making and control components, the second one is the sensors, and the last one is the power system. Figure 3 shows the hardware connection of all mobile robot parts, where remote PC is the upper computer base. The ROS master is executed from here, which sends all the control instructions using a common Wi-Fi signal channel. Raspberry Pi works as a second upper computer base that collects data from LiDAR and camera sensors. MyRIO works as the main controller, which receives control instructions from the upper computer base through Wi-Fi to control the DC (Direct Current) motors through the bridge driver, as well as send encoder data sets as a ROS node. For the power source of the robot, a battery of 24 V was used with BMS (Battery Management

system) and the power carrying capacity was 12 Ah. A 24 V to 12 V DC to DC converter is used here, as the motors' running voltage is 24 V, but myRIO and Raspberry Pi can operate with a 12 V maximum power supply.

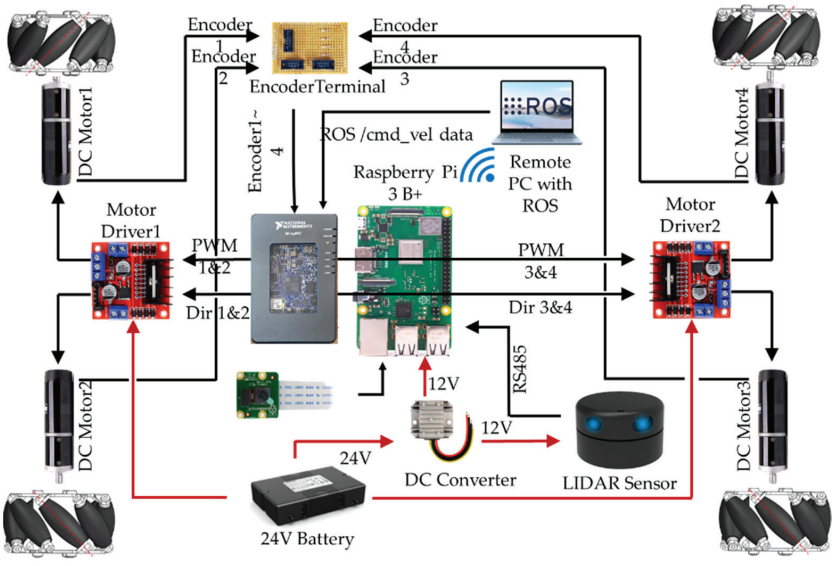


Figure 3. Connection Diagram of Different electrical components.

It is acknowledged that utilizing a single upper computer, such as Raspberry Pi, for processing heavy data may result in a decrease in performance. To ensure efficient monitoring and prompt response, a remote PC is utilized in conjunction with Raspberry Pi. Figure 4 shows the data flow within this connection mentioning the ROS topic name.

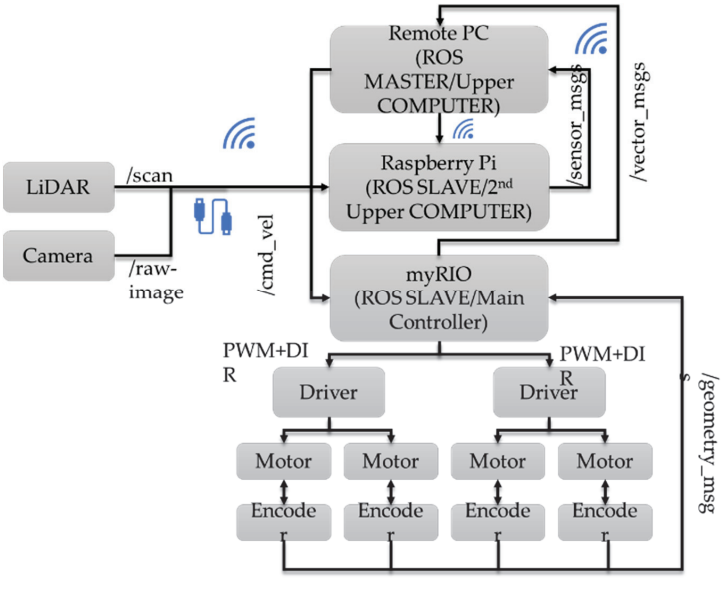


Figure 4. Visualization of Data flow.

3. Designing Software Architecture

The software architecture design will concentrate on the creation of velocity control mechanisms for the motors, the mapping of the surrounding environment, and the implementation of an autonomous navigation system. The control architecture has been bifurcated into two sections for comprehensive elucidation. The first component deals with the velocity control, which is referred to as the lower-level control and is exclusively accountable for executing directives without any decision-making capacity. Conversely, the higher-level control imbues the robot with the capacity to perceive its environment, generate trajectories towards a designated target, and make adaptive choices for obstacle avoidance.

3.1. Lower-Level Control Software Design

The control design of a mobile robot can be approached from either a dynamic or a kinematic perspective. While the dynamic approach involves the calculation of the real-time system and is more complex, the kinematic approach, which consists of both the kinematic loop and dynamics loop, is simpler and can guarantee stability through proper tuning. This study adopts the kinematic approach for the control design and classifies it into four sections. The first section focuses on finding the system identification and establishing a nominal model, followed by the feedback control loop, along with the feedforward and disturbance observer, in the second section. The third section addresses the design of various trajectories to evaluate the control performance, and the final section analyzes the robustness of the closed-loop system. LabVIEW programming was utilized for the lower-level control, providing a Human Machine Interface (HMI) that allows for real-time adjustment of control parameters and the creation of trajectories for automated guided robots.

3.1.1. System Identification

Since electrical components, such as motor resistance and inductance, are controlled by the motor driver, we will focus on the mechanical parts for system identification. The nominal model for each wheel was identified through this process. Figure 5 shows a block diagram of the process used for this process.

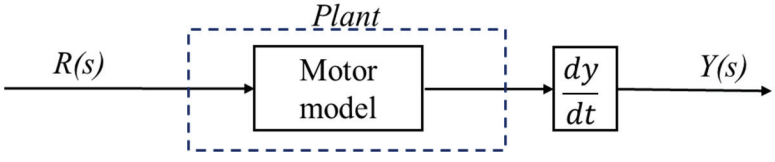


Figure 5. System Identification process block diagram.

For system identification [14] of four wheels, a chirp sine signal of 0~10 Hz was applied for 10 s. PWM value was 0~1%, and the sine magnitude was 0.7, 0.75, 0.8, and 0.85. Figure 6 shows the body plot diagram of model design.

As it can be seen from the body plot, the magnitude has dropped around 20 dB during 1 log-based frequency change, so we can be assured that the system model is the 1st order [15] and that the mathematical form of the nominal model should be:

$$\frac{\text{Output}}{\text{Input}} = \frac{1}{J_n s + B_n} \tag{13}$$

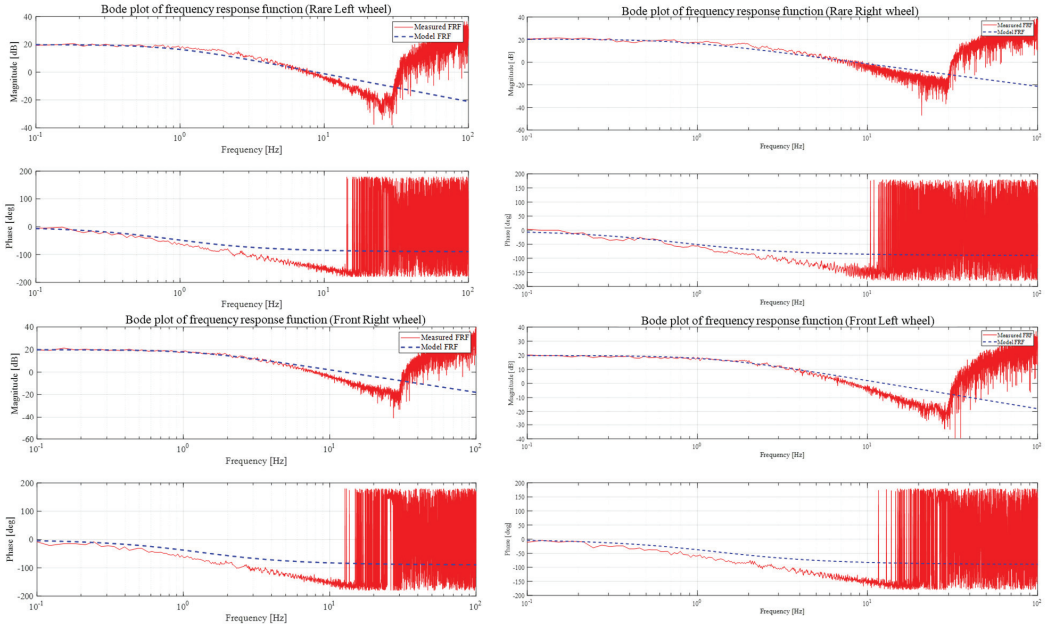


Figure 6. Bode plot Diagram.

3.1.2. Control Design

The method for motor control [4] used in this experiment was speed-voltage looped control. Voltage was considered equivalent to velocity, and control was designed for each individual motor. Then, from the forward kinematics on Equation (12), we can calculate the individual motor's velocity to find the total vehicle velocity. The actual velocity provided by each motor encoder can be calculated using Equation (11). Then, using the given velocity and actual velocity, a feedback control loop can be designed. Using the nominal model from Section 3.1.1, a feedback control loop was designed through pole-zero cancelation method [16]. The feedback control equation design was as follows:

$$C_{fb} = \omega_{fb} \cdot J_n + \frac{\omega_{fb} \cdot B_n}{s} \text{ (Here, } \omega_{fb} = 2\pi \times 2 \text{ Hz)} \tag{14}$$

To soothe the loading torque on the DC motor speed and make the response time fast, feedforward compensation was designed by taking the inverse of the nominal model and multiplying it with a low-pass filter. The feedforward control equation for this robot was as follows:

$$C_{ff} = \frac{(J_n s + B_n)}{\frac{s}{\omega_{ff}} + 1} \text{ (Here, } \omega_{ff} = 2\pi \times 10 \text{ Hz)} \tag{15}$$

Even though the use of both feedback and feedforward control were adequate for operating under no-load conditions, there was a noticeable degradation in the control system's performance under varying loads. Furthermore, it was necessary to consider model uncertainty. To mitigate this issue, a disturbance observer was incorporated. This addition will address system disturbances, as well as sensor noise, thereby leading to an enhanced control system performance. For designing a disturbance observer, we have used the inverse of our nominal model with a Q filter. The equation for the Q filter was as follows:

$$Q(s) = \frac{\omega_Q^2}{s^2 + 2\zeta\omega_Q s + \omega_Q^2} \text{ (Here, } \omega_Q = 2\pi \times 2 \text{ Hz)} \tag{16}$$

Figure 7 shows a block diagram of the control algorithm, where $\dot{x}_r$, $\dot{y}_r$, and θ_r are linear x, linear y, and the angular velocity of the robot, and they are controlled with a feedforward and a feedback loop, along with a disturbance observer. The list of symbols used in Figure 7 and there meanings are listed in Table 3. A study using such kind of control algorithms is conducted in a journal by Mu-Tian Yan and Yau-Jung Shiu [17], and it was established that this kind of control strategy was adequate for controlling motors.

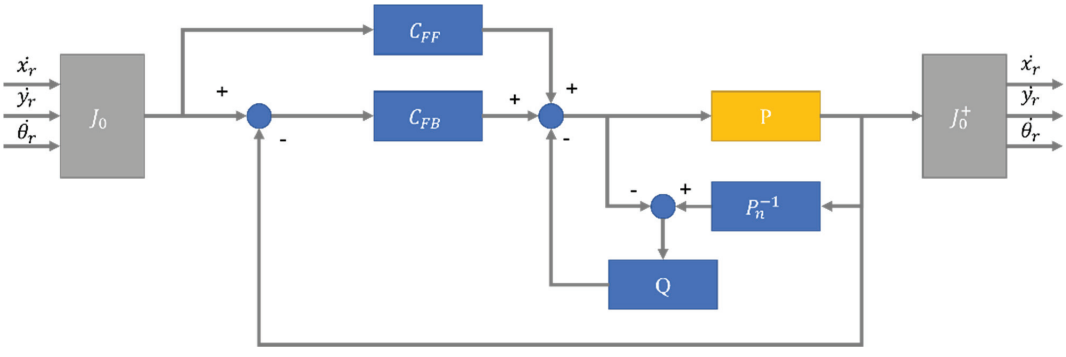


Figure 7. Block Diagram of control algorithm.

Table 3. Lower-level control system variable and definition.

Variable	Definition
J_n	Moment of inertia (0.0073969)
B_n	Friction constant (0.43571)
s	Output variable for Laplace transform
ω_{fb}	Feedback band width
C_{fb}	Feedback control
C_{ff}	Feedforward control
ω_{ff}	Feedback band width
ζ	Damping ratio
ω_Q	Q-filter band width

3.1.3. Control Performance Test

The performance evaluation of the lower-level control was conducted using a trajectory similar to the one shown in Figure 8. The trajectory incorporated straight motion, arc cornering, and turning motion with varying velocity for the purpose of testing. Data collection was performed utilizing the USATR (Universal Synchronous/Asynchronous Receiver/Transmitter) method [18], and the results were plotted using MATLAB. The velocity data was calculated directly from the kinematics, while the position data was obtained through the application of the discrete time integration method on the velocity data.

In Figure 9, the velocity plot and velocity error plot have been shown to follow the guided trajectory. Here, V_x, V_y, and W are longitudinal, lateral, and angular velocity, respectively.

From the error plot, it can be clearly seen that the velocity error is below 0.05 m/s on average. There is some overshoot on certain positions, but the overall system is stable and there is no steady state error.

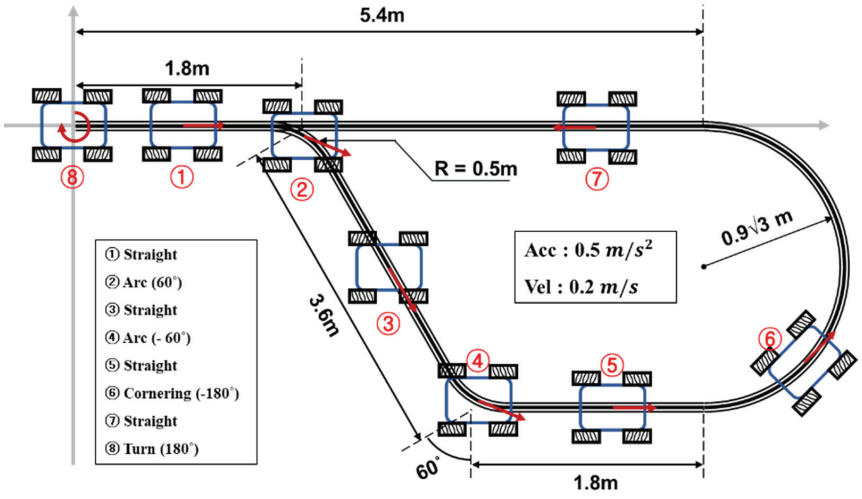


Figure 8. Trajectory of the experimental robot.

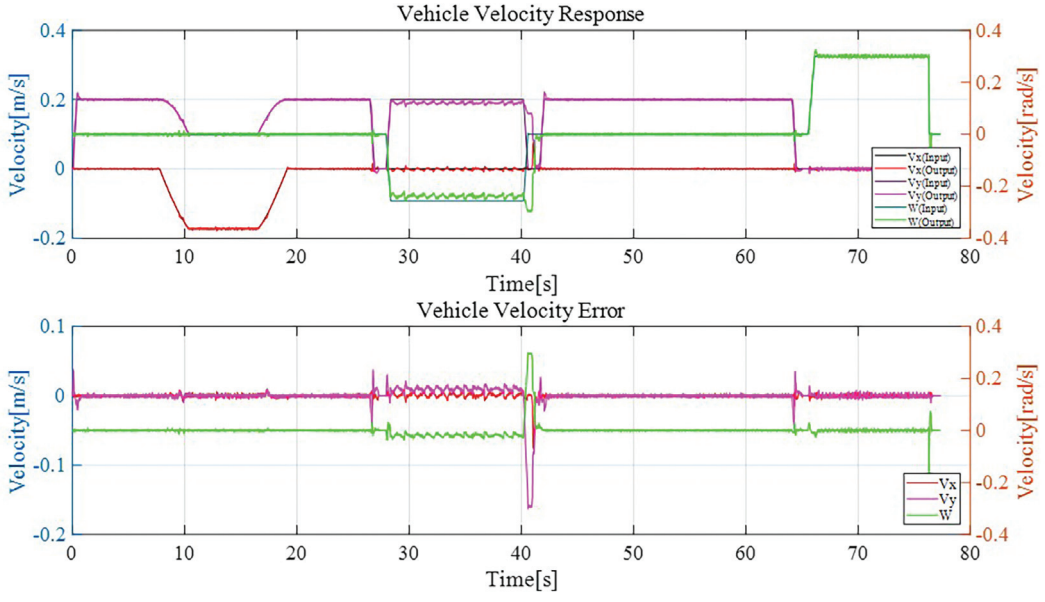


Figure 9. Velocity input vs. output plot.

Figure 10 displays a plot of the commanded position and the actual position, as calculated by the motor encoder. The plot demonstrates that the robot is capable of following the command effectively while traversing straight motion and cornering. However, a negligible error, due to overshoot, is observed during the turning motion. During the evaluation of the lower-level control, the possibility of wheel slip was not taken into account, as it is addressed during the design phase of the higher-level control.

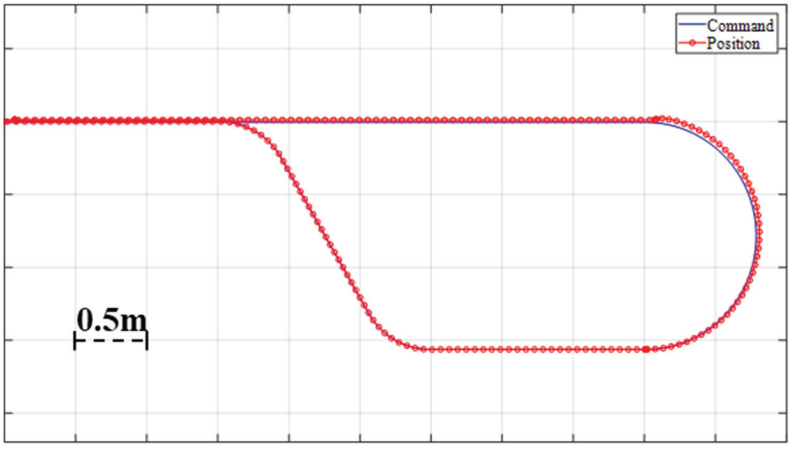


Figure 10. Position input vs. output plot.

3.1.4. Robust Performance Test

In this section, the robustness of the designed control system based on the disturbance observer (DOB) will be analyzed [19]. To analyze robustness, the selection of system uncertainty was studied first. The system uncertainty ±30% of the nominal model for both inertia (J_n) and friction (B_n) was selected and analyzed.

Next, uncertainty weight selection was conducted through the following equations. To describe the generic model uncertainty with a complex norm-bounded multiplicative uncertainty, the equation is:

$$P(s) = (1 + W_2(s)\Delta(s))P_n(s) \text{ where, } \|\Delta(s)\|_\infty \leq 1 \tag{17}$$

The weight $W_2(s)$ is selected so that:

$$\max_{P \in \mathcal{P}} \left| \frac{P(j\omega) - P_n(j\omega)}{P_n(j\omega)} \right| \leq |W_2(j\omega)| \tag{18}$$

Here, a set of perturbed plant models $\mathcal{P}$ is obtained by varying the values of J and B within their variability ranges:

$$\mathcal{P} = \left\{ P(s) = \frac{1}{Js + B} \; [\text{Here}, J = J_n \pm 30\%, \; B = B_n \pm 30\%] \right\} \tag{19}$$

Now, the driven equation is as follows:

$$\frac{P(j\omega) - P_n(j\omega)}{P_n(j\omega)} = \frac{\frac{1}{Js+B} - \frac{1}{J_n s + B_n}}{\frac{1}{J_n s + B_n}} \times \frac{(Js + B)(J_n s + B_n)}{(Js + B)(J_n s + B_n)} = \frac{(J_n s + B_n) - (Js + B)}{Js + B} = \frac{(J_n - J)s + (B_n - B)}{Js + B} \tag{20}$$

Figure 11 shows the selection of uncertainty weight function and its bode plot.

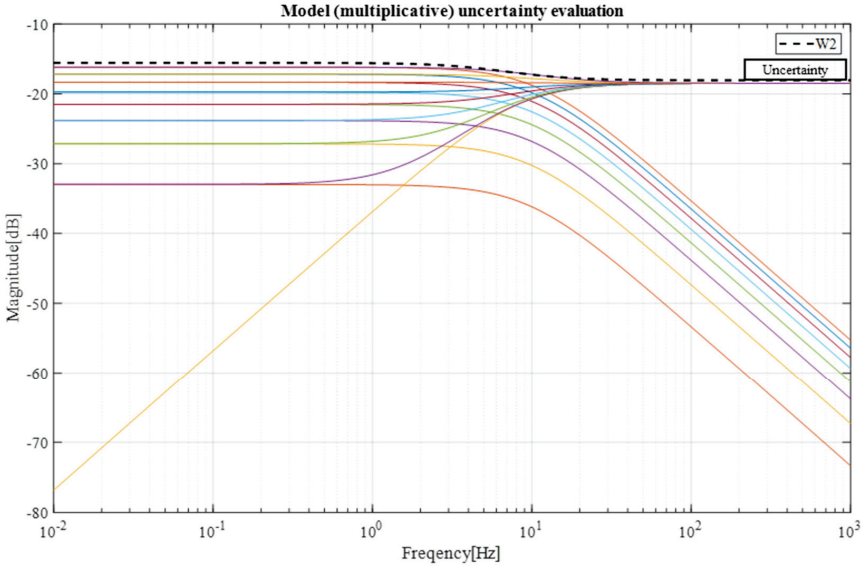


Figure 11. Uncertainty weight selection.

Here, uncertainty weight is selected as:

$$W_2 = K * \frac{1 + \frac{s}{\omega_z}}{1 + \frac{s}{\omega_p}} \left[\text{here, } \omega_z = 2 * \pi * 8,\ \omega_p = 2 * \pi * 6,\ K = 0.125\right] \tag{21}$$

The robust stability for the overall system follows $T' = \frac{P_n C + Q}{1 + P_n C}$, which is shown in Figure 12 for a feedback cutoff frequency from 2 to 10 Hz.

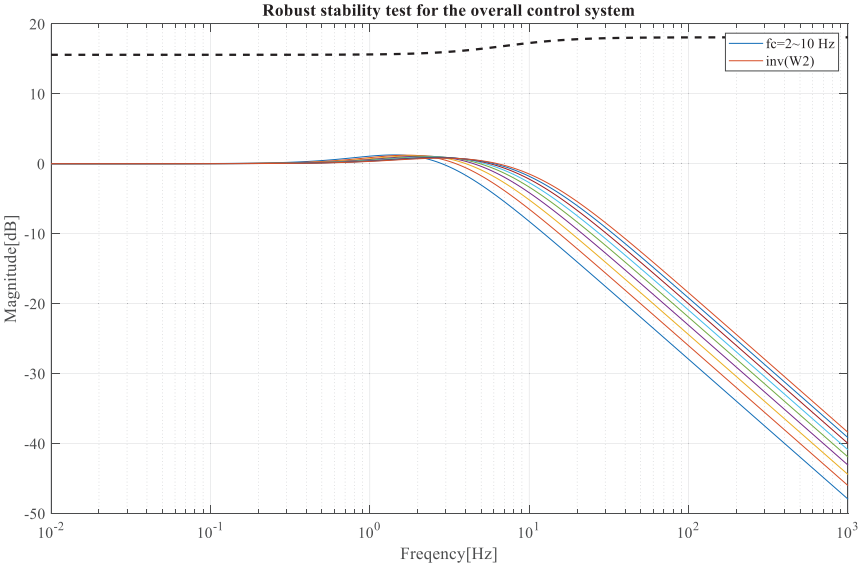


Figure 12. Robust stability for overall system.

3.2. Higher-Level Control Software Design

The higher-level controller plays a crucial role in ensuring the efficient and safe operation of the robot by generating a reference path that avoids potential collisions. This is achieved through the creation of a map of the environment that localizes the robot within it. The software utilized by the upper computer is based on the Robot Operating System (ROS), which serves as a framework for programming hardware components such as motors, sensors, and drivers.

ROS supports multiple programming languages, including C++, Python, and Java, and allows for the use of multiple programming languages across multiple connected computers. Additionally, ROS is capable of executing multiple executables in parallel, allowing for both synchronous and asynchronous data exchange between them. These executables, referred to as ROS nodes, share data through ROS topics.

ROS also provides graphical interfaces, such as RVIZ [20], from which we can visualize all the sensor data and related values in real time. ROS also comes with SLAM and Navigation stack packages, which have the adequate processes to make a perfect map of the environment and navigate it with safety. For designing the higher-level control software, the 'turtulebot3' [21] and 'Nox' [22] package structures were used with modification needed for our experimental robot. Additionally, as three LiDAR sensors were installed, we used a lidar merger package to combine those scan data.

3.2.1. ROS Package Modification

For architecting the higher-level software, several suitable modifications were performed, the most notable of which was the odometry package modification. As robots can also move in the lateral direction, a calculation was needed to consider this motion. Additionally, to use mechanomes we must consider the pose error due to slip ratio. To overcome this, we used the pose created by the wheel encoder data and made an estimated odometry using sensor fusion of the lidar sensor, IMU, and encoder data.

3.2.2. Connection of Higher and Lower Software

For this experiment, NI myRIO was used for lower-level control and collecting odometry data, which can be programmed by NI LabVIEW software. LabVIEW provides an add-on named "ROS for LabVIEW," which can be downloaded from the VI Package store. However, as ROS is operated mainly using the Ubuntu (Linux) system and LabVIEW software is mainly operated using the Windows system, we need to take several steps to connect these two systems. The preconditions to connect ROS with LabVIEW are:

1. All the Wi-Fi connections should be under the same network and the first 7 digits of the IP address have to be the same for all devices.
2. Host IP address should be added to both Ubuntu and Windows systems using Administrator's access.
3. Accessibility of each device should be checked using the "ping" command.
4. The antivirus network protection should be off, or new protocols should be made for those IP addresses.
5. ROS Master IP address and ROS Host IP address should be set before running ROSCORE.

If Windows Firewall does not allow the ROS network to communicate LabVIEW, then Windows Firewall Rule should be made. The steps are:

1. Open Control Panel > System and Security > Windows Firewall > Advanced Settings
2. Right-click "Inbound Rules" and select "New Rule"
3. Assign the following properties to the new rule
    - Select "Custom Rule" under "Rule Type."
    - Under the protocol and port for the protocol type, select "ICMPv4."
    - Apply to all local and remote IP addresses in the range.

- In terms of connections you are allowed to choose, check “Domain,” “Private,” and “Public” in Profile.
- Assign a name, such as “ICMPv4 rule for ROS communication,” and choose “Finish.”

After successful establishment of the ROS network, it is time to Run ROS on the LabVIEW system. Figure 13 shows a simple VI, which will subscribe to the /cmd_vel topic and read the twist message of Linear and Angular Velocity. Reading those messages from ROS, the LabView will execute Linear and Angular motion by running the motors through the myRIO device.

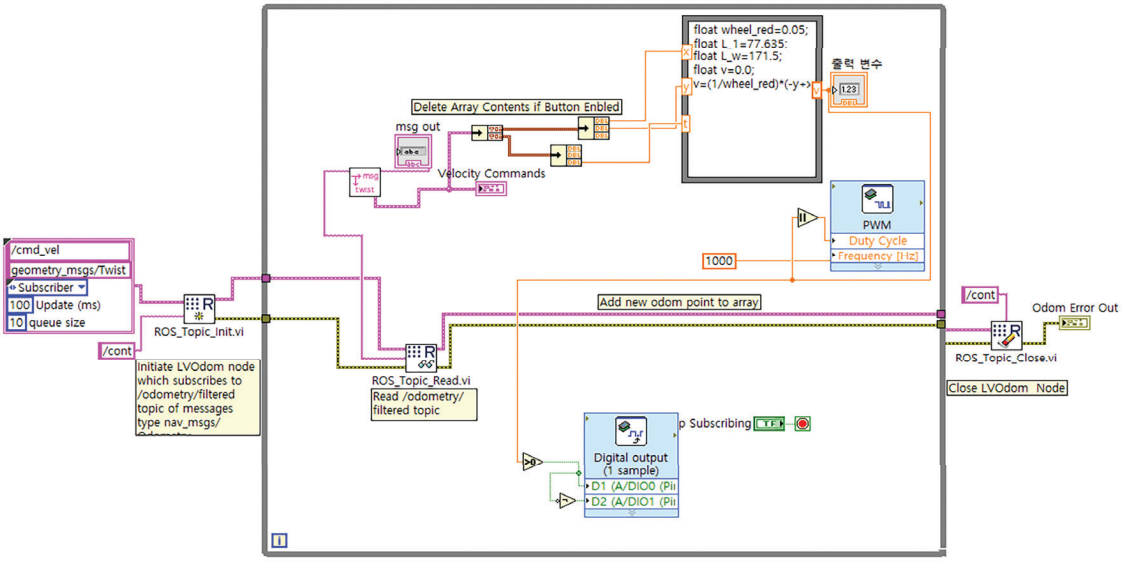


Figure 13. ROS Programming with LabVIEW (subscriber to cmd_vel topic).

Before running the VI, we should double click ROS_Topic_init.vi and re-correct the topic name and message type if needed. It is always best practice to run the ROS Master inside LabVIEW to ensure the node is working fine. Otherwise, some errors can occur, and it will become harder to reconnect.

The complete software, Architecture, is also divided into several tasks, such as receiving velocity commands through a node from the Master Computer, processing the input velocity through control algorithms to match that and generate the PWM and direction signal for motor drivers, and lastly, calculating the velocity of the robot reading the encoder data and sending it to the ROS Master through another node. Figure 14 shows a program in LabVIEW where a subscriber node is created, which will receive velocity command, and another publisher node is created, which will publish the linear and angular velocity of the robot.

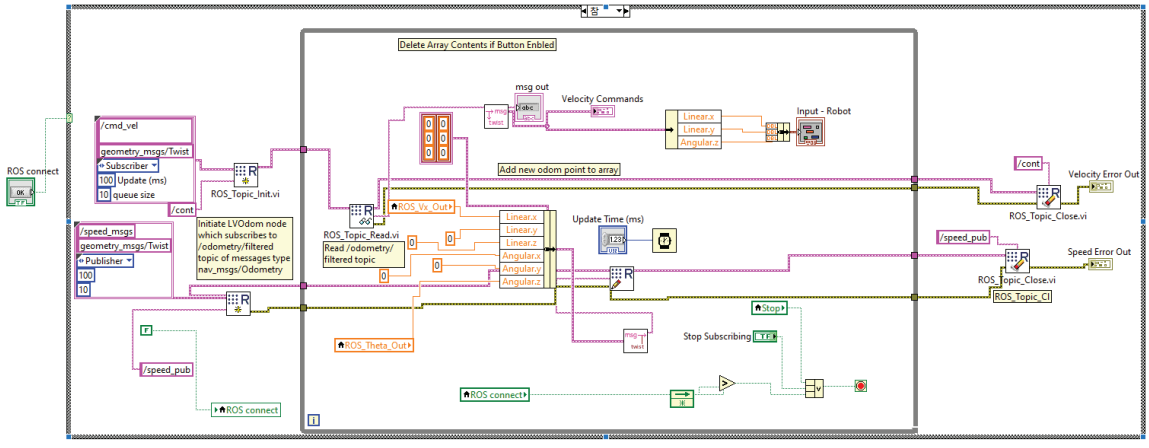


Figure 14. ROS Programming with LabVIEW (publisher and subscriber of different topic).

4. SLAM Based on ROS

SLAM refers to the process of creating a map of an unknown environment while simultaneously determining the robot's location within it. This is achieved through the use of sensors, such as lidar sensors or GPS, and wheel odometry. The task of simultaneously performing both localization and mapping presents a significant challenge, akin to navigating and mapping a large, unknown house. The SLAM algorithm is dependent on probabilistic models, which take into account uncertainty and estimation processes. Researchers from diverse fields are actively exploring ways to improve the representation of both the environment and the robot's position. The advancement of various sensors has led to the widespread use of SLAM in various applications, including rescue operations, archaeology, and military and industrial contexts. One of the most widely used SLAM methods in the ROS framework is the GMapping algorithm. This method is based on the Rao–Blackwellized particle filter (RBPF) [23] and has proven to be highly effective in acquiring maps of unknown dynamic environments. Other popular SLAM algorithms, such as Hector SLAM and First SLAM, have unique uses and capabilities, but GMapping stands out for its ability to fuse multiple sensor data sources together using a Kalman filter [24] to achieve more accurate estimations.

To make a perfect SLAM, four sets of data are required. First, the robot's position in both the steady and the moving condition is needed. For this experiment, the initial position was introduced to the robot. Second, sensing or measuring surrounded obstacles from the robot; this was carried out by using the LiDAR sensor. Third, the initial map of the robot, which can be made at the steady position of the robot, and fourth, a path by which the robot moves in that unknown environment, which was covered using odometry and IMU sensor data. However, as our robot is a medium-sized mobile robot, using only one LiDAR sensor is not enough. This is because if the LiDAR is installed only on the top, it cannot cover the area below that. To solve this problem, we have implemented three LiDAR sensors, shown in Figure 15. One LiDAR on the top will cover 360°, and the other two LiDAR on the front and back will cover 180° from the bottom. Merging them together will provide precise information about surrounding obstacles.

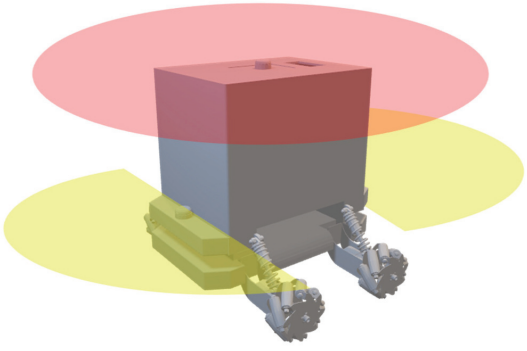


Figure 15. Position of LiDAR sensors and covering area.

To merge these three LiDAR data together, a ROS package was created by following different papers [25,26] related to multi-LiDAR sensor collaboration approaches. Figure 16 shows the algorithm used to merge three lidar sensor data and publish it as one laser data. In this algorithm, ROS slave on the Raspberry Pi board is responsible for collecting all data sets from three lidar sensors and publish it as a node with a different topic name for each individual LiDAR. Then, ROS Master, running on a laptop, will combine those topics and recollect those data sets. Then, through the synchronization of those data, a point cloud will be created. Then, we can merge data using the point cloud library and publish that merged point cloud data. After that process, we can convert the point cloud data into laser data and publish the merged laser data.

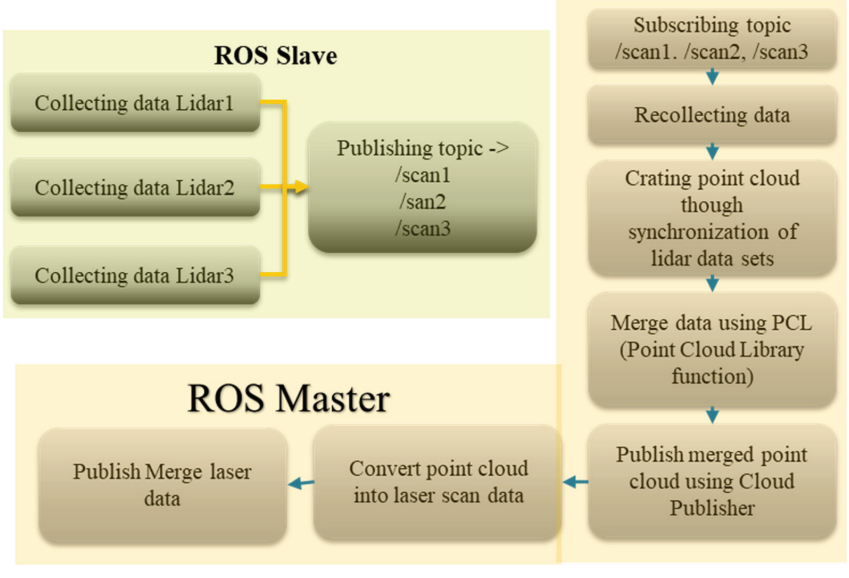
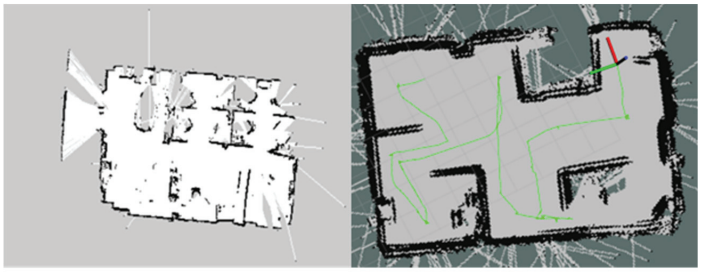


Figure 16. Algorithm for merging 3 lidar scan data for mapping.

Figure 17 shows the difference between performance of SLAM using single LiDAR and Multiple LiDAR. In Figure 17b, we can clearly see a better performance and clear map of the environment using multiple LiDAR. We can also see some errors which were mainly generated due to noise, and this can be reduced through further research and development. The green line in Figure 17b indicates the trajectory of the robot while making the map with the SLAM algorithm.



(**a**) (**b**)

Figure 17. (**a**) SLAM performance with single Lidar; (**b**) SLAM performance with multiple Lidar and IMU.

5. Navigation Based on ROS

The Navigation Stack is a highly advanced package of ROS software, capable of performing both localization and autonomous navigation with a planned trajectory. This package is comprised of three sub-packages, including the Adaptive Monte Carlo Localization (AMCL) [27] module, which is responsible for localizing the robot within a map using particle filters and odometry and laser data. In its initial position, the upper computer has limited data to calculate the exact position of the robot, resulting in a large circular area. However, as the robot moves, the point cloud accumulates more data, allowing for a more accurate calculation of the robot's position. Figure 18 shows an implementation of AMCL, where red arrows show the possible position of the robot within the map.

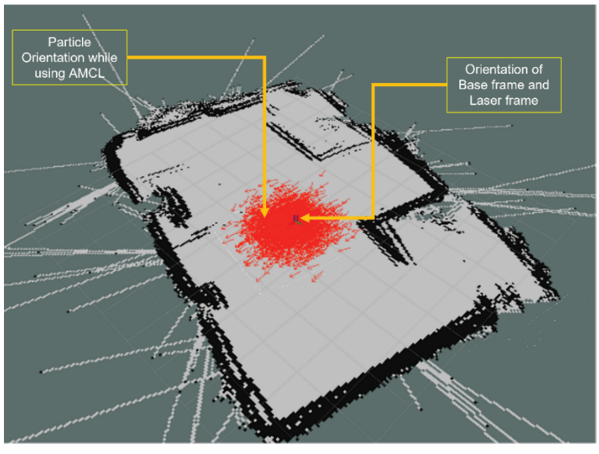


Figure 18. Monte Carlo Localization.

The second sub-package, the Map Server, is responsible for reading the map created by SLAM from disk storage and serving it as a topic named /map to the ROS master. The third sub-package, the Move Base package, is responsible for generating a secure and efficient path for autonomous navigation. This package reads various initial conditions, such as the robot's footprint dimensions, obstacle range, and maximum and minimum linear and angular velocity, from YAML files. It then generates a path using algorithms [28] such as A-star, Rapidly-exploring Random Tree (RRT), or RRT Star, and various optimization techniques, such as Genetic Algorithm (GA), Artificial Intelligence (AI), and Particle Swarm Optimization (PSO).

Figure 19 shows the roll of different ROS navigation stack files [29]. An important thing to note here is that the cost map is divided into a global cost map and a local cost map,

where the global cost map contains overall information about the entire environment and the local cost map contains information about the surrounding obstacles of the robot. The global path planner is responsible for creating the main trajectory to reach the goal, while the local path planner is responsible for avoiding small obstacles by correcting the main trajectory generated by the global planner. The path planner algorithm used in this paper is adopted from a research study conducted by LIU Tianyu, YAN Ruixin, WEI Guangrui, and SUN Lei [29].

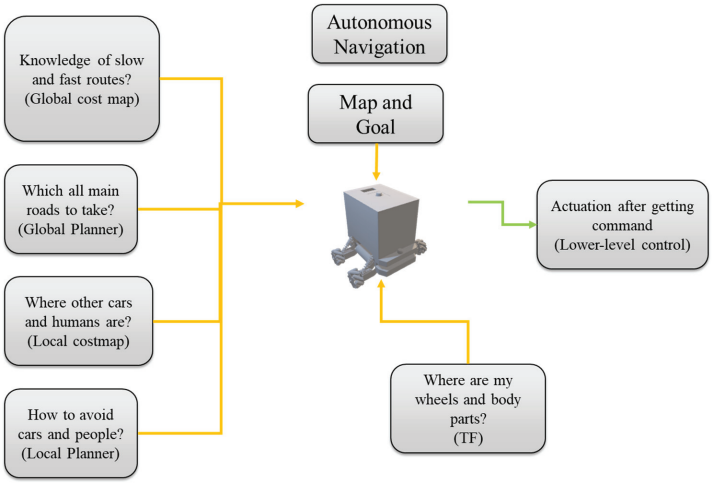


Figure 19. ROS Navigation Stack parts and their roles.

In order to perform autonomous navigation with the robot, modifications to the ROS navigation stack parameters were necessary to account for the specific dimensions and environment of the robot. A threshold of 350 mm was applied on the edge of obstacles to avoid collisions, and proper path planning was executed. In the event of new obstacles (e.g., a walking person) appearing in the path of the planned trajectory, which are not present in the global map, they are added to the local map and the move base package re-plans the path to reach the goal. Additionally, lateral path planning freedom was added by modifying various files in the ROS navigation stack.

6. Results and Discussion

In this section, results and analyses will be discussed to check accomplishments. Through that discussion, some issues and observations that were faced during testing the robot will be mentioned, and further research goals will be determined to take the robot to the next level. To best discuss the results, this section was divided into two sub-sections. In the first section, the lower-level control performance will be discussed, and in the second section, the higher-level control performance will be discussed.

6.1. Lower-Level Control Results

From the experiment result attached in Figure 9, it can be observed that the lower-level controller can perform with a gratifying accuracy. In the position plot in Figure 10, the error was less than 0.05. Through the robustness analysis, we found that both the disturbance observer loop and overall control loop were under the curve of uncertainty, weighting the function magnitude line shown in Figures 11 and 12. Thus, theoretically, both of the loops were stable, which means that even if we added 30% more load than expected, the velocity performance of the mobile robot would remain stable. Additionally, the control system parameter was adjustable with a graphic interface, which makes the robot suitable for operating with a variable load.

6.2. Higher-Level Control Results

In the higher computer, the software architecture was adequate to perform a successful SLAM, although lots of error lines can be found outside the boundary shown in Figure 20. These mainly occurred due to sensor noise and reflections from different light sources. Further noise reduction algorithms can be developed for future research. Additionally, the LiDAR sensor has less effectiveness while passing through a glass or mirror. This phenomenon can be avoided by using more precious sensors or 3D camera sensors.

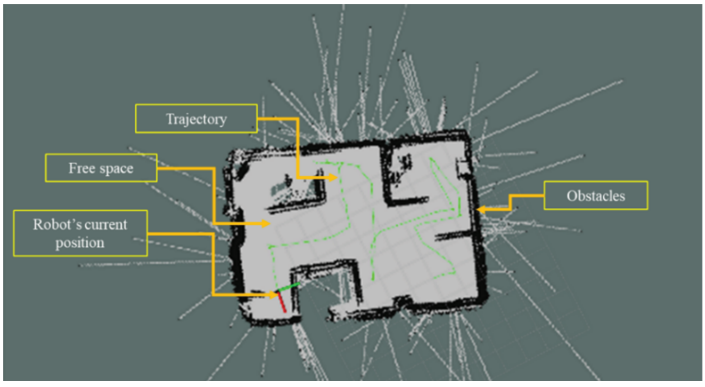


Figure 20. SLAM performance with MotionBot.

For the navigation architecture, the software prosperously made a path to the goal, avoiding all known and unknown obstacles. Thus, it can perform automatic navigation inside an indoor environment successfully. Figure 21 shows a performance of autonomous navigation of our mobile robot. To start the navigation, the initialization of the robot's current location should be input with the ROS RVIZ interface and, with some iteration, the robot can localize itself perfectly. Then, with the help of the RVIZ interface, or by directly commanding the goal pose, autonomous navigation can be initiated.

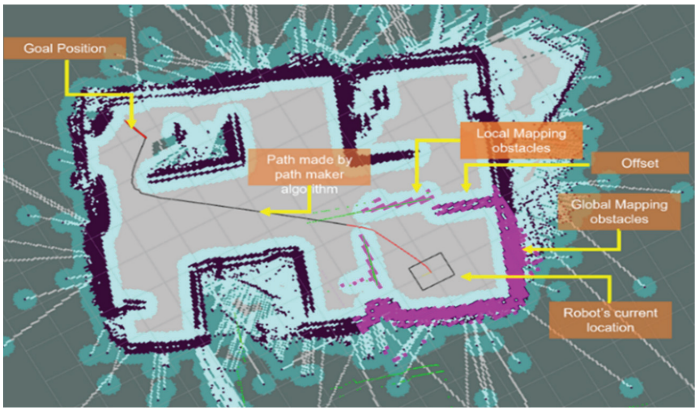
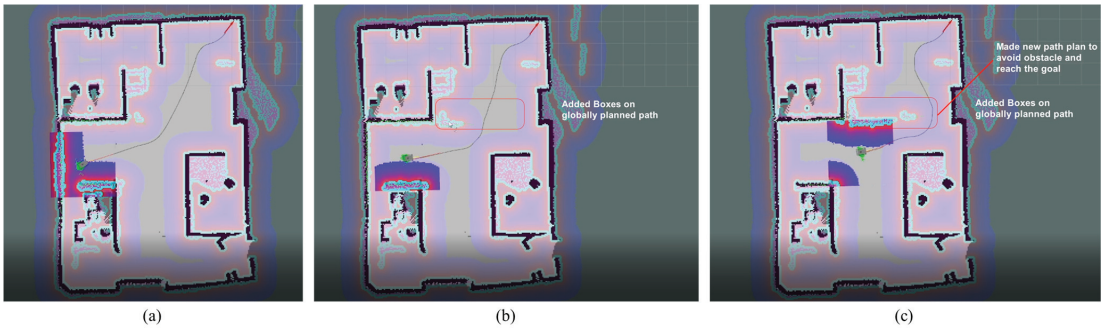


Figure 21. ROS autonomous navigation with MotionBot.

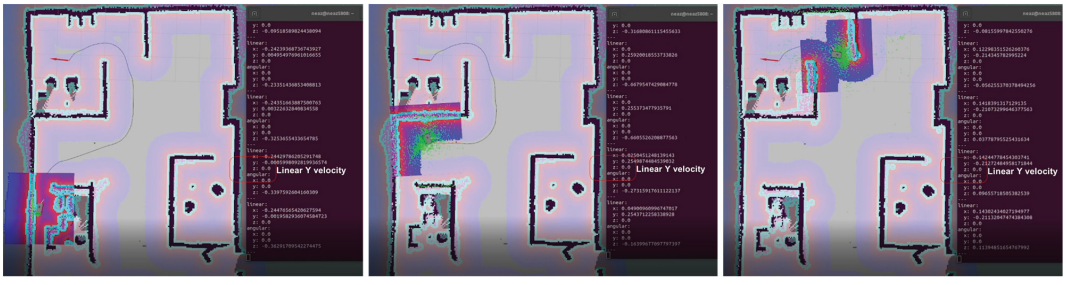
The ability to avoid sudden obstacles, such as a human or unknown object, is also checked with the experimental mobile robot. In Figure 22, it is shown that the robot creates the global path to reach the goal according to the global map. However, as soon as obstacles are detected on the path, the local path planner adjusts the global path to avoid that obstacle and reach the goal.



(a) (b) (c)

Figure 22. (**a**) Global path planner without considering local obstacles; (**b**) Added and detected obstacles on the Global path; (**c**) Correction of Global path to avoid obstacles.

Path planning that considers lateral motion is also checked. Figure 23 shows a successful implication of the linear X and Y axis directional path planning, allowing the experimental mobile robot to take the shortest path to reach the goal.



(a) (b) (c)

Figure 23. (**a**) Path planning in linear X direction; (**b**) path planning in linear Y direction; (**c**) path planning in both linear X and Y direction.

7. Conclusions

The present study aimed to design and develop an omnidirectional mobile robot, which combined the characteristics of both an Autonomous Mobile Robot and an Automated Guided Vehicle. The results obtained from the practical operation of the 'MotionBot' robot, as discussed in previous sections, demonstrated the reliability, improvement, and effectiveness of the proposed techniques. The focus of the study was on enhancing the lower-level control through feedback and feedforward controllers to optimize vibrations and increase stability through a low computational cost. Additionally, the robustness of the robot was considered since it was expected to operate in different environments with different loads. A study and analysis of robustness was conducted, and the results confirmed its adequacy.

In order to enhance the sensing capabilities of a robot, a fusion of three LiDAR data was executed to improve the accuracy of localization and positioning. The performance of single LiDAR and multiple LiDAR using G-mapping SLAM was evaluated to increase mapping accuracy in unknown environments. The robot successfully reached the goal point while avoiding obstacles in a dynamic environment. A user-friendly GUI was developed using LabVIEW software. However, future research could be conducted to reduce LiDAR noise, address the wheel slip ratio problem, and implement object recognition and tracking technologies. The utilization of OpenCV and TensorFlow can enable the robot to analyze objects, such as human bodies, and follow them using object-following algorithms. The

potential for further improvement, leveraging the capabilities of the ROS platform, holds promise for the logistics and courier industries.

Author Contributions: A.N. and S.L. take the lead in writing papers, developing control algorithms, and conducting experiments. K.N. has reviewed the overall contents and supervised the control development. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the 2020 Yeungnam University Research Grant (No. 220A380108) and was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2022R1C1C1011785).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: This study did not report any data.

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

Abbreviations

The following abbreviations are used in this article:

AMR	Autonomous Mobile Robot
AGV	Automated Guided Robot
ROS	Robot Operating System
DOB	Disturbance Observer
CNC	Computer Numerical Control
IMU	Inertial Measurement Unit
CoM	Center Of Gravity
USATR	Universal Synchronous/Asynchronous Receiver/Transmitter
SLAM	Simultaneous Localization And Mapping
Lidar	Light Detection And Ranging
URDF	Unified Robot Description Format
DoF	Degree Of Freedom
GPS	Global Positioning System
RBPF	Rao-Blackwellized Particle Filter
AMCL	Adaptive Monte Carlo Localization
YAML	Yet Another Markup Language
PSO	Particle Swarm Optimization
RRT	Rapidly Exploring Random Tree

Appendix A. Symbol and Definition

This appendix consists of a list (Table A1) with all the symbols used in this paper with their definitions.

Table A1. List of symbols and their definition.

Variable	Definition
$\dot{Y}_r$	Instantaneous longitudinal velocity of the robot
$\dot{X}_r$	Instantaneous lateral velocity of the robot
$\dot{\theta}_r$	Angular velocity of the robot
$\dot{Y}_{Wi}$	Instantaneous longitudinal velocity of the i wheel
$\dot{X}_{Wi}$	Instantaneous lateral velocity of the i wheel
$\dot{\theta}_{Wheel}$	Angular velocity of the wheel along X_{Wi} axis (pitch axis)

Table A1. *Cont.*

Variable	Definition
$\dot{\theta}_{Rot}$	Angular velocity of the wheel along Z_{Wi} axis (yaw axis)
$\dot{\theta}_{Roller}$	Angular velocity of the Roller when it contacts the ground
γ_i	Rotation angle between the i wheel frame and the roller frame
α_i	Angle between robot main frame and the i wheel frame
L_w	Distance between robot coordinate and i wheel along x-axis
L_l	Distance between robot coordinate and i wheel along y-axis
R	Wheel radius
r	Roller radius
CoM	Center of Mass of the robot
J_n	Moment of inertia (0.0073969)
B_n	Friction constant (0.43571)
s	Output variable for Laplace transform
W_{fb}	Feedback band width
C_{fb}	Feedback control
C_{ff}	Feedforward control
W_{ff}	Feedback band width
ζ	Damping ratio
W_Q	Q-filter band width
$e_i(k)$	Uncorrelated observation errors
L_{pi}	Jacobian matrix of observation model with respect to landmarks
L_v	Jacobian matrix of observation model with respect to robot odometry
L_i	Observation matrix that relates to the sensor output
z_i	The state vector $x(k)$ when observing i^{th} landmark

References

1. Wang, X.V.; Wang, L. A literature survey of the robotic technologies during the COVID-19 pandemic. *J. Manuf. Syst.* **2021**, *60*, 823–836. [CrossRef] [PubMed]
2. Pani, A.; Mishra, S.; Golias, M.; Figliozzi, M. Evaluating public acceptance of autonomous delivery robots during COVID-19 pandemic. *Transp. Res. Part Transp. Environ.* **2020**, *89*, 102600. [CrossRef]
3. Lozoya, C.; Marti, P.; Velasco, M.; Fuertes, J.M. Effective Real-Time Wireless Control of an Autonomous Guided Vehicle. In Proceedings of the 2007 IEEE International Symposium on Industrial Electronics, Vigo, Spain, 4–7 June 2007; IEEE: Vigo, Spain, 2007; pp. 2876–2881.
4. Fragapane, G.; de Koster, R.; Sgarbossa, F.; Strandhagen, J.O. Planning and control of autonomous mobile robots for intralogistics: Literature review and research agenda. *Eur. J. Oper. Res.* **2021**, *294*, 405–426. [CrossRef]
5. Dosoftei, C.-C.; Popovici, A.-T.; Sacaleanu, P.-R.; Gherghel, P.-M.; Budaciu, C. Hardware in the Loop Topology for an Omnidirectional Mobile Robot Using Matlab in a Robot Operating System Environment. *Symmetry* **2021**, *13*, 969. [CrossRef]
6. Sariyildiz, E.; Oboe, R.; Ohnishi, K. Disturbance Observer-Based Robust Control and Its Applications: 35th Anniversary Overview. *IEEE Trans. Ind. Electron.* **2020**, *67*, 2042–2053. [CrossRef]
7. Autonomous 2D SLAM and 3D Mapping of an Environment Using a Single 2D LIDAR and ROS | IEEE Conference Publication | IEEE Xplore. Available online: https://ieeexplore.ieee.org/document/8215333 (accessed on 8 February 2023).
8. Balasuriya, B.L.E.A.; Chathuranga, B.A.H.; Jayasundara, B.H.M.D.; Napagoda, N.R.A.C.; Kumarawadu, S.P.; Chandima, D.P.; Jayasekara, A.G.B.P. Outdoor robot navigation using Gmapping based SLAM algorithm. In Proceedings of the 2016 Moratuwa Engineering Research Conference (MERCon), Moratuwa, Sri Lanka, 5–6 April 2016; pp. 403–408.
9. Szayer, G. Kinematic and Dynamic Limits of Holonomic Mobile Robots. Ph.D. Thesis, Budapest University of Technology and Economics (Hungary), Budapest, Hungary, 2018.
10. Kanjanawanishkul, K. Omnidirectional wheeled mobile robots: Wheel types and practical applications. *Int. J. Adv. Mechatron. Syst.* **2015**, *6*, 289. [CrossRef]
11. Fahmizal; Kuo, C.-H. Trajectory and heading tracking of a mecanum wheeled robot using fuzzy logic control. In Proceedings of the 2016 International Conference on Instrumentation, Control and Automation (ICA), Bandung, Indonesia, 29–31 August 2016; pp. 54–59.
12. Chevallereau, C.; Khalil, W. A new method for the solution of the inverse kinematics of redundant robots. In Proceedings of the 1988 IEEE International Conference on Robotics and Automation, Philadelphia, PA, USA, 24–29 April 1988; Volume 1, pp. 37–42.
13. Tutunji, T.A. DC Motor Identification using Impulse Response Data. In Proceedings of the EUROCON 2005—The International Conference on "Computer as a Tool", Belgrade, Serbia, 21–24 November 2005; IEEE: Belgrade, Serbia, 2005; pp. 1734–1736.
14. Ljung, L.; Glover, K. Frequency domain versus time domain methods in system identification. *Automatica* **1981**, *17*, 71–86. [CrossRef]

15. You, S.H.; Bonn, K.; Kim, D.S.; Kim, S.-K. Cascade-Type Pole-Zero Cancellation Output Voltage Regulator for DC/DC Boost Converters. *Energies* **2021**, *14*, 3824. [CrossRef]
16. Yan, M.-T.; Shiu, Y.-J. Theory and application of a combined feedback–feedforward control and disturbance observer in linear motor drive wire-EDM machines. *Int. J. Mach. Tools Manuf.* **2008**, *48*, 388–401. [CrossRef]
17. Burke, J.L.; Murphy, R.R. Human-robot interaction in USAR technical search: Two heads are better than one. In Proceedings of the RO-MAN 2004. 13th IEEE International Workshop on Robot and Human Interactive Communication (IEEE Catalog No.04TH8759), Kurashiki, Japan, 22–22 September 2004; IEEE: Kurashiki, Japan, 2004; pp. 307–312.
18. Agarwal, J.; Parmar, G.; Gupta, R. Application of Sine Cosine Algorithm in Optimal Control of DC Motor and Robustness Analysis. *Wulfenia* **2017**, *24*, 77–95.
19. Kam, H.R.; Lee, S.-H.; Park, T.; Kim, C.-H. RViz: A toolkit for real domain data visualization. *Telecommun. Syst.* **2015**, *60*, 337–345. [CrossRef]
20. Guizzo, E.; Ackerman, E. The TurtleBot3 Teacher [Resources_Hands On]. *IEEE Spectr.* **2017**, *54*, 19–20. [CrossRef]
21. Joshi, R.; Bhaiya, D.; Purkayastha, A.; Patil, S.; Deshpande, A. Simultaneous Navigator for Autonomous Identification and Localization Robot. In Proceedings of the 2021 IEEE Region 10 Symposium (TENSYMP), Jeju, Republic of Korea, 23–25 August 2021; IEEE: Jeju, Republic of Korea, 2021; pp. 1–6.
22. Tam, N.D. The Implementation of Particle Filter Method in ROS for Localization. Bachelor's Thesis, Vietnamese-German University, Bến Cát, Vietnam, 2017.
23. Sasiadek, J.Z.; Hartana, P. Sensor data fusion using Kalman filter. In Proceedings of the Proceedings of the Third International Conference on Information Fusion, Paris, France, 10–13 July 2000; Volume 2.
24. Gao, C.; Spletzer, J.R. On-line calibration of multiple LIDARs on a mobile vehicle platform. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010; pp. 279–284.
25. Ballardini, A.; Fontana, S.; Furlan, A.; Sorrenti, D. ira_laser_tools: A ROS LaserScan manipulation toolbox. *arXiv* **2014**. [CrossRef]
26. dos Reis, W.P.N.; da Silva, G.J.; Junior, O.M.; Vivaldini, K.C.T. An extended analysis on tuning the parameters of Adaptive Monte Carlo Localization ROS package in an automated guided vehicle. *Int. J. Adv. Manuf. Technol.* **2021**, *117*, 1975–1995. [CrossRef]
27. Korkmaz, M.; Durdu, A. Comparison of optimal path planning algorithms. In Proceedings of the 2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, 20–24 February 2018; pp. 255–258.
28. Guimarães, R.L.; de Oliveira, A.S.; Fabro, J.A.; Becker, T.; Brenner, V.A. ROS Navigation: Concepts and Tutorial. In *Robot Operating System (ROS): The Complete Reference (Volume 1)*; Koubaa, A., Ed.; Studies in Computational Intelligence; Springer International Publishing: Cham, Switzerland, 2016; pp. 121–160. ISBN 978-3-319-26054-9.
29. Tianyu, L.; Ruixin, Y.; Guangrui, W.; Lei, S. Local Path Planning Algorithm for Blind-guiding Robot Based on Improved DWA Algorithm. In Proceedings of the 2019 Chinese Control And Decision Conference (CCDC), Nanchang, China, 3–5 June 2019; pp. 6169–6173.

sensors

MDPI

Article

Parametric Dynamic Distributed Containment Control of Continuous-Time Linear Multi-Agent Systems with Specified Convergence Speed

Fei Yan, Siyi Feng, Xiangbiao Liu and Tao Feng *

College of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756, China
* Correspondence: taof@swjtu.edu.cn

Abstract: This paper focuses on the distributed containment control of continuous-time linear multi-agent systems (MASs) with multiple leaders over fixed topology. A parametric dynamic compensated distributed control protocol is proposed in which both the information from the observer in the virtual layer and actual adjacent agents are employed. The necessary and sufficient conditions of the distributed containment control are derived based on the standard linear quadratic regulator (LQR). On this basis, the dominant poles are configured by using the modified linear quadratic regulator (MLQR) optimal control and Geršgorin's circle criterion, hence the containment control with specified convergence speed of the MAS is achieved. Another main advantage of the proposed design is, in the case of virtual layer failure, by adjusting parameters the dynamic control protocol reduces to static, and the convergence speed can still be specified through the dominant pole assignment method combined with inverse optimal control. Finally, typical numerical examples are presented to demonstrate the effectiveness of theoretical results.

Keywords: continuous-time MAS; containment control; dominant poles assignment; convergence speed

Citation: Yan, F.; Feng, S.; Liu, X.; Feng, T. Parametric Dynamic Distributed Containment Control of Continuous-Time Linear Multi-Agent Systems with Specified Convergence Speed. *Sensors* **2023**, *23*, 2696. https://doi.org/10.3390/s23052696

Academic Editor: Dawid Połap

Received: 29 January 2023
Revised: 21 February 2023
Accepted: 24 February 2023
Published: 1 March 2023

1. Introduction

Since the past few decades, the distributed coordination of MASs has sparked a surge in interest from a wide variety of scientific fields for its possibilities of the extensive application seen in the Refs. [1–4]. As one of the most essential and fundamental problems in cooperative control of MASs, consensus control is to bring all agents into alignment on a feature or a state by designing appropriate distributed protocols [5], which has achieved a series of results [6–10]. Consensus studies mostly assume that there is no or only one leader in the MAS. However, in practical applications, MAS networks with multiple leaders are more typical. Then, the containment control arises, where the followers enter into a given geometric space spanned by the leaders.

There have been plenty of valuable outcomes. In the Ref. [11], a hybrid containment control algorithm was proposed to drive the followers into the convex hull spanned by the leaders. A second-order multi-agent containment control with random switching interconnection topology was considered in the Ref. [12]. In the Ref. [13], the robust containment problem with time-variant uncertainties was solved by an adaptive protocol. In the Ref. [14], the necessary and sufficient condition of containment control with time-delay was proved. In the Ref. [15], the fastest containment control of a discrete-time MAS was achieved under static protocol control, but the convergence speed of the system could not be adjusted arbitrarily. It can be seen that great efforts have been put into the system stability and static properties of MASs in containment control, while the adaptability and dynamic properties of the system have been little discussed.

In addition to the design of the distributed control protocol based on consensus, convergence speed is also an important indicator, describing how fast the agents reach

an agreement, which is one of the most important research challenges in the design of distributed consensus algorithms for MASs. Many researchers in the Refs. [16–21] control the convergence speed by transforming or reconstructing topological structures since the network connectedness is critical in guaranteeing the convergence of consensus algorithms. However, this method is inapplicable for the MASs with fixed topology. According to the Ref. [3], the minimal non-zero eigenvalue of the Laplacian matrix can determine the convergence speed. It was figured out that in the Ref. [22] the convergence speed can be adjusted by configuring the closed-loop poles of the MAS. In the Ref. [23], the cooperative output regulation problem of linear MASs was solved by designing a distributed dynamic full information feedback control law with the distributed observer. Meanwhile, under the presented dynamic protocol, the idea of dynamic performance tuning by configuring poles has been proposed. However, the method in the Ref. [23] will not be able to achieve cooperative control of the system and adjustment of the dynamic performance, if the observer fails.

In this paper, we aim to propose a new distributed control protocol, which reduces the constraints of communication topology and provides better cooperative control performance. Moreover, the specified convergence speed of containment control will be achieved even if the distributed observer becomes invalid. The main contributions of the paper are reflected as follows:

(1) For a continuous-time linear MAS with multiple leaders over a directed topology, a new parametric dynamic compensated distributed control protocol for containment control is proposed. Compared with the containment control strategy in the Refs. [13,15], the co-states of agents in the virtual layer are introduced to reduce the limitation of the communication topology on the dynamic performance of the MAS. Compared with the protocol designed in the Ref. [23], the information about the actual state from the sensors of the physical layer is added, which can promote compatibility of the MAS and the adjustment to dynamic performance. Necessary and sufficient conditions for containment control are given based on the standard LQR design.

(2) Compared with the research of containment control in the Refs. [12,13,15], we focus on the arbitrary adjustment of the dynamic performance of the system. The accurate dominant pole configuration of the global closed-loop error system through the MLQR method and the Geršgorin's circle criterion is used to achieve containment control with specified convergence speed. For the case where the virtual layer fails, the dynamic protocol will reduce to a static protocol based on the cooperative information from the physical sensors. Meanwhile, the convergence speed is specified by configuring the dominant poles of the resulting closed-loop error system combined with the inverse optimal regulator.

In the course of our research, we employed knowledge related to graph theory. A multi-agent system (MAS) can be abstracted in the form of a directed weighted graph $\mathcal{G}$ with N nodes $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$. The adjacency matrix is denoted by $\mathcal{A} = \left[a_{ij}\right] \in \mathbb{R}^{N\times N}$, if the information flows from node j to i then $a_{ij} > 0$, otherwise, $a_{ij} = 0$, $i, j \in \mathcal{N}$, $\mathcal{N} = \{1, 2, \ldots, N\}$. The set of neighbors of node i is denoted by $\mathcal{N}_i$. Define the in-degree matrix as $\mathcal{D} = diag\{d_1, d_2, \ldots, d_N\}, d_i = \sum_{j\in\mathcal{N}_i} a_{ij}$ and the Laplacian matrix as $\mathcal{L} = \mathcal{D} - \mathcal{A}$. The adjacency matrix $\mathcal{A}$ of an undirected graph must be symmetric, where $a_{ij} = a_{ji}$. When there exists a directed path from node i to every other node in the directed graph $\mathcal{G}$, then $\mathcal{G}$ is said to have a spanning tree.

The remainder of the paper is organized as below. In Section 2, the main results will be proposed. Firstly, we introduce the parametric dynamic compensated distributed protocol and propose necessary and sufficient conditions for containment control over the directed graph. The specified convergence speed of the containment of agents is guaranteed by using the poles assignment technique for cases of the observers which are working and invalid. Section 3 gives three numerical examples to verify the developed theoretical results. Conclusions are given in Section 4.

Notations: $\mathbb{R}^{m\times n}$ denotes the $m \times n$ real matrix space. $\mathbf{0}_{m\times n}$ describes the zero matrix in $\mathbb{R}^{m\times n}$. I_n represents the n dimensional identity matrix in $\mathbb{R}^{n\times n}$. A^T denotes the

transposition of matrix A, A^H denotes the conjugate transposition of matrix A, and $A > \mathbf{0}$ ($A \geq \mathbf{0}$) means matrix A is positive definite (semi-definite). $\mathcal{N}_i$ denotes the ith node of the node set, $\mathcal{N}_\mathcal{D}$ denotes the leader nodes, and $\mathcal{N}_\mathcal{F}$ denotes the follower nodes; moreover, $\mathcal{N}_i = \mathcal{N}_\mathcal{D} \cup \mathcal{N}_\mathcal{F}$. A matrix is Hurwitz if all of its eigenvalues have negative real parts.

2. Design of Dynamic Distributed Containment Control Protocol

2.1. Dynamic Containment Control

A continuous-time linear MAS with $N + M$ nodes can be described by:

$$\dot{x}_i = Ax_i + Bu_i \quad \forall i \in \mathcal{N}, \tag{1}$$

where $x_i \in \mathbb{R}^n$, $u_i \in \mathbb{R}^m$, $\mathcal{N} = \{1, 2, \ldots, N + M\}$ is node set, and the matrices A, B are the system matrix and control input matrix, respectively.

Assumption 1. *The matrix pair (A, B) is controllable.*

Under Assumption 1, consider that there are N followers, which can be described by a directed graph while M leaders do not receive information from any other agent. Then the follower set and leader set are captured, which are, respectively, $\mathcal{F} \triangleq \{1, \ldots, N\}$ and $\mathcal{D} \triangleq \{N + 1, \ldots, N + M\}$.

The dynamics of each leader and follower are:

$$\dot{x}_i = Ax_i \quad \forall i \in \mathcal{D} \tag{2}$$

$$\dot{x}_j = Ax_j + Bu_j \quad \forall j \in \mathcal{F}, \tag{3}$$

where x_i is the state vector of leaders, x_j is the state vector of followers, and u_j is the control input vector of agent j.

The compact form of (2) and (3) can be written as:

$$\dot{x}_l = (I_M \otimes A)x_l \tag{4}$$

$$\dot{x_f} = (I_N \otimes A)x_f + (I_N \otimes B)u, \tag{5}$$

where $x_f = (x_1^T, x_2^T, \ldots, x_N^T)^T$ is the global state vector of followers, $x_l = (x_{N+1}^T, x_{N+2}^T, \ldots, x_{N+M}^T)^T$ is the global state vector of leaders, and $u = (u_1^T, u_2^T, \ldots, u_N^T)^T$ is the global control input vector.

Assumption 2. *For each follower in the MAS, there exists at least one leader that has a directed path to it.*

The communication topology graph of the MAS (1) is represented by $\mathcal{G}$, and the structural characteristics of $\mathcal{G}$ can be described by a Laplacian matrix $\mathcal{L}$. Since leaders are independent of each other, $\mathcal{L}$ can be written as a block matrix:

$$\mathcal{L} = \begin{bmatrix} \mathcal{L}_f & \mathcal{L}_l \\ \mathbf{0}_{M\times N} & \mathbf{0}_{M\times M} \end{bmatrix}, \tag{6}$$

where $\mathcal{L}_f \in \mathbb{R}^{N\times N}$ represents the information transmission situation related to followers, and $\mathcal{L}_l \in \mathbb{R}^{N\times M}$ represents the relation to leaders.

From (4), the following equation can be obtained by multiplying $(\mathcal{L}_f^{-1}\mathcal{L}_l) \otimes I_n$ to both sides:

$$[(\mathcal{L}_f^{-1}\mathcal{L}_l) \otimes I_n]\dot{x}_l = (I_M \otimes A)[(\mathcal{L}_f^{-1}\mathcal{L}_l) \otimes I_n]x_l. \tag{7}$$

Under Assumption 2, all eigenvalues of $\mathcal{L}_f$ have positive real parts, each entry of $-(\mathcal{L}_f^{-1}\mathcal{L}_l)$ is nonnegative, and each row of $-(\mathcal{L}_f^{-1}\mathcal{L}_l)$ has a sum of 1 [24], thus the linear combination of x_l as follows can be referred to as the convex hull spanned by each element of x_l.

$$-[(\mathcal{L}_f^{-1}\mathcal{L}_l) \otimes I_n]x_l \tag{8}$$

Lemma 1 ([25]). *When $x_f \to -[(\mathcal{L}_f^{-1}\mathcal{L}_l) \otimes I_n]x_l$, the states of all followers in the MAS will move into the convex hull spanned by leaders, hence the containment control is achieved.*

2.2. Parametric Dynamic Compensated Distributed Containment Control

Consider the following dynamic distributed control protocol with parameters:

$$u_i = -cK\left[\omega_i(x_i - v_i) + \sum_{j\in\mathcal{N}_i} a_{ij}(x_i - x_j)\right], \quad i \in \mathcal{F}, \tag{9}$$

where v_i is the corresponding co-state for each follower which is generated by the following distributed dynamic compensator:

$$\dot{v}_i = Av_i - rW\left[\sum_{j\in\mathcal{N}_\mathcal{F}} a_{ij}(v_i - v_j) + \sum_{h\in\mathcal{N}_\mathcal{D}} a_{ih}(v_i - x_h)\right], \tag{10}$$

where the weight $\omega_i > 0$, $K \in \mathbb{R}^{m\times n}$, and $W \in \mathbb{R}^{n\times n}$ are the feedback control gain matrices, the coupling coefficients $c > 0$, $r > 0$, v_i is the corresponding co-state that the agent goes to track, x_h represents the state of a particular leader node. The compact form of (9) and (10) can be written as:

$$u = -c[(\Omega \otimes K)(x_f - v_f) + (\mathcal{L}_f \otimes K)x_f + (\mathcal{L}_l \otimes K)x_l] \tag{11}$$

$$\dot{v}_f = (I_N \otimes A)v_f - r[(\mathcal{L}_f \otimes W)v_f + (\mathcal{L}_l \otimes W)x_l], \tag{12}$$

where

$$\Omega = \mathrm{diag}\{\omega_1, \omega_2, \ldots, \omega_N\} \tag{13}$$

and v_f represents the co-state of each follower.

The proposed dynamic compensated distributed control law drives co-states into the convex hull, meanwhile each follower is able to follow the corresponding co-state, and thus dynamic containment control of the MAS can be achieved.

Remark 1. *In the control protocol* (9)*, the first term implements followers tracking of the co-states; the second term achieves cooperative control by introducing actual relative information from physical sensors between agents. In a practical application scenario, such as a number of vehicles departing from different locations are required to drive into the safety zone formed by multiple mobile escort vehicles. At this time, the parameters of the cooperative control section can be regulated to achieve a special requirement of vehicles assembling into groups first and then driving into the safety zone, which increases the overall strike resistance of the convoy.*

Denote the error between the convex hull and co-states as δ, the error between each follower and the corresponding co-state as θ:

$$\delta = v_f + (\mathcal{L}_f^{-1}\mathcal{L}_l \otimes I_n)x_l \tag{14}$$

$$\theta = x_f - v_f. \tag{15}$$

According to (7), the following equation holds:

$$(\mathcal{L}_f^{-1}\mathcal{L}_l \otimes I_n)\dot{x}_l = (\mathcal{L}_f^{-1}\mathcal{L}_l \otimes A)x_l,$$

then $\dot{v}_f$ can be written as:

$$\dot{v}_f = (I_N \otimes A)v_f - r(\mathcal{L}_f \otimes W)[v_f + (\mathcal{L}_f^{-1}\mathcal{L}_l \otimes I_n)x_l]. \tag{16}$$

Adding (7) and (16), we obtain:

$$\dot{\delta} = \dot{v}_f + (\mathcal{L}_f^{-1}\mathcal{L}_l \otimes I_n)\dot{x}_l$$

$$\dot{\delta} = [(I_N \otimes A) - r(\mathcal{L}_f \otimes W)]\delta. \tag{17}$$

Similarly, $\dot{\theta}$ can be calculated as:

$$\dot{\theta} = \dot{x}_f - \dot{v}_f = (I_N \otimes A)\theta - c[(\Omega \otimes BK) + (\mathcal{L}_f \otimes BK)]\theta - [c(\mathcal{L}_f \otimes BK) - r(\mathcal{L}_f \otimes W)]\delta. \tag{18}$$

Combining (17) and (18) yields the global closed-loop error system:

$$\begin{bmatrix}\dot{\delta}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\Xi_{11} & \mathbf{0}\\ \Xi_{21} & \Xi_{22}\end{bmatrix}\begin{bmatrix}\delta\\ \theta\end{bmatrix}, \tag{19}$$

where

$$\begin{aligned}\Xi_{11} &= (I_N \otimes A) - r(\mathcal{L}_f \otimes W)\\ \Xi_{21} &= -c(\mathcal{L}_f \otimes BK) + r(\mathcal{L}_f \otimes W)\\ \Xi_{22} &= (I_N \otimes A) - c(\mathcal{L}_f + \Omega) \otimes (BK).\end{aligned}$$

Denote the eigenvalues of $\mathcal{L}_f$ and $(\mathcal{L}_f + \Omega)$ as λ_i, χ_i, respectively. Note that there exists nonsingular matrices Φ and Ψ such that:

$$(\Phi \otimes I_n)^{-1}\Xi_{11}(\Phi \otimes I_n) = \begin{bmatrix}A - r\lambda_1 W & & & \\ * & A - r\lambda_2 W & & \\ \vdots & \vdots & \ddots & \\ * & * & \cdots & A - r\lambda_N W\end{bmatrix} \tag{20}$$

$$(\Psi \otimes I_n)^{-1}\Xi_{22}(\Psi \otimes I_n) = \begin{bmatrix}A - c\chi_1 BK & & & \\ * & A - c\chi_2 BK & & \\ \vdots & \vdots & \ddots & \\ * & * & \cdots & A - c\chi_N BK\end{bmatrix}. \tag{21}$$

Let $\tilde{\delta} = (\Phi \otimes I_n)^{-1}\delta$, $\tilde{\theta} = (\Psi \otimes I_n)^{-1}\theta$, then the error system (19) can be transformed into the following form:

$$\begin{bmatrix}\dot{\tilde{\delta}}\\ \dot{\tilde{\theta}}\end{bmatrix} = \begin{bmatrix}\hat{\Xi}_{11} & \mathbf{0}\\ \hat{\Xi}_{21} & \hat{\Xi}_{22}\end{bmatrix}\begin{bmatrix}\tilde{\delta}\\ \tilde{\theta}\end{bmatrix}, \tag{22}$$

where $\hat{\Xi}_{11}$, $\hat{\Xi}_{22}$ are shown as (20) and (21), respectively.

Theorem 1. *Under Assumptions 1 and 2, the containment control can be achieved by the dynamic distributed protocol* (9) *if and only if the matrices*

$$\begin{aligned}A - r\lambda_i W, &\quad i = 1, \ldots, N\\ A - c\chi_j BK, &\quad j = 1, \ldots, N\end{aligned}$$

are Hurwitz.

Proof of Theorem 1. The error system (19) is asymptotically stable, that is, (22) is stable, if and only if the following $2N$ matrices

$$A - r\lambda_1 W, \ldots, A - r\lambda_N W, A - c\chi_1 BK, \ldots, A - c\chi_N BK$$

are Hurwitz. Stability of the error system (19) and (22) indicates that $\|\delta\| \to 0$, $\|\theta\| \to 0$, which means the error between the convex hull and co-states δ, the error between followers and co-states θ tend to 0. It implies that containment control is achieved. □

In the following, we will prove the appropriate choice of the coupling gains c and r.

Theorem 2. *Under Assumptions 1 and 2, using the control protocol* (9)*, where* $W = R^{-1}P_1$ *in which* P_1 *is a symmetric positive definite matrix and the solution of the following Riccati equation:*

$$P_1 A + A^T P_1 - P_1 R^{-1} P_1 + Q = \mathbf{0}. \tag{23}$$

Similarly, $K = R^{-1}B^T P_2$ *in which* P_2 *is the solution of* $P_2 A + A^T P_2 - P_2 B R^{-1} B^T P_2 + Q = \mathbf{0}$*. If the coupling gains c and r satisfy:*

$$r > \frac{1}{2\lambda_{\min}}, \quad c > \frac{1}{2\chi_{\min}}, \tag{24}$$

where $\lambda_{\min} = \min\{\mathrm{Re}(\lambda_1), \mathrm{Re}(\lambda_2), \ldots, \mathrm{Re}(\lambda_N)\}$ *and* $\chi_{\min} = \min\{\mathrm{Re}(\chi_1), \mathrm{Re}(\chi_2), \ldots, \mathrm{Re}(\chi_N)\}$*, then the global error system* (19) *is asymptotically stable, that is, the containment control is achieved.*

Proof of Theorem 2. According to Theorem 1, it is sufficient to make the subsystems $(A - r\lambda_i W)$ and $(A - c\chi_i BK)$ asymptotically stable. Take the subsystem $(A - r\lambda_i W)$ as an example, constructing the Lyapunov function:

$$V(x) = \tilde{\delta}_i^H P_1 \tilde{\delta}_i \quad i = 1, 2, \ldots, N,$$

where P_1 is the symmetric positive definite matrix and the solution of (23).

Taking the derivation of the function $V(x)$ with respect to time yields:

$$\dot{V} = \dot{\tilde{\delta}}_i^H P_1 \tilde{\delta}_i + \tilde{\delta}_i^H P_1 \dot{\tilde{\delta}}_i = \tilde{\delta}_i^H P_1 (A - r\lambda_i W)\tilde{\delta}_i + \tilde{\delta}_i^H (A - r\lambda_i W)^H P_1 \tilde{\delta}_i = \tilde{\delta}_i^H \Big[P_1 (A - r\lambda_i W) + (A - r\lambda_i W)^H P_1 \Big] \tilde{\delta}_i. \tag{25}$$

Replacing $R^{-1}P_1$ in (23) with the feedback gain matrix W, we have:

$$A^T P_1 + P_1 A = W^T R W - Q.$$

Similarly, the Riccati equation corresponding to subsystem $(A - r\lambda_i W)$ has the following form:

$$(A - r\lambda_i W)^H P_1 + P_1 (A - r\lambda_i W) = -Q + [1 - 2r\mathrm{Re}(\lambda_i)] W^T R W.$$

Bringing into the equation (25), we can obtain:

$$\dot{V} = \tilde{\delta}_i^H \Big\{ -Q + [1 - 2r\mathrm{Re}(\lambda_i)] W^T R W \Big\} \tilde{\delta}_i. \tag{26}$$

From the Lyapunov theorem of asymptotic stability, if the subsystem $(A - r\lambda_i W)$ is to be asymptotically stable, then $V(x)$ needs to satisfy $\dot{V} < 0$. Due to Q being a positive definite matrix and R being a symmetric positive definite matrix, we can obtain that $-Q < \mathbf{0}$ and $W^T R W \geq \mathbf{0}$.

To ensure $\dot{V} < 0$, it is only needed to satisfy $1 - 2r\mathrm{Re}(\lambda_i) < 0$. That is when the coupling gain r satisfies:

$$r > \frac{1}{2\lambda_{\min}},$$

where $\lambda_{\min} = \min\{\mathrm{Re}(\lambda_1), \mathrm{Re}(\lambda_2), \ldots, \mathrm{Re}(\lambda_N)\}$. Then the subsystem $(A - r\lambda_i W)$ is asymptotically stable with $W = R^{-1}P_1$.

Similarly, when the coupling gain c satisfies:

$$c > \frac{1}{2\chi_{\min}}.$$

The subsystem $(A - c\chi_i BK)$ is asymptotically stable with $K = R^{-1}B^T P_2$.

Therefore, when the global closed-loop error system is asymptotically stable, the MAS can achieve containment control. The proof is completed. □

2.3. Dynamic Distributed Containment Control with Specified Convergence Speed

It is known that the convergence speed is determined by the closed-loop poles that are closest to the imaginary axis. Therefore, the convergence speed will be specified by configuring the dominant poles of the global close-loop error system on dynamic containment control.

In the global closed-loop error system (22), the subsystems $(A - r\lambda_i W)$ and $(A - c\chi_i BK)$ are located in $\hat{\Xi}_{11}$ and $\hat{\Xi}_{22}$, respectively, so it is sufficient to design for these two blocks.

The eigenvalues of Ξ_{11} ($\hat{\Xi}_{11}$) without parameters ω_i are designed as non-dominated poles according to the MLQR [22] optimal control scheme.

Let $\overline{A} = A + \sigma I_n$, where there is the error system:

$$\dot{\delta} = \Big[(I_N \otimes A) - r(\mathcal{L}_f \otimes W)\Big]\delta, \tag{27}$$

the subsystem is converted to

$$\overline{A} - r\lambda_i W = A + \sigma I_n - r\lambda_i W.$$

It is shown that when all eigenvalues lie to the left of the complex plane $-\sigma$, the error system (27) will be asymptotically stable and converge at a speed σ.

Therefore, if the value of σ is large enough, the poles of the virtual layer will move away from the imaginary axis and become non-dominant poles. At this time, the dominant poles of the closed-loop error system will be determined by Ξ_{22}, that is, subsystem $(A - c\chi_i BK)$.

For Ξ_{22} ($\hat{\Xi}_{22}$), the parameters ω_i are designed based on the Geršgorin circle theorem to configure the specified dominant poles.

Lemma 2. *Under Assumption 2, for the appropriate choice of $\omega_i > 0$, all eigenvalues of the matrix $(\mathcal{L}_f + \Omega)$ are distinct and positive. Then, let $0 < \chi_1 < \chi_2 < \cdots < \chi_N$, $\forall \mu > 0$, where the eigenvalues are all real and the ratio χ_N/χ_1 satisfies:*

$$\frac{\chi_N}{\chi_1} < \frac{1+\mu}{1-\mu}. \tag{28}$$

Proof of Lemma 2. As shown in Figure 1, according to the Geršgorin circle theorem, the eigenvalue χ_i lies in the ith Geršgorin circle $\mathbb{Q}_i = \{d \mid |d - O_i| \le r_i\}$,where:

$$O_i = \sum_{j \in \mathcal{N}_i} a_{ij} + \omega_i, \quad r_i = \sum_{j \in \mathcal{N}_i} a_{ij} \quad i = 1, 2, \ldots, N. \tag{29}$$

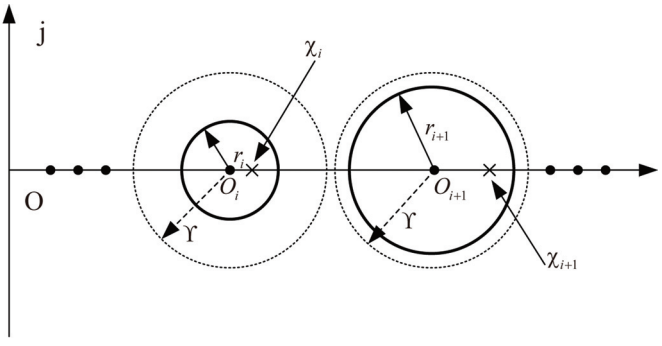


Figure 1. Gerschgorin circles.

It is known that, all eigenvalues χ_i of $(\mathcal{L}_f + \Omega)$ are distinct if all Geršgorin circles are separated, that is, $\mathbb{Q}_i \cap \mathbb{Q}_j = \varnothing, i \neq j$, which is equivalent to

$$|O_{i+1} - O_i| > |r_i + r_{r+1}| \quad i = 1, 2, \ldots, N.$$

For constants $\Upsilon > \max_{i\in\mathcal{N}}\{r_i\}$ and any $\eta > 0$, there always exists $\omega_i > 0$ such that the following equation holds:

$$O_i = \sum_{j\in\mathcal{N}_i} a_{ij} + \omega_i = \eta + 2i\Upsilon. \tag{30}$$

That is, ω_i satisfies the following equation:

$$\omega_i = \eta + 2i\Upsilon - \sum_{j\in\mathcal{N}_i} a_{ij}. \tag{31}$$

Obviously, $\omega_i > 0$. Using (30), then one has:

$$|O_{i+1} - O_i| = 2\Upsilon > |r_i + r_{r+1}| \quad i = 1, 2, \ldots, N, \quad O_1 = \eta + 2\Upsilon > \Upsilon > 0. \tag{32}$$

Thus all Geršgorin circles are separated. At this point, there must be at least one eigenvalue that lies into each Geršgorin circle; moreover, the first Geršgorin circle is located in the right-half of the complex plane. Therefore, all eigenvalues χ_i of $(\mathcal{L}_f + \Omega)$ are distinct and positive. From (32), we can obtain:

$$0 < \eta + \Upsilon < \chi_1 < \eta + 3\Upsilon$$

$$\vdots$$

$$\eta + (2N-1)\Upsilon < \chi_N < \eta + (2N+1)\Upsilon,$$

thus,

$$\frac{\chi_N - \chi_1}{\chi_N + \chi_1} < \frac{N\Upsilon}{\eta + N\Upsilon}. \tag{33}$$

Denote that

$$\mu = \frac{\chi_N - \chi_1}{\chi_N + \chi_1}, \quad \eta_0 = \frac{1-\mu}{\mu} N\Upsilon.$$

Then (33) holds when and only when $\eta > \eta_0$, that is, satisfies

$$\mu > \frac{\chi_N - \chi_1}{\chi_N + \chi_1}.$$

Simplify to obtain:

$$\frac{\chi_N}{\chi_1} < \frac{1+\mu}{1-\mu}. \tag{34}$$

The proof is completed. □

Remark 2. *For the readability of the readers, we here introduce the Geršgorin circle theorem for details. The Geršgorin circle theorem is used to bind the spectrum of a square matrix. Given a $N \times N$ matrix A with entries a_{ij}, for each $i = 1, \ldots, N$, we define*

$$r_i = \sum_{j \neq i} |a_{ij}|, \quad \mathbb{Q}_i = \{d \in a_{ij} \mid |d - a_{ii}| \leq r_i\}$$

where $\mathbb{Q}_i$ are denoted the Gerschgorin circles of A. Then every eigenvalue of A lies within at least one of the Gerschgorin circles $\mathbb{Q}_i$.

We use the Geršgorin circle theorem in order to prove the following Theorem 3 and configure the dominant poles so that all the dominant poles can reach the specified position.

Theorem 3. *When the value of σ is chosen to be large enough and the poles of $(A - c\chi_i BK)$ are configured to the specified position, if the coupling gain c satisfies:*

$$c = \frac{1}{\chi_{\min}}, \tag{35}$$

then the MAS can achieve dynamic containment control at a specified convergence speed.

Proof of Theorem 3. According to Lemma 2 and its proof, the eigenvalues χ_i of $(\mathcal{L}_f + \Omega)$ are nearly the same, so $1/\chi_i$ can be approximately equal to $1/\chi_{\min}$. If the coupling gain $c = 1/\chi_{\min}$, the closed-loop dominant poles of $(A - c\chi_i BK)$ can be approximately equal to the eigenvalues of the matrix $A - BK$. Through the poles configuration, the absolute value of the real part of the conjugate eigenvalue of the matrix $A - BK$ closest to the imaginary axis is ε. At this point, part of the poles of the closed-loop system (22) are located away from the imaginary axis to the left complex plane, and others converge to $-\varepsilon$, that is, the MAS (1) achieve containment control with the given convergence speed. The proof is completed. □

Remark 3. *The MLQR [22] optimal control scheme can be used on poles' configuration for the whole closed-loop error system* (22)*, and the convergence speed of the MAS can also be adjusted. However, for subsystem $(A - c\chi_i BK)$, a large overshoot occurs. Therefore the dominant pole configuration is applied to it by using Lemma 2.*

2.4. Regulation of the Convergence Speed in Case of Virtual Layer Failure

Consider a practical case where the observer fails in some agents, and where the information of co-states cannot be transmitted. At this time, the connectivity of the virtual layer topology is not guaranteed. However, the actual adjacent information in the physical layer can still be collected by sensors. Therefore, the control protocol can only use the adjacent information $x_i - x_j$ feedback to the system of the physical layer.

It can be regarded as the virtual layer subsystem being totally disabled. The dynamic distributed control protocol (9) reduces to a static form [26].

Under Assumption 1, let us investigate the containment control scheme and dynamic performance of the MAS (1) in Section 2.1. Without loss of generality, the matrices A and B are set as the following forms:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ I_m \end{bmatrix}. \tag{36}$$

Similarly, A, B, and the control gain matrix $\widetilde{K}$ satisfy the Riccati equation

$$A^T P + PA - PBR^{-1}B^T P + Q = \mathbf{0}. \tag{37}$$

Give a feedback control law

$$u_i = -\widetilde{K} \sum_{j \in \mathcal{F} \cup \mathcal{D}} a_{ij}(x_i - x_j), \quad \widetilde{K} = \underline{\alpha}[F \quad I] \tag{38}$$

$$u = -(\mathcal{L}_f \otimes \widetilde{K})[x_f + (\mathcal{L}_f{}^{-1}\mathcal{L}_l \otimes I_n)x_l]. \tag{39}$$

Similarly, with Section 2.2, the error between followers and convex hull spanned by leaders in MAS (1) is

$$\zeta = x_f + (\mathcal{L}_f^{-1}\mathcal{L}_l \otimes I_n)x_l. \tag{40}$$

$$\dot{\zeta} = [(I_N \otimes A) - (\mathcal{L}_f \otimes B\widetilde{K})]\zeta = \Xi\zeta. \tag{41}$$

Note that there exists a nonsingular matrix $\widetilde{\Phi}$ such that:

$$(\widetilde{\Phi} \otimes I_n)^{-1}\Xi(\widetilde{\Phi} \otimes I_n) = \begin{bmatrix} A - \lambda_1 B\widetilde{K} & & & \\ * & A - \lambda_2 B\widetilde{K} & & \\ \vdots & \vdots & \ddots & \\ * & * & \cdots & A - \lambda_N B\widetilde{K} \end{bmatrix} \tag{42}$$

According to Theorem 1, the containment control can be achieved by the protocol (38) if and only if the matrices

$$A - \lambda_i B\widetilde{K}, \quad i = 1, \ldots, N. \tag{43}$$

are Hurwitz.

According to the algorithm in the Ref. [27], $\widetilde{K}$ can be determined through the specified poles (or the desired transient characteristics). There exists a lower limit α_0 for the value of $\underline{\alpha}$, that is, $\underline{\alpha} > \alpha_0$, while α_0 is determined by the procedure in the Ref. [26]. It is known that from Section 2.2, the convergence speed relies on the locations of eigenvalues of $A - \lambda_i B\widetilde{K}, i = 1, \ldots, N$, whose asymptotic behavior shows in the following lemma.

Lemma 3 ([28]). *Take the value of F in* (38) *such that the $n - m$ eigenvalues of $A - B\widetilde{K}$ are the specified closed-loop poles $\{d_1^*, \ldots, d_{n-m}^*\}$, then as $\underline{\alpha} \to \infty$:*

(1) $n - m$ eigenvalues of $A - \lambda_i B\widetilde{K}$ satisfy $d_i \to d_i^*, \quad i = 1, \ldots, n - m$;

(2) the rest m eigenvalues of $A - \lambda_i B\widetilde{K}$ satisfy $d_i \to -\infty, \quad i = n - m + 1, \ldots, n$.

Now the control protocol can be written as:

$$u_i = -\underline{\alpha}_i[F_i \quad I] \sum_{j \in \mathcal{F} \cup \mathcal{D}} a_{ij}(x_i - x_j), \tag{44}$$

whose compact form can be written as:

$$u = -(\mathcal{L}_f \otimes L_\alpha[F \quad I])\left\{x_f + (\mathcal{L}_f^{-1}\mathcal{L}_l \otimes I_n)x_l\right\}.$$

Theorem 4. *For a continuous-time MAS* (1) *with a given communication topology, under Assumptions 1 and 2, there exists a static control protocol* (44)*, such that the MAS (1) not only achieves containment control (2) but also achieves the specified convergence speed by configuring the $n - m$ dominant poles to the desired locations.*

Proof of Theorem 4. For (1), according to Equations (38)–(43), it is similar to the proof of Theorem 1. For (2) [26], according to Lemma 3, under the condition $\underline{\alpha} \to \infty$, the eigenvalues $\left\{d_1^*, \ldots, d_{n-m}^*\right\}$ can be viewed as the dominant poles. By configuring the $n-m$ closed-loop eigenvalues for each agent, the MAS can asymptotically achieve the desired performance (i.e., the specified convergence speed). □

Compared with the results of the research in the Ref. [23], according to Lemma 3, it is still possible to regulate the convergence speed of the MAS by configuring dominant poles based on the static control law (38), even the observer fails. However, compared with the dynamic protocol (9), this control method has disadvantages, such as complex structure and more severe overshoot in the initial phase of the response, seen in the simulation example of Section 3.3.

3. Simulation Examples

In this section, the correctness of the theoretical results and the effectiveness of the designed distributed control protocols will be verified by typical numerical examples.

In a multi-vehicle escort application scenario, where the leaders are the escort vehicles and the followers are the protected vehicles, the whole convoy can be considered as a continuous-time MAS described by Figure 2 which has M leader agents ($M = 4$) and N follower agents ($N = 6$). The system matrix A and control input matrix B are set as follows:

$$A = \begin{bmatrix} -0.01 & 0.02 \\ 0.01 & -0.012 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 10 \end{bmatrix}.$$

The MAS can be written as:

$$\dot{x}_j = Ax_j + Bu \quad \forall j \in \{1, 2, 3, 4, 5, 6\}$$

$$\dot{x}_i = Ax_i \quad \forall i \in \{7, 8, 9, 10\}.$$

The communication graph $\mathcal{G}$ is given by Figure 2, then the Laplacian matrix $\mathcal{L}$ is given by:

$$\mathcal{L} = \begin{bmatrix} \mathcal{L}_f & \mathcal{L}_l \\ \mathbf{0}_{4\times 6} & \mathbf{0}_{4\times 4} \end{bmatrix}$$

where

$$\mathcal{L}_f = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & -2 \\ -1 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & -1 & 3 \end{bmatrix}$$

and

$$\mathcal{L}_l = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & -2 \end{bmatrix}.$$

Let $Q = I_2$, $R = I$.

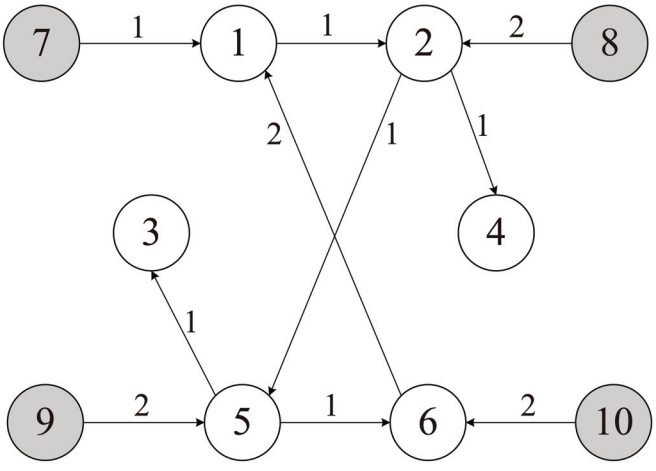


Figure 2. The communication topology of the MAS.

Set the initial position of each vehicle as follows

$$x_j = \begin{bmatrix} 12 & -6 \\ 6 & -20 \\ 2 & 20 \\ 8 & 4 \\ -6 & -12 \\ -8 & -8 \end{bmatrix}, \quad x_{vj} = \begin{bmatrix} 14 & -8 \\ 8 & -38 \\ 2 & 26 \\ 10 & 6 \\ -8 & -14 \\ -10 & -10 \end{bmatrix}, \quad x_i = \begin{bmatrix} 2 & 4 \\ -4 & 2 \\ -1 & -10 \\ 4 & -6 \end{bmatrix}$$

where x_j and x_{vj} are the initial positions of the followers, that is, the protected vehicles, in the physical and virtual layers, respectively, x_i is the initial positions of the leaders, that is, the escort vehicles which form the convex hull.

3.1. Parametric Dynamic Compensated Distributed Containment Control

According to LQR optimal control, the feedback gain matrix of the physical and virtual layer can be calculated as:

$$K = \begin{bmatrix} 0.6184 & 1.0000 \end{bmatrix}, \quad W = \begin{bmatrix} 0.9901 & 0.0148 \\ 0.0148 & 0.9883 \end{bmatrix}.$$

Let $\Omega = I_6$, where it can be calculated that $\lambda_{\min} = 1$, $\chi_{\min} = 2$. According to Theorem 2, the coupling gains can be selected as $c = 15$, $r = 0.6$.

Under the dynamic distributed control protocol (9), the convergence curve of the error system is respectively described in Figures 3 and 4.

In Figures 3 and 4, we can see that the co-states of the protected vehicles have entered into the convex hull formed by the escort vehicles, and the states of the protected vehicles have been the same as the co-states. It means that the protected vehicles have entered the convex hull, that is, the containment control has been achieved.

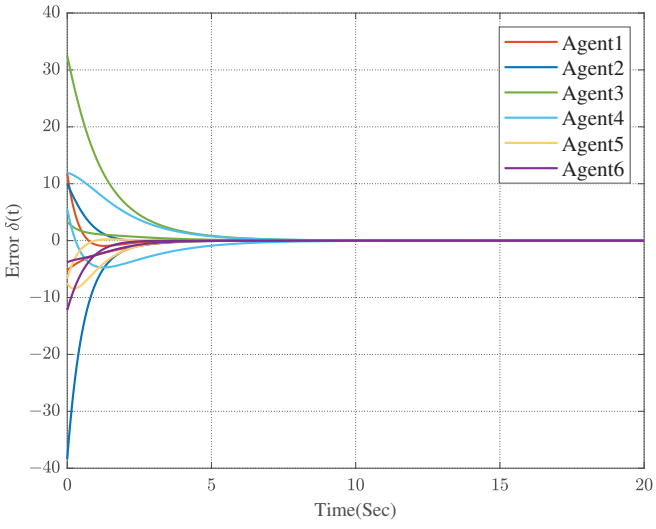


Figure 3. Error between co-states and the convex hull.

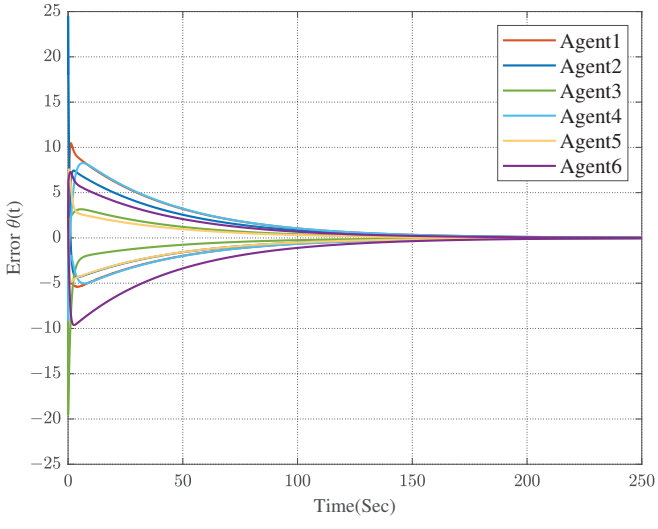


Figure 4. Error between followers and co-states.

3.2. Containment Control with Specified Convergence Speed

Now we regulate the convergence speed of the MAS (1) by using the pole configuration method proposed in Section 2.3.

Let $\sigma = 10$, then the poles of the global error system matrix (22) lie to the left of the complex plane $-\sigma = -10$.

Set the dominant poles as $\{-0.5 + 0.1j, -0.5 - 0.1j\}$, then the feedback gain matrix of the physical layer is calculated as $K = [1.2515 \quad 0.0978]$.

According to Lemma 2, we set $\omega_i = 10^4 * \text{diag}\{2, 2.1, 2.05, 3.09, 2.08, 3\}$ then according to Theorem 3, the coupling gain $c = 3.2361 \times 10^{-5}$. At this point

$$c\chi_i \approx 1, \quad A - c\chi_i BK \to A - BK,$$

which means that the poles of the physical layer subsystem are the configured poles, that is, the system converges at the specified speed.

The convergence curve of the error system and the state transition curves of the protected vehicles are shown in Figures 5 and 6, respectively.

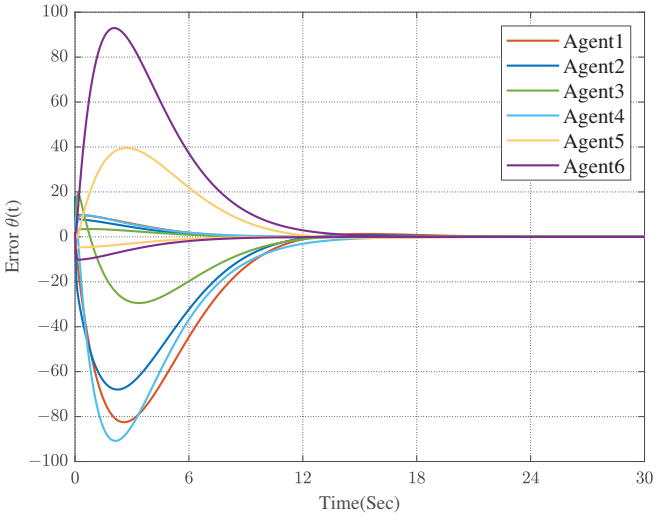


Figure 5. Convergence curves of the error between followers and co-states with specified convergence speed.

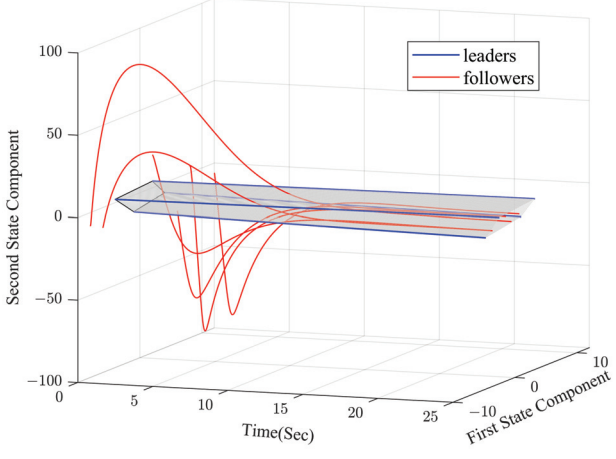


Figure 6. Agents converge into the convex hull (the two state components represent the position coordinates of each vehicle).

Comparing Figure 4 with Figure 5, we can see that the convergence time has been significantly reduced (the former is approximately 230 s while the latter is approximately 24 s). In Figure 6, we can obtain that, all of the followers, that is, the protected vehicles are stabilized into the convex hull spanned by the leaders, that is, the escort vehicles under the control of the dynamic distributed control protocol (9) with a specified convergence speed.

3.3. Containment Control in Case of Virtual Layer Failure

Let

$$A = \begin{bmatrix} -0.05 & 0.09 \\ 0.1 & 0.08 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Set the dominant poles as -1, by taking sufficiently large value of $\underline{\alpha}$, the matrix $\widetilde{K}$ is obtained $\widetilde{K} = [52.7778 \quad 5]$.

The convergence curve of the new error system ζ and the state transition curve of the protected vehicles are shown in Figures 7 and 8, respectively.

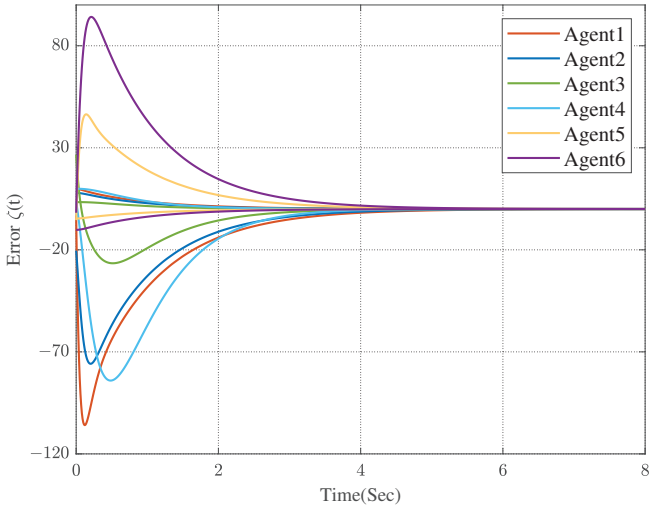


Figure 7. Error between followers and the convex hull.

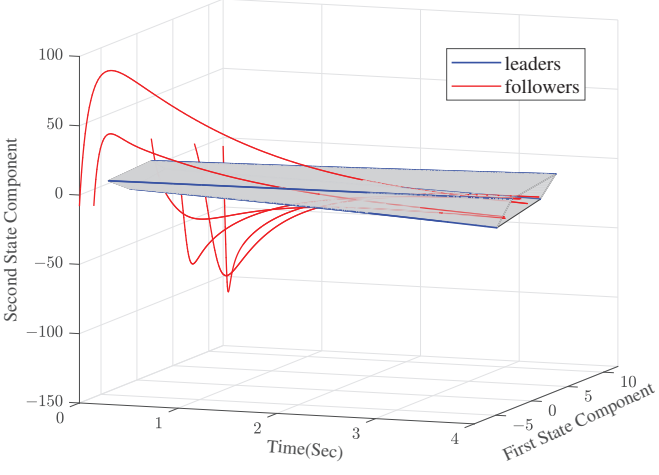


Figure 8. Followers converge into convex hull (the two state components represent the position coordinates of each vehicle).

According to Figures 7 and 8, apparently, when the observer fails, all the protected vehicles are still able to access the convex hull under the control of the static protocol (43) with a specified convergence speed.

4. Conclusions

This paper proposed a parametric dynamic compensated distributed control protocol with a co-state for each follower. For the containment control of the MAS, the necessary and sufficient conditions for taking values of the coupling gains have been derived. The dominant poles of the global closed-loop error system have been configured to specify the convergence speed of MAS. For the virtual layer subsystem, the poles have been configured as non-dominated poles by the MLQR optimal control; for the physical layer subsystem with parameters, the parameters have been designed based on the Geršgorin's circle criterion to configure the desired dominated poles. When the virtual layer fails, the protocol reduced to a static control law to achieve containment control. Moreover, combined with inverse optimal control, the convergence speed can also be specified through dominant pole configuration. Simulation examples have been given to demonstrate the effectiveness of the developed design method.

Author Contributions: Conceptualization, investigation, funding acquisition, F.Y.; writing, methodology, software, validation, visualization, S.F.; software, visualization, X.L.; methodology, resources, T.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant No. 61873215), and the Natural Science Foundation of Sichuan Province of China (Grant No. 2022NSFSC0470).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Liu, Y.; Jia, Y. An iterative learning approach to formation control of multi-agent systems. *Syst. Control. Lett.* **2012**, *61*, 148–154. [CrossRef]
2. Fax, J.A.; Murray, R.M. Information flow and cooperative control of vehicle formations. *IEEE Trans. Autom. Control* **2004**, *49*, 1465–1476. [CrossRef]
3. Olfati-Saber, R. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Trans. Autom. Control* **2005**, *51*, 401–420. [CrossRef]
4. Zhang, D.; Xie, G.; Yu, J.; Wang, L. Adaptive task assignment for multiple mobile robots via swarm intelligence approach. *Robot. Auton. Syst.* **2007**, *55*, 572–588. [CrossRef]
5. Olfati-Saber, R.; Murray, R.M. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. Autom. Control* **2004**, *49*, 1520–1533. [CrossRef]
6. Li, Y.; Fang, H.; Chen, J.; Shang, C. Distributed fault detection and isolation for multi-agent systems using relative information. In Proceedings of the 2016 American Control Conference (ACC), Boston, MA, USA, 6–8 July 2016; pp. 5939–5944.
7. Li, Z.; Ding, Z. Distributed multiobjective optimization for network resource allocation of multiagent systems. *IEEE Trans. Cybern.* **2021**, *51*, 5800–5810. [CrossRef] [PubMed]
8. Ren, W.; Beard, R.W.; Atkins, E.M. Information consensus in multivehicle cooperative control. *IEEE Control Syst.* **2007**, *27*, 71–82.
9. Ren, W.; Beard, R.W. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Trans. Autom. Control* **2005**, *50*, 655–661. [CrossRef]
10. Zheng, Y.; Ma, J.; Wang, L. Consensus of hybrid multi-agent systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 1359–1365. [CrossRef] [PubMed]
11. Cao, Y.; Stuart, D.; Ren, W.; Meng, Z. Distributed containment control for double-integrator dynamics: Algorithms and experiments. In Proceedings of the 2010 American Control Conference, Baltimore, MD, USA, 30 June–2 July; pp. 3830–3835.
12. Lou, Y.; Hong, Y. Target containment control of multi-agent systems with random switching interconnection topologies. *Automatica* **2012**, *48*, 879–885. [CrossRef]
13. Wen, G.; Hu, G.; Zuo, Z.; Zhao, Y.; Cao, J. Robust containment of uncertain linear multi-agent systems under adaptive protocols. *Int. J. Robust Nonlinear Control* **2017**, *27*, 2053–2069. [CrossRef]
14. Wang, D.; Wang, W. Necessary and sufficient conditions for containment control of multi-agent systems with time delay. *Automatica* **2019**, *103*, 418–423. [CrossRef]
15. Zhang, J.; Yan, F.; Feng, T.; Deng, T.; Zhao, Y. Fastest containment control of discrete-time multi-agent systems using static linear feedback protocol. *Inf. Sci.* **2022**, *614*, 362–373. [CrossRef]

16. Kim, Y.; Mesbahi, M. On maximizing the second smallest eigenvalue of a state-dependent graph laplacian. *IEEE Trans. Autom. Control* **2006**, *51*, 116–120. [CrossRef]
17. Kim, Y.; Mesbahi, M. Ultrafast consensus in small-world networks. In Proceedings of the American Control Conference, Portland, OR, USA, 8–10 June 2005; pp. 2371–2378.
18. Zhou, Z.; Lin, S.; Xi, Y. A fast network partition method for large-scale urban traffic networks. *J. Control Theory Appl.* **2013**, *11*, 359–366. [CrossRef]
19. Xi, Y.; Li, X. Hierarchical structure design for multi-agent consensus. *Control Theory Appl.* **2015**, *32*, 1191–1199.
20. Kim, Y. Bisection algorithm of increasing algebraic connectivity by adding an edge. *IEEE Trans. Autom. Control* **2010**, *55*, 170–174.
21. Kim, Y.; Gu, D.-W.; Postlethwaite, I. Spectral radius minimization for optimal average consensus and output feedback stabilization. *Automatica* **2009**, *45*, 1379–1386. [CrossRef]
22. Wang, J.; Cheng, D.; Hu, X. Consensus of multi-agent linear dynamic systems. *Asian J. Control* **2008**, *10*, 144–155. [CrossRef]
23. Su, Y.; Huang, J. Cooperative Output Regulation of Linear Multi-Agent Systems. *IEEE Trans. Autom. Control* **2012**, *57*, 1062–1066.
24. Meng, Z.; Ren, W.; You, Z. Distributed finite-time attitude containment control for multiple rigid bodies. *Automatica* **2010**, *46*, 2092–2099. [CrossRef]
25. Li, Z.; Ren, W.; Liu, X.; Fu, M. Distributed containment control of multi-agent systems with general linear dynamics in the presence of multiple leaders. *Int. J. Robust Nonlinear Control* **2013**, *23*, 534–547. [CrossRef]
26. Zhang, J.; Feng, T.; Zhang, H.; Wang, X. The Decoupling Cooperative Control With Dominant Poles Assignment. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 1205–1213. [CrossRef]
27. Fujii, T. A new approach to the LQ design from the viewpoint of the inverse regulator problem. *IEEE Trans. Autom. Control* **1987**, *32*, 995–1004. [CrossRef]
28. Zhang, H.; Feng, T.; Yang, G.-H.; Liang, H. Distributed Cooperative Optimal Control for Multiagent Systems on Directed Graphs: An Inverse Optimal Approach. *IEEE Trans. Cybern.* **2015**, *45*, 1315–1326. [CrossRef] [PubMed]

sensors

MDPI

Article

Adaptive Model Predictive Control for Mobile Robots with Localization Fluctuation Estimation

Jie Meng [1,2], Hanbiao Xiao [1,2], Liyu Jiang [3], Zhaozheng Hu [1,2], Liquan Jiang [4] and Ning Jiang [5,*]

[1] Intelligent Transportation Systems Research Center, Wuhan University of Technology, 1178 Heping Avenue, Wuhan 430000, China
[2] Chongqing Research Institute, Wuhan University of Technology, 598 Liangjiang Avenue, Chongqing 400000, China
[3] Hubei Institute of Measurement and Testing Technology, 2 Maodianshan Middle Road, Wuhan 430000, China
[4] The State Key Laboratory of New Textile Materials and Advanced Processing Technologies, Wuhan Textile University, 1 Yangguang Avenue, Wuhan 430000, China
[5] School of Mechanical Science and Engineering, Huazhong University of Science and Technology, 1037 Luoyu Road, Wuhan 430000, China
* Correspondence: jiangning@hust.edu.cn; Tel.: +86-155-0168-8903

Abstract: Mobile robots are widely employed in various fields to perform autonomous tasks. In dynamic scenarios, localization fluctuations are unavoidable and obvious. However, common controllers do not consider the impact of localization fluctuations, resulting in violent jittering or poor trajectory tracking of the mobile robot. For this reason, this paper proposes an adaptive model predictive control (MPC) with an accurate localization fluctuation assessment for mobile robots, which balances the contradiction between precision and calculation efficiency of mobile robot control. The distinctive features of the proposed MPC are three-fold: (1) Integrating variance and entropy—a localization fluctuation estimation relying on fuzzy logic rules is proposed to enhance the accuracy of the fluctuation assessment. (2) By using the Taylor expansion-based linearization method—a modified kinematics model that considers that the external disturbance of localization fluctuation is established to satisfy the iterative solution of the MPC method and reduce the computational burden. (3) An improved MPC with an adaptive adjustment of predictive step size according to localization fluctuation is proposed, which alleviates the disadvantage of a large amount of the MPC calculation and improves the stability of the control system in dynamic scenes. Finally, verification experiments of the real-life mobile robot are offered to verify the effectiveness of the presented MPC method. Additionally, compared with PID, the tracking distance and angle error of the proposed method decrease by 74.3% and 95.3%, respectively.

Keywords: model predictive control; mobile robots; localization fluctuations; fuzzy estimation

Citation: Meng, J.; Xiao, H.; Jiang, L.; Hu, Z.; Jiang, L.; Jiang, N. Adaptive Model Predictive Control for Mobile Robots with Localization Fluctuation Estimation. *Sensors* **2023**, *23*, 2501. https://doi.org/10.3390/s23052501

Academic Editor: Carlos Silvestre

Received: 31 December 2022
Revised: 12 February 2023
Accepted: 16 February 2023
Published: 23 February 2023

1. Introduction

Mobile robots are being progressively used in numerous scenarios such as unmanned factories, logistics centers, and exhibition halls, thanks to their superior flexibility and maneuverability [1–3]. For unmanned operations, autonomous navigation technology is intuitively important for robots [4]. To follow a given trajectory, mobile robots have to be able to control their pose precisely and robustly based on the localization results [5]. However, localization and control issues are often studied independently, leaving robot control performers to be improved.

The control system needs accurate localization results as a reference to maintain good trajectory-tracking accuracy [6]. In traditional control methods, kinematic or dynamic modelling or the control theory have been given more attention, and localization results are always seen as an absolute truth value [7,8]. However, in practice, mobile robots, whether using vision-based or LiDAR-based localization solutions, are subject to noise interference

from their external sensors, which may lead to fluctuations in localization [9,10]. If the controller still treats the localization result as absolute truth, this can lead to severe jittering or large tracking deviations. *To this end, this paper solves the robust control problem of mobile robots under the localization fluctuations, including the accurate localization fluctuation estimation and the robust controller design*.

Through the above analysis, to improve the operation accuracy of the four-wheel differential mobile robot, this paper analyses the localization fluctuation state in the dynamic scene and then realizes the adaptive adjustment of the predictive step size of the model predictive control. Therefore, the operation adaptability of the four-wheel differential mobile robot is improved.

(1) Integrating variance and information entropy—an enhanced localization fluctuation estimation method based on fuzzy logic rules is proposed to improve the accuracy of the fluctuation assessment.
(2) A modified kinematics model with external disturbance using the Taylor expansion-based linearization is established, which is convenient for controller design under localization fluctuations.
(3) An improved MPC with an adaptive adjustment of predictive step size related to localization fluctuation is proposed, which ensures the stability of the control system in dynamic scenes.
(4) The proposed method has been tested in real dynamic scenarios and compared with mainstream methods, and its effectiveness has been demonstrated.

2. Related Works

2.1. Localization Fluctuation Estimation

The problem of localization can be divided into simultaneous localization and map building (SLAM) and localization based on an a priori map, depending on the presence or absence of an a priori map [11]. Mainstream SLAM methods such as ORB-SLAM [12] and LEGO-LOAM [13] can generate a map of the environment while obtaining localization results. However, this method is subject to cumulative errors and has poor real-time performance. In contrast, a priori map-based localization provides accurate and efficient positional information and is often used as a reference for control [11]. Bayesian filter-based localization frameworks are currently the dominant approach using a priori map, such as Kalman filtering or Monte Carlo localization (MCL) [14–16]. In particular, MCL is widely used due to its ability to adapt to non-Gaussian non-linear scenarios [14,17]. Although there is much research around the robustness of MCL, existing algorithms are not immune to localization fluctuations in highly dynamic scenarios. The accurate description of the localization fluctuations is of great significance for the design of subsequent navigation systems. For this reason, Zapata et al. propose to use the maximum particle weight in the MCL as a benchmark to determine the current localization reliability [18]. When the maximum particle weight is less than the weight threshold, this is an indication that the currently estimated pose is not reliable. Nevertheless, it is not robust to use the weight of only one particle to measure localization reliability. In turn, variance and entropy values are common in addition to valid metrics for estimating localization fluctuations, which take into account the set of particles with weights in an integrated manner [19–21]. The variance is well understood mathematically and physically—the larger the variance, the greater the localization fluctuation—but is poorly described for localization data with multi-peaked distributions. Higher information entropy indicates smaller differences in particle weights, demonstrating greater uncertainty in localization. In particular, entropy is more accurate for non-convex data evaluations [22,23]. However, relying solely on a numerical metric is prone to misclassification. *In this regard, the main objective of this paper is to design a robust assessment method that integrates more localization fluctuation metrics into consideration*.

2.2. Mobile Robot Control

As one of the key modules, control technology has a great impact on the stability and accuracy of mobile robots [24]. In the process of motion, it is usually subject to external disturbances such as model uncertainty and parameter perturbation, resulting in motion oscillation and deviation, and even skidding and rollover [25]. To suppress disturbances, many scholars have made efforts to improve the performance of mobile bots. For example, the literature [26] designs a sliding mode control (SMC) scheme based on a reduced-order-extended-state observer, which realizes the active compensation of friction under the condition of uncertain parameters and ensures the stability of an operation. Through the smooth fitting of the path and design of MPC, the literature [27] realizes the high-speed movement of the four-wheeled independently steering robot with action delay. For the tracking problem of wheeled mobile robots with bounded disturbances and various practical constraints, a robust MPC method is proposed to ensure the safety and comfort during an operation [28]. At present, SMC and MPC have been the focus of research because of their excellent characteristics in high-speed and high-precision motion control methods [6,29,30]. Although SMC has the advantage of being insensitive to disturbances, the oscillation produced by itself is difficult to be eliminated, which makes it difficult to be widely used [31]. The gradual iterative optimization is brought by MPC, which can well handle the model constraints caused by structure, dynamic system, etc., and is conducive to achieving smooth motion [32,33]. This advantage improves the robustness of the control. In the design process of existing MPC methods, the observed localization data are usually treated as accurate values, which can ensure the stability of motion in static- or high-localization accuracy scenes [34,35]. Therefore, in the dynamic scene, the design of the controller needs to take into account the localization fluctuations, so as to avoid causing motion oscillation. *For this purpose, how to improve the robustness and accuracy of the controller in the localization fluctuation scenario has become a key issue to be studied in our work*.

3. System Modelling and Problem Formulation

3.1. System Modelling

Figure 1 shows the four-wheel differential platform model. The four-wheel differential platform has good motion performance and can achieve zero radius turning by adjusting the speed of the left and right wheels, which improves its adaptability to complex scenes. As the special case of mobile robots, the general modelling method can improve the general adaptability of the model [36,37]. Therefore, to further analyze the four-wheel differential platform and improve its motion controllability, the following general kinematics model of a mobile robot is given:

$$\chi = f(\chi, u) \tag{1}$$

where $\chi = [x, y, \theta]^T$ is the state variable; $u = [v, \omega]^T$ represents the control variable, x and y are the position of the mobile robot center point in the global fixed coordinate system, and θ denotes the robot heading angle; v and ω are the linear velocity and angular velocity of the mobile robot, respectively. The Taylor formula is used to expand the nonlinear mobile robot model at the reference point (χ_r, u_r) to obtain the linear model of the mobile robot, so that the modelling accuracy can be guaranteed.

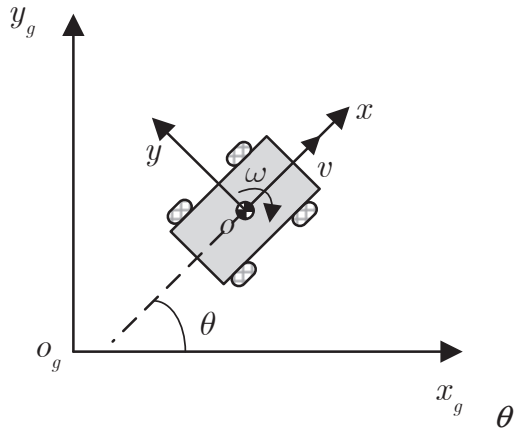


Figure 1. The four-wheel model and single-track model of the considered IWMD-MR.

At the same time, the linear model reduces the amount of computation, convenient for the design of the controller and is conducive to the actual implementation. Furthermore, at the Taylor expansion for Equation (1) at the point (χ_r, u_r), we have:

$$\dot{\chi} = f(\chi_r, u_r) + f_{\chi_r}(\chi - \chi_r) + f_{u_r}(u - u_r) + O_r \tag{2}$$

where χ_r is the reference control input; u_r denotes the calculated reference input, $f_{\chi_r}(\cdot)$ and are coefficient matrices of function $f(\cdot)$ expanded at (χ_r, u_r); $O(r)$ is the higher order remainder of the Taylor expansion. By defining $e = \chi - \chi_r$, we have

$$\dot{e} = f_{\chi_r} e + f_{u_r} \widetilde{u} + O_r \tag{3}$$

where e represents the following error of the mobile robot; $\widetilde{u} = u - u_r$ is the change of input control law. In the actual motion process, it is difficult to realize the continuous control of the mobile robot. Therefore, obtaining a discrete-time model of the four-wheel mobile robot is necessary. Setting the sampling period as T, where

$$e(k+1) = e(k) + T\dot{e}(k) \tag{4}$$

where k is the sampling time. The above model is rewritten as

$$e(k+1) = A(k)e(k) + B(k)\widetilde{u}(k) + O_r(k) \tag{5}$$

where,

$$A(k) = \begin{bmatrix} 1 & 0 & -T \cdot v_r \sin\theta_r(k) \\ 0 & 1 & T \cdot v_r \cos\theta_r(k) \\ 0 & 0 & 1 \end{bmatrix}, B(k) = \begin{bmatrix} T\cos\theta_r(k) & 0 \\ T\sin\theta_r(k) & 0 \\ 0 & T \end{bmatrix} \tag{6}$$

Therefore, in order to improve the controllability of the four-wheel mobile robot, we realized the linear modelling of the mobile robot kinematics model. This facilitates the design of the controller and makes the control of the mobile robot simpler.

3.2. Localization Problem Formulation

The mobile robot localization problem is often regarded as a typical Bayesian estimation problem. Bayesian filter-based localization algorithms solve robot localization problems by estimating the probability distribution of robot poses in the pose space and

assigning a probability to each possible hypothetical pose using a confidence level (*Belief*), which is expressed as

$$\mathcal{B}(\mathrm{s}_k) = p(\mathrm{s}_k|\boldsymbol{o}_{1:k}, \boldsymbol{u}_{1:k}, \boldsymbol{M}) \tag{7}$$

where s_k is the robot's pose in the two-dimensional plane at time k, which can be expressed as (x_k, y_k, θ_k); x_k and y_k are the robot's position, and θ_k is the robot's heading; $\boldsymbol{o}_{1:k}$ and $\boldsymbol{u}_{1:k}$ represent the sensor observations and motion control from the initial time to time k, respectively; $\boldsymbol{M}$ is the a priori map.

According to the Markov hypothesis and Bayes rule, the Bayesian filter-based localization method can recursively estimate the robot's poses, but it involves a large number of integral operations and nonlinear non-Gaussian features of the observation and motion models, which make the efficient and accurate solution a concern. To this end, this paper uses the MCL approach to compute Equation (7), which uses a particle set with weights to represent $\mathcal{B}(s_k)$, i.e.,

$$\mathcal{B}(s_k) \approx \sum_{n_p=1}^{N_p} \omega_k^{[n_p]} \delta(s_k - s_k^{[n_p]}) \tag{8}$$

where the set of particles with weights is denoted as $\left\{ \left\langle s_k^{[n_p]}, \omega_k^{[n_p]} \right\rangle \right\}_{n_p=1}^{N_p}$; $\omega_k^{[n_p]}$ is the weight of the n_p-*th* particle $s_k^{[n_p]}$ and N_p means the total number of particles; $s_k^{[n_p]}$ can be written as $\left(x_k^{[n_p]}, y_k^{[n_p]}, \theta_k^{[n_p]}\right)$; $x_k^{[n_p]}$, $y_k^{[n_p]}$ and $\theta_k^{[n_p]}$ are the position and orientation of the particle; $\delta(\cdot)$ is the Dirichlet function. Usually, the particle with the highest weight is selected as the current localization result.

4. Localization Fluctuations Estimation

Both variance and information entropy have their own unique advantages in expressing localization fluctuations and can reflect the characteristics of the particle set. However, a single performance metric still has a large randomness that affects the accuracy of the fluctuation assessment. To solve the above problems, a fuzzy logic rule incorporating variance and entropy is proposed to evaluate the localization fluctuations. Localization fluctuations are represented as follows:

$$L_f = [L_{fx}, L_{fy}, L_{f\theta}]^{\mathrm{T}} = [f_{Vx}(V_x) + f_E(E), f_{Vy}(V_y) + f_E(E), f_{V\theta}(V_\theta) + f_E(E)]^{\mathrm{T}} \tag{9}$$

$$f_{Vi}(V_i) = \begin{cases} \alpha_V & 0 \le V_i < \eta_1 V_{Ti} \\ \beta_V & \eta_1 V_{Ti} \le V_i < \eta_2 V_{Ti} \\ \lambda_V & \eta_2 V_{Ti} \le V_i \end{cases}, \; f_E(E) = \begin{cases} \alpha_E & 0 \le E < \eta_3 E_T \\ \beta_E & \eta_3 E_T \le E < \eta_4 E_T \\ \lambda_E & \eta_4 E_T \le E \end{cases} \tag{10}$$

where V_i and E are the variance and entropy, respectively; $i = x, y, \theta$; $f_{Vi}(V_i)$ and $f_{Ei}(E_i)$ are the localization fluctuation factors based on V_i and E; α_V, β_V, λ_V, α_E, β_E and λ_E are the fluctuation parameters; $0 < \alpha_V < \beta_V < \lambda_V$ and $0 < \alpha_E < \beta_E < \lambda_E$; η_1, η_2, η_3 and η_4 are the weight coefficient; $0 < \eta_1 < \eta_2$ and $0 < \eta_3 < \eta_4$; V_{Ti} and E_T are the variance and entropy threshold value, severally. L_f is a three-dimensional vector $[L_{fx}, L_{fy}, L_{f\theta}]^{\mathrm{T}}$ that represents the fluctuations of x, y, and θ. Meanwhile, L_f is determined by the variance and entropy values, where the entropy values are only related to the particle weights, so all three dimensions are set uniformly.

It is worth stating that the calculation of V_i and E usually requires a series of localization results over a period of time. However, it is not practical to perform a large number of localization experiments in situ to determine the current localization fluctuation state, and we would prefer to conduct the evaluation depending on the localization data at a certain

time. Fortunately, the MCL result is expressed as a particle set $\left\{\left\langle s_k^{[n_p]}, \omega_k^{[n_p]}\right\rangle\right\}_{n_p=1}^{N_p}$, so that V_i and E can be represented as

$$V_{px} = \sum_{i=1}^{N_p} \omega_k^{[n_p]} \left(x_k^{n_p} - \overline{x}_k\right)^2,\ V_{py} = \sum_{i=1}^{N_p} \omega_k^{[n_p]} \left(y_k^{n_p} - \overline{y}_k\right)^2,\ V_{p\theta} = \sum_{i=1}^{n} \omega_t^i \left(\theta_t^i - \overline{\theta}_t\right)^2 \tag{11}$$

$$E_p = -\sum_{n_p=1}^{N_p} \omega_k^{[n_p]} \log \omega_k^{[n_p]} \tag{12}$$

where V_{pi} and E_p are the variance and entropy from $\left\{\left\langle s_k^{[n_p]}, \omega_k^{[n_p]}\right\rangle\right\}_{n_p=1}^{N_p}$; $\overline{*}$ denotes the mean value of the related particle's pose.

Although V_i and E can be reflected by the localization results at a certain time based on $\left\{\left\langle s_k^{[n_p]}, \omega_k^{[n_p]}\right\rangle\right\}_{n_p=1}^{N_p}$, the exact relationship between V_{pi}, V_i, E_p and E are still difficult to be indicated analytically. Fuzzy logic rules are an effective means to infer an output based on input variables. Hence, we integrate V_{pi} and V_i into a fuzzy formula using the following fuzzy logic rules:

$$\begin{array}{llllll} \text{rule 1}: & if & 0 \le V_{pi} \le V_{pi1} & then\ f_{Vi1} = f_{Vi}(V_i) & s.t. & 0 \le V_i < \eta_1 V_{Ti} \\ \text{rule 2}: & if & V_{pi2} \le V_{pi} \le V_{pi3} & then\ f_{Vi2} = f_{Vi}(V_i) & s.t. & \eta_1 V_{Ti} \le V_i < \eta_2 V_{Ti} \\ \text{rule 3}: & if & V_{pi4} \le V_{pi} & then\ f_{Vi3} = f_{Vi}(V_i) & s.t. & \eta_2 V_{Ti} \le V_i \end{array} \tag{13}$$

where $V_{pi1}, V_{pi2}, V_{pi3}$ and V_{pi4} are fuzzy demarcation boundaries for V_{pi}; f_{Vi1}, f_{Vi3} and f_{Vi3} are the fluctuation values based on variance.

Similarly, the fuzzy formula for describing the mapping relation between E_p and E can be written as

$$\begin{array}{llllll} \text{rule 1}: & if & 0 \le E_p \le E_{p1} & then\ f_{E1} = f_E(E) & s.t. & 0 \le E < \eta_3 E_T \\ \text{rule 2}: & if & E_{p2} \le E_p \le E_{p3} & then\ f_{E2} = f_E(E) & s.t. & \eta_3 E_T \le E < \eta_4 E_T \\ \text{rule 3}: & if & E_{p4} \le E_p & then\ f_{E3} = f_E(E) & s.t. & \eta_4 E_T \le E \end{array} \tag{14}$$

where $E_{p1},\ E_{p2},\ E_{p3}$ and E_{p4} are fuzzy demarcation boundaries for E_p; $f_{E1},\ f_{E3}$ and f_{E3} are the fluctuation values based on entropy.

For implementation, the boundaries V_{pi1} to V_{pi4} and E_{p1} to E_{p4} can be learned from a mass of variance and entropy of localization results in different dynamic environments. As the next step of a standard procedure of constructing a fuzzy logic system, defuzzification is achieved by the weighted average method. Additionally, the fluctuation factor L_f is obtained as follows:

$$L_f = [L_{fx}, L_{fy}, L_{f\theta}]^{\mathrm{T}} = \left[\frac{\sum_{m=1}^{r} \chi_m f_{Vxm}}{\sum_{m=1}^{r} \chi_m} + \frac{\sum_{m=1}^{r} \chi_m f_{Em}}{\sum_{m=1}^{r} \chi_m}, \frac{\sum_{m=1}^{r} \chi_m f_{Vym}}{\sum_{m=1}^{r} \chi_m} + \frac{\sum_{m=1}^{r} \chi_m f_{Em}}{\sum_{m=1}^{r} \chi_m}, \frac{\sum_{m=1}^{r} \chi_m f_{V\theta m}}{\sum_{m=1}^{r} \chi_m} + \frac{\sum_{m=1}^{r} \chi_m f_{Em}}{\sum_{m=1}^{r} \chi_m}\right]^{\mathrm{T}} \tag{15}$$

where r is the number of fuzzy rule bases; χ_m denotes the trigger strength of m-th rule. If the V_{pi} or E_p satisfies the fuzzy rule, then χ_m is 1. Otherwise, χ_m is set as 0.

5. Adaptive MPC Considering Localization Fluctuation

In order to realize the robust control of the four-wheel mobile robot, based on the state space discrete model of Equation (5), we construct the cost function required in the optimal control process

$$J(e(k),\widetilde{u}(k)) = \sum_{i=1}^{N_p}[e(k+i|k)]^T P[e(k+i|k)] + \sum_{i=0}^{N_c-1}[\widetilde{u}(k+i|k)]^T R[\widetilde{u}(k+i|k)] \tag{16}$$

where $J(\cdot)$ is the cost function; $e(k+i|k)$ and $\widetilde{u}(k+i|k)$ represent the error value and control fluctuation value of time $k+1$ predicted at time k, respectively; N_p and N_c are prediction and control horizon, respectively, and the value of the control horizon is not greater than the prediction horizon; P and R are the weight matrix.

To ensure the feasibility of optimal prediction, the control variables of the prediction process are as follows:

$$\begin{gathered} e_{\min} \le e(k) \le e_{\max} \\ u_{\min} \le u(k) \le u_{\max} \\ \widetilde{u}_{\min} \le \widetilde{u}(k) \le \widetilde{u}_{\max} \\ u_{\min} \le u(k) + T\widetilde{u}(k) \le u_{\max} \\ O_{r\min} \le O_r(k) \le O_{r\max} \end{gathered} \tag{17}$$

where $e_{\min}$ and $e_{\max}$ are the minimum and maximum errors, respectively. $u_{\min}$ and $u_{\max}$ are the minimum and maximum values of the control law increment, respectively. $O_{r\min}$ and $O_{r\max}$ are the minimum and maximum perturbations, respectively.

Considering the existence of localization fluctuation, to ensure the stability of operation, N_p and N_c need to be further adjusted as follows:

$$\begin{gathered} N_p = [k_1 \max(L_f)] + k_p \\ N_c = [k_2 \max(L_f)] + k_c \end{gathered} \tag{18}$$

where L_f is the estimated localization fluctuation value obtained from Equation (9), $\max()$ is the maximum value function of a vector, $k_{1,2}$ is the adjustment coefficient. $k_c, k_p \in N_+$ is the minimum adjustment coefficient. $[X]$ is the maximum integer value not greater than X. Considering the constraint state of the predictive control method, there are $N_c \le N_p$.

Furthermore, using the cost function of Equation (16), the following state equation of the prediction horizon is obtained:

$$\overline{e}(k) = F(k)\overline{\widetilde{u}}(k) + L(k)e(k) + G(k)\widetilde{u}(k-1) + \overline{O}_r(k) \tag{19}$$

with

$$\begin{gathered} \overline{e}(k) = [e(k+1|k), \ldots, e(k+N_p|k)]^T \\ \overline{\widetilde{u}}(k) = [\widetilde{u}(k+1|k), \ldots, \widetilde{u}(k+N_c-1|k)]^T \\ \overline{O}_r(k) = [O_r(k+1|k), \ldots, O_r(k+N_p-1|k)]^T \end{gathered} \tag{20}$$

The coefficient matrix is expressed as:

$$F(k) = \begin{bmatrix} B(k) & \cdots & 0 \\ A(k+1)B(k) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ B(k+N_p-1) + \prod_{i=k+1}^{i=k+N_p-1} A(i)B(k) & \cdots & B(k+N_p-1) + \prod_{i=k+1}^{i=k+N_p-1} A(i)B(k+N_c) \end{bmatrix} \tag{21}$$

$$L(k) = \left[A(k), A(k+1)A(k), \ldots, \prod_{i=k}^{i=k+N_p-1} A(i) \right]^T \tag{22}$$

$$G(k) = \begin{bmatrix} B(k) \\ B(k+1) + A(k+1)B(k) \\ \vdots \\ B(k+N_p-1) + \ldots + \prod_{i=k+1}^{i=k+N_p-1} A(i)B(k) \end{bmatrix} \tag{23}$$

Therefore, in order to ensure the optimal operation process, the following optimization objectives are obtained by combining Formulas (16), (17) and (19):

$$\min \bar{e}^T(k) P \bar{e}(k) + \bar{\widetilde{u}}^T(k) R \bar{\widetilde{u}}(k) \tag{24}$$

Considering modelling error and disturbance, Equation (19) is introduced into the optimization objective (24), thus:

$$\min \overline{O}_r{}^T(k) P \overline{O}_r(k) + \bar{\widetilde{u}}^T(k) R \bar{\widetilde{u}}(k) \tag{25}$$

subject to

$$\begin{gathered} \min(e_{\max} - M(k), \overline{O}_{r\max}) \le \overline{O}_r(k) \le \max(e_{\min} - M(k), \overline{O}_{r\min}) \\ u_{\min} \le u(k) \le u_{\max} \\ \widetilde{u}_{\min} \le \widetilde{u}(k) \le \widetilde{u}_{\max} \\ u_{\min} \le u(k) + T\widetilde{u}(k) \le u_{\max} \end{gathered} \tag{26}$$

where $M(k) = F(k)\bar{\widetilde{u}}(k)L(k)e(k) - G(k)\widetilde{u}(k-1)$ is the intermediate variable.

By adjusting the above equation, the control law optimization under the scenario of disturbance fluctuation is realized.

Furthermore, the stability proof of the designed controller is given as follows:

Theorem 1. *When the optimal control strategy of (19)–(25) is adopted and the following conditions are satisfied:*

$$e(k+N_p|k) = 0 \tag{27}$$

$$\widetilde{u}(k+N_c) = 0 \tag{28}$$

Then, the system will approach stability.

Proof. The optimization function of the system can be rewritten as:

$$\begin{aligned} J(e(k+1), \widetilde{u}(k+1)) &= \sum_{i=1}^{N_p} [e(k+i+1|k+1)]^T P [e(k+i+1|k+1)] \\ &+ \sum_{i=0}^{N_c-1} [\widetilde{u}(k+i+1|k+1)]^T R [\widetilde{u}(k+i+1|k+1)] \\ = \sum_{i=1}^{N_p} [e*(k+i+1|k)]^T P [e*(k+i+1|k)] &+ \sum_{i=0}^{N_c-1} [\widetilde{u}*(k+i+1|k)]^T R [\widetilde{u}(k+i+1|k)] \end{aligned} \tag{29}$$

Combined with Formulas (27) and (28), there is

$$\begin{aligned} J(e(k+1),\widetilde{u}(k+1)) &= \sum_{i=1}^{N_p}[e*(k+i|k)]^T P[e*(k+i|k)] + \sum_{i=0}^{N_c-1}[\widetilde{u}*(k+i|k)]^T R[\widetilde{u}*(k+i|k)] \\ &\quad -[e*(k+1|k)]^T P[e*(k+1|k)] - [\widetilde{u}*(k|k)]^T R[\widetilde{u}*(k|k)] \\ &= J*(e(k),\widetilde{u}(k)) - [e*(k+1|k)]^T P[e*(k+1|k)] - [\widetilde{u}*(k|k)]^T R[\widetilde{u}*(k|k)] \end{aligned} \tag{30}$$

with $J^*(e(k+1),\widetilde{u}(k+1))$, which is the optimal control value at time k. Then, $e^*(k+i|k)$ and $\widetilde{u}^*(k+i|k)$ are the predicted values obtained by solving the optimization function at time k to $k+i$. Hence, combining Equations (27) and (28), we have:

$$e^*(k+1) = e^*(k+1|k),\ \widetilde{u}^*(k) = \widetilde{u}^*(k|k) \tag{31}$$

Then,

$$\begin{aligned} J^*(e(k+1),\widetilde{u}(k+1)) &\leq J(e(k+1),\widetilde{u}(k+1)) \\ &= J*(e(k),\widetilde{u}(k)) - [e*(k+1)]^T P[e*(k+1)] - [\widetilde{u}(k)]^T R[\widetilde{u}^*(k)] \\ &\leq J^*(e(k),\widetilde{u}(k)) \end{aligned} \tag{32}$$

Therefore, the optimal value of the cost function will gradually decrease in the process of iterative updating. It shows that the optimization method can guarantee the asymptotic convergence of the system. □

6. Experimental Validations

6.1. Experimental Implementation

The experimental scenes and four-wheel differentially driven mobile robot are shown in Figure 2. The mobile robot basically consists of an industrial computer (Intel(R) Core (TM) i7-6500U CPU @2.50 GHz, 8 GB of RAM, 64-bit operating system), two LiDARs, four motor encoders and some related sensors, such as an ultrasonic transducer and anti-collision strip. More specifically, two UTM-30LX 2-D LiDARs with a range of 30 m and scanning rate of 40 Hz are used, which guarantee that the robot has enough field of view to ensure safety and real-time pose tracking. To avoid the blind spot of LiDAR, we installed four MaxBotix MB7360 ultrasonic sensors around the robot with a high resolution of 1 mm and a measuring distance of 5 m, which is sufficient to detect possible dynamic obstacles during the robot's movement. As demonstrated in Figure 2b, the experimental environment is a dynamic and wide square surrounded by overgrown grass and a large number of pedestrians. Three sites are selected for the localization experiments to determine the fuzzy rule parameters. To comprehensively demonstrate the performance of the proposed control system, we first conduct localization experiments in different dynamic environments. Then, through real control experiments, the effectiveness of the proposed MPC method is verified.

(**a**) (**b**) (**c**)

Figure 2. Experimental scene and platform. (**a**) Platform prototype. (**b**) Dynamic scene. (**c**) Grid map of the scene.

6.2. Experimental Results and Discussions

6.2.1. Experimental Results of Localization Fluctuation Estimation

To construct fuzzy logic rules for localization fluctuation estimation, 100 frames of LiDAR measurement are collected at each of the three sites in different dynamic scenarios using a mobile robot. In total, 900 frames of LiDAR data were provided for the fuzzy estimation. The LiDAR data from low to high dynamic scenes at Site 2 in the square are shown in Figure 3a. The LiDAR data from low to high dynamic scenes at Site 2 in the square are shown in Figure 1. Specifically, as shown in Figure 1, the low dynamic scenes have less than 10 percent of the observed noise, which is ideal for localization. The medium dynamic scenario is depicted in Figure 3b, where the observed noise accounts for 20 to 30 percent of the overall observed data, a common and realistic scenario. Extremely dynamic scenarios, such as the one shown in Figure 3c, will have noise levels close to 50 percent, which can easily lead to localization fluctuations or even localization failure.

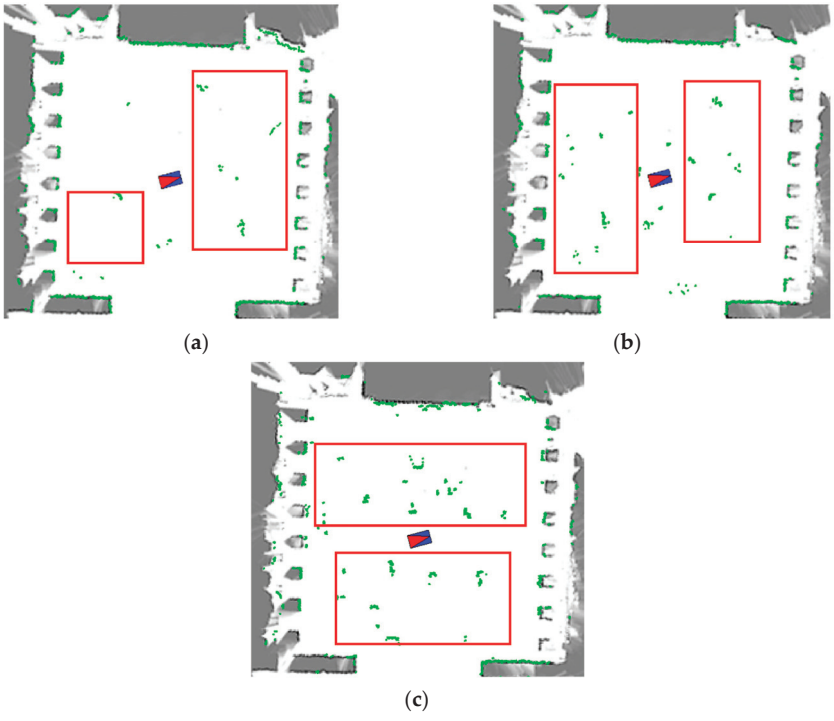


Figure 3. LiDAR measurement in different dynamic scenes at site 2. (**a**) Low dynamic environment. (**b**) Medium dynamic environment. (**c**) High dynamic environment.

Next, we determine the relationship between V_p and E, and the mapping between E_p and E. We put a different number of obstacles around the robot and record 100 localization results. V_i and E correspond to each localization result at three sites. In this way, as indicated in Table 1, we get the V_i and E of 100 localization results at three sites in different dynamic environments. And, as the dynamic level of the scene increases, V_i and E become correspondingly larger.

Table 1. The results of the V_i and E of multiple localization results in different dynamic environments.

	Low Dynamic Scene				Medium Dynamic Scene				High Dynamic Scene			
	$V_x/(m^2)$	$V_y/(m^2)$	$V_\theta/(rad^2)$	E	$V_x/(m^2)$	$V_y(m^2)$	$V_\theta/(rad^2)$	E	$V_x/(m^2)$	$V_y(m^2)$	$V_\theta/(rad^2)$	E
Site 1	1.24×10^{-4}	3.23×10^{-4}	1.53×10^{-6}	6.33	2.19×10^{-4}	8.55×10^{-4}	1.93×10^{-6}	6.41	4.67×10^{-4}	6.28×10^{-3}	7.48×10^{-5}	6.56
Site 2	1.62×10^{-4}	5.71×10^{-4}	5.57×10^{-7}	6.32	2.09×10^{-4}	7.45×10^{-4}	1.04×10^{-6}	6.43	4.33×10^{-4}	1.26×10^{-3}	2.70×10^{-6}	6.51
Site 3	1.80×10^{-4}	2.63×10^{-4}	1.48×10^{-6}	6.37	2.66×10^{-4}	9.09×10^{-4}	4.76×10^{-6}	6.39	3.45×10^{-4}	1.83×10^{-3}	8.97×10^{-6}	6.54

According to the V_i and E in different dynamic scenes, we can know the status of the localization fluctuations to which V_p and E belong. Meanwhile, for example, when $i = x$, Equation (10) can be set to

$$f_{Vx}(V_x) = \begin{cases} 0.00 & 0.00 \le V_x < 1.93 \times 10^{-4} \\ 0.25 & 1.93 \times 10^{-4} \le V_x < 3.23 \times 10^{-4} \\ 0.5 & 3.23 \times 10^{-4} \le V_x \end{cases}, \; f_E(E) = \begin{cases} 0 & 0.00 \le E < 6.37 \\ 0.25 & 6.37 \le E < 6.47 \\ 0.5 & 6.47 \le E \end{cases} \tag{33}$$

Specifically, the relationship between V_{pi} and V_i, and the mapping between E_p and E, are shown in Table 2. We have grouped V_{pi} and E_p, which fit into different fluctuation ranges, into one category. The 300 V_{pi} and V_i ranges are linked. Similarly, the mapping of the ranges of E and E_p is found.

Table 2. The results of V_{pi} and E_p in different ranges.

Range	Range1: $0.00 \le V_x < 1.93 \times 10^{-4}$			Range2: $1.93 \times 10^{-4} \le V_x < 3.23 \times 10^{-4}$			Range3: $3.23 \times 10^{-4} \le V_x$		
$V_{px}/(m^2)$	7.03×10^{-4}	3.06×10^{-4}	5.05×10^{-4}	1.17×10^{-2}	3.54×10^{-4}	3.92×10^{-4}	1.18×10^{-1}	9.10×10^{-4}	5.84×10^{-4}
	98 lines of V_{px}								
	4.76×10^{-4}	3.37×10^{-4}	4.57×10^{-4}	3.96×10^{-4}	2.88×10^{-4}	1.49×10^{-4}	1.39×10^{-3}	7.85×10^{-4}	8.83×10^{-4}
Range	range1: $0.00 \le V_y < 6.11 \times 10^{-4}$			range2: $6.11 \times 10^{-4} \le V_y < 1.98 \times 10^{-3}$			range3: $1.98 \times 10^{-3} \le V_y$		
$V_{py}/(m^2)$	8.01×10^{-4}	4.08×10^{-4}	1.45×10^{-4}	2.61×10^{-2}	3.24×10^{-4}	9.01×10^{-4}	2.82×10^{-1}	1.20×10^{-2}	1.37×10^{-3}
	98 lines of V_{px}								
	2.75×10^{-4}	1.86×10^{-4}	2.69×10^{-4}	2.18×10^{-4}	3.21×10^{-4}	1.40×10^{-4}	1.12×10^{-3}	8.20×10^{-4}	1.14×10^{-3}
Range	range1: $0.00 \le V_\theta < 1.88 \times 10^{-6}$			range2: $1.88 \times 10^{-6} \le V_\theta < 1.57 \times 10^{-5}$			range3: $1.57 \times 10^{-5} \le V_\theta$		
$V_{p\theta}/(rad^2)$	4.59×10^{-6}	1.03×10^{-6}	3.50×10^{-7}	3.14×10^{-4}	8.30×10^{-7}	9.34×10^{-7}	5.13×10^{-3}	1.20×10^{-4}	1.99×10^{-5}
	98 lines of $V_{p\theta}$								
	7.66×10^{-6}	2.45×10^{-6}	3.01×10^{-6}	9.88×10^{-7}	2.12×10^{-6}	1.69×10^{-6}	1.09×10^{-4}	2.11×10^{-5}	5.95×10^{-5}
Range	range1: $0.00 \le E$ <6.37			range2: $6.37 \le E$ <6.47			range3: $6.47 \le E$		
E_p	6.22	2.51	5.18	5.05	6.30	5.00	1.11	8.85	7.79
	98 lines of E_p								
	4.88	5.03	4.07	5.73	5.39	5.19	7.29	7.98	7.94

The distribution of V_{pi} and E_p in the different ranges of V_i and E are shown in Figure 4. We have used box plots to illustrate the distribution characteristics. It is easy to see that there are overlapping parts of V_{pi} and E_p in different ranges, which provide for the construction of fuzzy rules. According to Figure 4, the boxes in different ranges have a distinct overlap. Hence, we can easily choose the fuzzy partition boundary $V_{pi1} \square V_{pi4}$ and $E_{p1} \square E_{p4}$ for V_{pi} and E_p, respectively. Finally, the "premise" of the fuzzy rule base defines V_{pi} and E_p, and the fluctuation values $f_{Vi1} \square f_{Vi3}$ and $f_{E1} \square f_{E3}$ are the "conclusion". Equations (13) and (14), when $i = x$, can be rewritten as

$$\begin{array}{lllll} \text{rule 1}: if & 0.00 \le V_{px} \le 5\times10^{-4} & then\ f_{Vx1} = f_{Vx}(V_x) & s.t. & 0.00 \le V_x < 1.93\times10^{-4} \\ \text{rule 2}: if & 1.62\times10^{-4} \le V_{px} \le 6.07\times10^{-4} & then\ f_{Vi2} = f_{Vx}(V_x) & s.t. & 1.93\times10^{-4} \le V_x < 3.23\times10^{-4} \\ \text{rule 3}: if & 4.94\times10^{-4} \le V_{px} & then\ f_{Vi3} = f_{Vi}(V_x) & s.t. & 3.23\times10^{-4} \le V_x \end{array} \tag{34}$$

$$\begin{array}{lllll} \text{rule 1}: if & 0.00 \le E_p \le 5.21 & then\ f_{E1} = f_E(E) & s.t. & 0.00 \le E < 6.37 \\ \text{rule 2}: if & 4.46 \le E_p \le 6.27 & then\ f_{E2} = f_E(E) & s.t. & 6.37 \le E < 6.47 \\ \text{rule 3}: if & 6.04 \le E_p & then\ f_{E3} = f_E(E) & s.t. & 6.47 \le E \end{array} \tag{35}$$

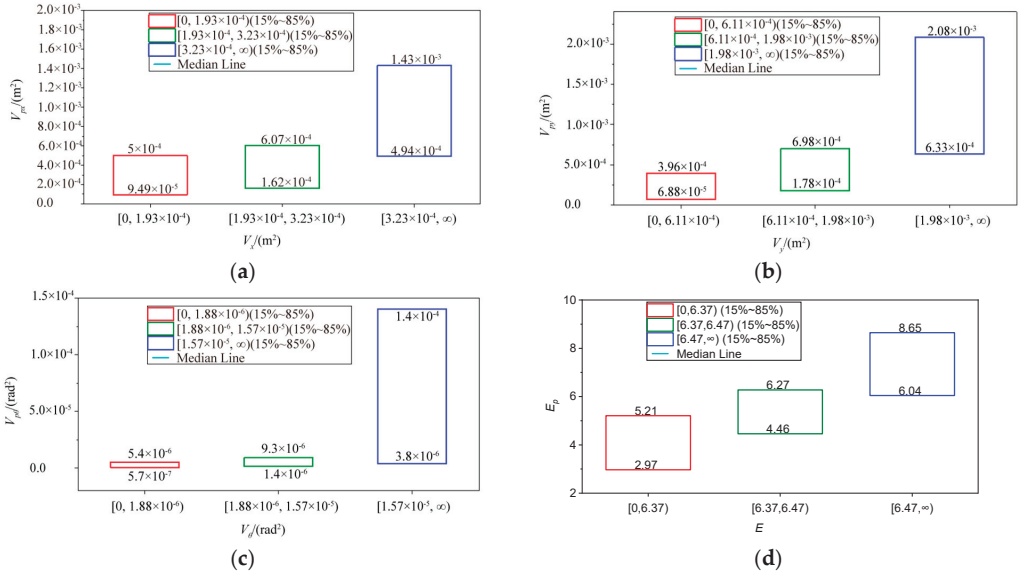


Figure 4. The distribution of V_p and E_p in the different ranges of V and E. (**a**) V_{px} in the different ranges of V_x. (**b**) V_{py} in the different ranges of V_y. (**c**) $V_{p\theta}$ in the different ranges of V_θ. (**d**) E_p in the different ranges of E.

6.2.2. Experimental Results of Adaptive MPC

Furthermore, to verify the superiority of the control method considering location uncertainty, we choose the following comparison methods: (1) The traditional PID control method with optimizing parameters, where $K_p = 0.5$, $K_i = 1.5$ and $K_d = 0.1$. (2) The proposed control method does not consider the localization fluctuation (NMPC) with $N_c = N_p = 5$. In the proposed control method, the control parameters are set as follows: $k_p = k_c = 5$, $k_1 = k_2 = 10$. The specific tracking process of the comparison method is described as follows:

The trajectory tracking of various comparison methods is shown in Figure 5. The initial point of the four-wheel differential mobile robot is (10, 0). It can be seen from Figure 5 that there is an error between the attitude angle of the initial point and the slope of the curve, resulting in a large tracking error. With time adjustment, all control methods can achieve fast convergence. It can be seen from the enlarged drawing of Figure 5 that compared with the proposed method and NMPC method, PID has a large oscillation. Further analysis shows that the proposed method has a better tracking effect.

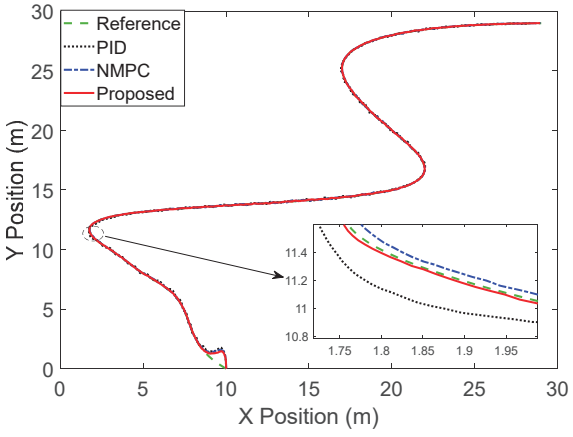


Figure 5. Tracking response of the comparison method.

Figures 6 and 7 are the distance error and angle error, respectively. Specifically, Figure 7 shows that the overshoots of the PID and NMPC are 0.5903 m and 0.5502 m, while the proposed method is 0.4603 m. Furthermore, by comparing the proposed method with NMPC, it is found that the tracking process is optimized through adaptive step adjustment. On closer inspection, the average errors in the stable stage (10 s~60 s) of PID, NMPC and proposed methods are 0.0374 m, 0.0177 m and 0.0096 m, which show that the errors are reduced by 74.3% and 45.8%, respectively. It is found in 0 that the angle error of the PID method fluctuates greatly, while the NMPC and the proposed method will be limited to a small range. The average angle errors of PID and NMPC are 0.1001 rad and 0.0062 rad during a 10~60 s stabilization period, while the proposed method with a small error is 0.0047 rad. The stable errors are reduced up to 95.3% and 31.9%, respectively. By comparison, the proposed method achieves better error suppression.

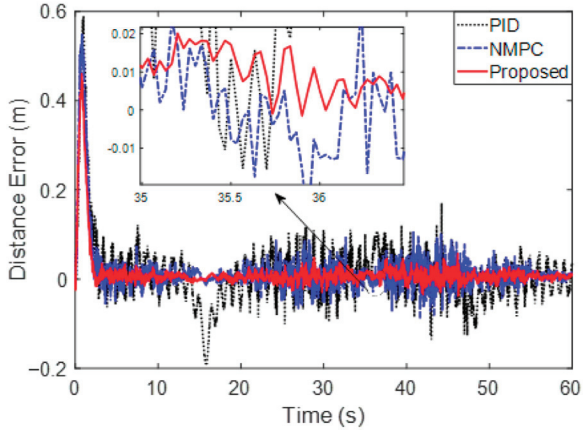


Figure 6. Distance error of the comparison method.

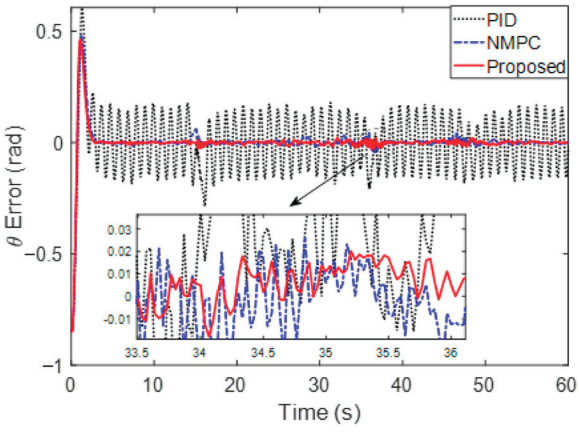


Figure 7. Orientation error of the comparison method.

From Figure 8, the adjustment of the prediction horizon is given through the localization fluctuation output by the localization module to ensure the control's stability and accuracy under the high localization fluctuation scenario. Therefore, compared with the traditional PID and NMPC methods, the proposed method can realize the dynamic adjustment of the step size, improve tracking accuracy and reduce motion fluctuation.

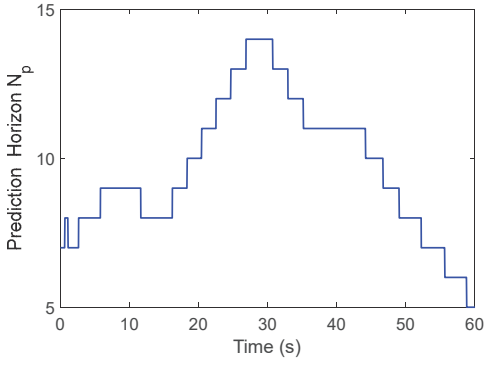


Figure 8. Adjustment of the prediction horizon.

7. Conclusions and Outlook

In this paper, an adaptive MPC for mobile robots with the fuzzy estimation of localization fluctuations was discussed, which accurately evaluates the localization state and effectively improves control accuracy and robustness. First of all, a localization fluctuation estimation based on fuzzy logic rules was proposed, which takes into account both the effects of variance and information entropy, thereby obtaining accurate estimates of localization fluctuations. Then, a modified kinematics model with external disturbance using the Taylor expansion-based linearization was constructed. In addition, according to localization fluctuation, an enhanced MPC with an adaptive adjustment of predictive step size was proposed to maintain the stability of the control system in dynamic scenes. The experimental results show that the step size of MPC can be effectively and adaptively adjusted with localization fluctuation in different dynamic scenes, avoiding the generation of oscillation. In the stable tracking phase, the proposed method reduced the tracking distance error by 74.3% and 45.8% compared with PID and NMPC. In addition, in comparison

to PID and NMPC, the tracking angle errors of the proposed method decreased by 95.3% and 31.9%, respectively. In future work, we will explore the potential of deep learning for applications in localization fluctuation assessments and improve the control framework to better integrate localization fluctuation information.

Author Contributions: Conceptualization, J.M. and N.J.; methodology, J.M., N.J. and L.J. (Liquan Jiang); software, H.X. and L.J. (Liyu Jiang); validation, L.J. (Liyu Jiang) and Z.H.; formal analysis, L.J. (Liyu Jiang); investigation, H.X.; resources, J.M. and N.J.; data curation, L.J. (Liquan Jiang); writing—original draft preparation, J.M. and N.J.; writing—review and editing, J.M.; visualization, L.J. (Liquan Jiang) and Z.H.; supervision, N.J.; project administration, N.J.; funding acquisition, N.J. and J.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by the Natural Science Foundation of Hubei Province, China under Grant no. 20221j0060, in part by the Fund of National Engineering Research Center for Water Transport Safety under Grant no. A202303, in part by the Natural Science Foundation of Chongqing, China under Grant no. CSTB2022NSCQ-MSX1566 and in part by Hubei Provincial Engineering Research Center for Intelligent Textile and Fashion (Wuhan Textile University) under Grant no. 2022HBITF01.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Notations and Abbreviations

The following key notations and abbreviations are used in this paper.

Notations/Abbreviations	Descriptions
SMC	sliding mode control
MPC	model predictive control
$J(\cdot)$	cost function
N_p, N_c	Prediction/control horizon
P, R	weight matrix
v	linear velocity
ω	angular velocity
$f_{\chi_r}(\cdot), f_{u_r}(\cdot)$	coefficient matrices
$O(r)$	higher order remainder of the Taylor expansion
P_l	estimated localization fluctuation value
$k_{1,2}$	adjustment coefficient
$k_c, k_p \in N_+$	minimum adjustment coefficient

References

1. Rubio, F.; Valero, F.; Llopis-Albert, C. A review of mobile robots: Concepts, methods, theoretical framework, and applications. *Int. J. Adv. Robot. Syst.* **2019**, *16*, 1729881419839596. [CrossRef]
2. Skoczeń, M.; Ochman, M.; Spyra, K.; Nikodem, M.; Krata, D.; Panek, M.; Pawłowski, A. Obstacle detection system for agricultural mobile robot application using RGB-D cameras. *Sensors* **2021**, *21*, 5292. [CrossRef] [PubMed]
3. Meng, J.; Wang, S.; Jiang, L.; Xie, Y.; Zheng, S.; Wu, H. Robust lateral stabilization control of in-wheel-motor-driven mobile robots via active disturbance suppression approach. *Sensors* **2020**, *20*, 5238. [CrossRef]
4. Higgins, J.; Bezzo, N. Negotiating visibility for safe autonomous navigation in occluding and uncertain environments. *IEEE Robot. Autom. Lett.* **2021**, *6*, 4409–4416. [CrossRef]
5. Zhang, X.; Xie, Y.; Jiang, L.; Li, G.; Meng, J.; Huang, Y. Fault-tolerant dynamic control of a four-wheel redundantly-actuated mobile robot. *IEEE Access* **2019**, *7*, 157909–157921. [CrossRef]

6. Jiang, L.; Wang, S.; Xie, Y.; Xie, S.; Zheng, S.; Meng, J.; Ding, H. Decoupled Fractional Supertwisting Stabilization of Interconnected Mobile Robot Under Harsh Terrain Conditions. *IEEE Trans. Ind. Electron.* **2021**, *69*, 8178–8189. [CrossRef]
7. Tzafestas, S.G. Mobile robot control and navigation: A global overview. *J. Intell. Robot. Syst.* **2018**, *91*, 35–58. [CrossRef]
8. Jiang, L.; Wang, S.; Xie, Y.; Xie, S.Q.; Zheng, S.; Meng, J. Fractional robust finite time control of four-wheel-steering mobile robots subject to serious time-varying perturbations. *Mech. Mach. Theory* **2022**, *169*, 104634. [CrossRef]
9. Meng, J.; Wan, L.; Wang, S.; Jiang, L.; Li, G.; Wu, L.; Xie, Y. Efficient and reliable LiDAR-based global localization of mobile robots using multiscale/resolution maps. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–15. [CrossRef]
10. Zhang, M.; Liu, X.; Xu, D.; Cao, Z.; Yu, J. Vision-based target-following guider for mobile robot. *IEEE Trans. Ind. Electron.* **2019**, *66*, 9360–9371. [CrossRef]
11. Meng, J.; Wang, S.; Xie, Y.; Li, G.; Zhang, X.; Jiang, L.; Liu, C. A safe and efficient LIDAR-based navigation system for 4WS4WD mobile manipulators in manufacturing plants. *Meas. Sci. Technol.* **2021**, *32*, 045203. [CrossRef]
12. Mur-Artal, R.; Montiel, J.M.M.; Tardos, J.D. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
13. Shan, T.; Englot, B. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 4758–4765.
14. Chung, M.-A.; Lin, C.-W. An improved localization of mobile robotic system based on AMCL algorithm. *IEEE Sens. J.* **2021**, *22*, 900–908. [CrossRef]
15. Li, G.; Meng, J.; Xie, Y.; Zhang, X.; Huang, Y.; Jiang, L.; Liu, C. Reliable and fast localization in ambiguous environments using ambiguity grid map. *Sensors* **2019**, *19*, 3331. [CrossRef] [PubMed]
16. Meng, J.; Wang, S.; Jiang, L.; Hu, Z.; Xie, Y. Accurate and Efficient Self-localization of AGV Relying on Trusted Area Information in Dynamic Industrial Scene. In *IEEE Transactions on Vehicular Technology*; IEEE: Manhattan, NY, USA, 2023.
17. Ge, G.; Zhang, Y.; Wang, W.; Jiang, Q.; Hu, L.; Wang, Y. Text-MCL: Autonomous mobile robot localization in similar environment using text-level semantic information. *Machines* **2022**, *10*, 169. [CrossRef]
18. Zhang, L.; Zapata, R.; Lepinay, P. Self-adaptive Monte Carlo localization for mobile robots using range finders. *Robotica* **2012**, *30*, 229–244. [CrossRef]
19. Sun, H.; Wang, S.; Meng, J.; Liu, Y.; Xie, Y. Accurate Pose Tracking of Mobile Robot Using Entropy-based TrimICP in Dynamic Environment. In Proceedings of the IECON 2022–48th Annual Conference of the IEEE Industrial Electronics Society, Brussels, Belgium, 17–20 October 2022; pp. 1–6.
20. Bukhori, I.; Ismail, Z.H. Detection of kidnapped robot problem in monte carlo localization based on the natural displacement of the robot. *Int. J. Adv. Robot. Syst.* **2017**, *14*, 1729881417717469. [CrossRef]
21. Meng, J.; Wang, S.; Li, G.; Jiang, L.; Zhang, X.; Liu, C.; Xie, Y. Iterative-learning error compensation for autonomous parking of mobile manipulator in harsh industrial environment. *Robot. Comput.-Integr. Manuf.* **2021**, *68*, 102077. [CrossRef]
22. Toomaj, A.; Di Crescenzo, A. Connections between weighted generalized cumulative residual entropy and variance. *Mathematics* **2020**, *8*, 1072. [CrossRef]
23. Zidek, J.V.; Van Eeden, C. Uncertainty, entropy, variance and the effect of partial information. *Lect. Notes-Monogr. Ser.* **2003**, *42*, 155–167.
24. Chen, Y.; Li, Z.; Kong, H.; Ke, F. Model predictive tracking control of nonholonomic mobile robots with coupled input constraints and unknown dynamics. *IEEE Trans. Ind. Inform.* **2018**, *15*, 3196–3205. [CrossRef]
25. Singh, P.; Nandanwar, A.; Behera, L.; Verma, N.K.; Nahavandi, S. Uncertainty compensator and fault estimator-based exponential supertwisting sliding-mode controller for a mobile robot. In *IEEE Transactions on Cybernetics*; IEEE: Manhattan, NY, USA, 2021.
26. Ren, C.; Li, X.; Yang, X.; Ma, S. Extended state observer-based sliding mode control of an omnidirectional mobile robot with friction compensation. *IEEE Trans. Ind. Electron.* **2019**, *66*, 9480–9489. [CrossRef]
27. Liu, X.; Wang, W.; Li, X.; Liu, F.; He, Z.; Yao, Y.; Ruan, H.; Zhang, T. MPC-based high-speed trajectory tracking for 4WIS robot. *ISA Trans.* **2022**, *123*, 413–424. [CrossRef] [PubMed]
28. Dai, L.; Lu, Y.; Xie, H.; Sun, Z.; Xia, Y. Robust tracking model predictive control with quadratic robustness constraint for mobile robots with incremental input constraints. *IEEE Trans. Ind. Electron.* **2020**, *68*, 9789–9799. [CrossRef]
29. Sun, Z.; Xia, Y.; Dai, L.; Liu, K.; Ma, D. Disturbance rejection MPC for tracking of wheeled mobile robot. *IEEE/ASME Trans. Mechatron.* **2017**, *22*, 2576–2587. [CrossRef]
30. Jiang, L.; Xie, Y.; Jiang, Z.; Meng, J.; Li, W. Adaptive model predictive control of mobile robot with local path refitting. In Proceedings of the 2022 IEEE 17th Conference on Industrial Electronics and Applications (ICIEA), Chengdu, China, 16–19 December 2022; pp. 596–601.
31. Li, J.; Wang, J.; Peng, H.; Hu, Y.; Su, H. Fuzzy-torque approximation-enhanced sliding mode control for lateral stability of mobile robot. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *52*, 2491–2500. [CrossRef]
32. Ding, T.; Zhang, Y.; Ma, G.; Cao, Z.; Zhao, X.; Tao, B. Trajectory tracking of redundantly actuated mobile robot by MPC velocity control under steering strategy constraint. *Mechatronics* **2022**, *84*, 102779. [CrossRef]
33. Zhang, Y.; Zhao, X.; Tao, B.; Ding, H. Point stabilization of nonholonomic mobile robot by Bézier smooth subline constraint nonlinear model predictive control. *IEEE/ASME Trans. Mechatron.* **2020**, *26*, 990–1001. [CrossRef]

34. Hu, Y.; Su, H.; Fu, J.; Karimi, H.R.; Ferrigno, G.; De Momi, E.; Knoll, A. Nonlinear model predictive control for mobile medical robot using neural optimization. *IEEE Trans. Ind. Electron.* **2020**, *68*, 12636–12645. [CrossRef]
35. Taheri, H.; Zhao, C.X. Omnidirectional mobile robots, mechanisms and navigation approaches. *Mech. Mach. Theory* **2020**, *153*, 103958. [CrossRef]
36. Wang, J.; Luo, Z.; Wang, Y.; Yang, B.; Assadian, F. Coordination control of differential drive assist steering and vehicle stability control for four-wheel-independent-drive EV. *IEEE Trans. Veh. Technol.* **2018**, *67*, 11453–11467. [CrossRef]
37. Cui, M.; Liu, W.; Liu, H.; Jiang, H.; Wang, Z. Extended state observer-based adaptive sliding mode control of differential-driving mobile robot with uncertainties. *Nonlinear Dyn.* **2016**, *83*, 667–683. [CrossRef]

sensors

MDPI

Article

Indoor Location Technology with High Accuracy Using Simple Visual Tags

Feng Gao * and Jie Ma

College of Mechanical and Vehicle Engineering, Chongqing University, Chongqing 400044, China
* Correspondence: gaofeng1@cqu.edu.cn; Tel.: +86-189-9618-8196

Abstract: To achieve low-cost and robustness, an indoor location system using simple visual tags is designed by comprehensively considering accuracy and computation complexity. Only the color and shape features are used for tag detection, by which both algorithm complexity and data storage requirement are reduced. To manage the nonunique problem caused by the simple tag features, a fast query and matching method is further presented by using the view field of the camera and the tag azimuth. Then, based on the relationship analysis between the spatial distribution of tags and location error, a pose and position estimation method using the weighted least square algorithm is designed and works together with the interactive algorithm by the designed switching strategy. By using the techniques presented, a favorable balance is achieved between the algorithm complexity and the location accuracy. The simulation and experiment results show that the proposed method can manage the singular problem of the overdetermined equations effectively and attenuate the negative effect of unfavorable label groups. Compared with the ultrawide band technology, the location error is reduced by more than 62%.

Keywords: indoor location; visual location; error analysis; weighted least squares

Citation: Gao, F.; Ma, J. Indoor Location Technology with High Accuracy Using Simple Visual Tags. *Sensors* **2023**, *23*, 1597. https://doi.org/10.3390/s23031597

Academic Editors: Yuanlong Xie, Shiqi Zheng and Zhaozheng Hu

Received: 2 January 2023
Revised: 19 January 2023
Accepted: 30 January 2023
Published: 1 February 2023

1. Introduction

With the development of artificial intelligent technologies, intelligent mobile platforms are becoming more widely used in the fields of healthcare, warehousing, logistics, industrial production, etc. [1,2]. Indoor location with high accuracy, as one of the key technologies, is the basis for autonomous functions such as navigation and decision. Compared with other application scenarios, it has special requirements on computation complexity, convenience of deployment, and robustness to environmental disturbances. Outdoors, as a mature technology, satellites can provide accurate location and timing signals at any time. In doors, however, the satellite signal becomes ineffective because of attenuation caused by buildings, interference during transmission, and multipath effect [3].

The indoor location technology can be divided into two types according to the signals used, i.e., wireless signal and visual information. The widely used wireless signals include WIFI [4], Bluetooth [5] and UWB (ultrawide band) [6], etc. Indoor location can be realized by measuring the distance, angle, or location fingerprint [3–6]. However, the cost to deploy base stations is comparatively high. Recently, with the development of Internet-of-Things technologies, some researchers devote themselves to indoor location using wireless local networks, which have already been deployed in many indoor environments [7,8]. Alhammadi et al. developed a three-dimensional indoor location system based on the Bayesian graphical model [9]. However, the average location error is more than 3 m and it can hardly be used in autonomous driving systems of intelligent mobile platforms. Moreover, there exists a multipath effect for the wireless communication signals. The visual location mainly includes SLAM (simultaneous location and mapping) [10] and location by visual tags [11]. SLAM can be applied to unknown environments by extracting and matching the texture feature automatically, but the algorithm has a higher computation

complexity and its performance is also easily degraded by environmental disturbances. These pose great challenges on practical applications, especially for the scenario with a high demand on real-time and robust performance. Comparatively, the location algorithm using visual tags can achieve low computation complexity by prior storage of the feature and position information of tags. The selection of visual tags and pose calculation are important to location performance.

For the selection of visual tags, Ref. [12] uses special objects such as signs, desks, and doors, whose SURF feature is extracted to form the offline tag database. This method need not change the environment, but the information of texture features is still very large and it is also easily affected by environmental factors such as light. To reduce the storage requirement, only roof corners are considered in [13]. In [14], semantic segmentation technology is combined with the location algorithm for better performance, but much computing resource is required by the semantic segmentation process. No extra effort is needed to reconstruct the environment if using these already existing indoor objects. For the same type of object, however, their texture features are similar, which makes it difficult to discriminate different tags. To manage this problem, special tags are designed by coding the information of ID, coordinate, etc., in the tag to ensure its uniqueness [11]. On the contrary, the complexity of a QR code requires more time for detection, which even reaches hundreds of milliseconds [15]. Moreover, the texture information of complex QR codes is also easily degraded by environmental light and contaminants such as dust and dirt.

For the pose calculation, image matching and geometrical constraint are two widely used methods. In [11], the geometrical relationship of view between the tag image recorded in the database and that acquired in real time is used to calculate the position. Additionally, template matching is adopted in [16] to realize location by affine transformation. The fundamental of image matching is to find key texture features and establish their relationships between different images. This leads to a high degree of algorithm complexity. Comparatively, the computation of the algorithm using geometrical constraint is much less. It realizes location by geometrical relationship between the tags and sensors. The location precision reaches centimeter level by using a binocular camera for the detection of distance [13]. However, the computation resource required for the distance measurement by stereovision is very high [17]. In [18], the direct measured distance from multiple tags is used to construct overdetermined equations according to geometrical constraints, and the location is estimated by LSM (least square method). This approach has better comprehensive advantage on both location accuracy and computation complexity. When the redundancy of tags is not enough or the spatial distribution is special, however, the equations will become singular, which leads to a sharp increase in location error [19].

Considering the aforementioned problems, an indoor location system only using simple visual tags is designed for the autonomous driving system of intelligent mobile platforms. To manage the nonunique problem caused by the simple texture feature of tags, a fast query and matching algorithm is designed by using the FOV (field of view) of the camera and the tag azimuth. Furthermore, based on the relationship analysis between the spatial distribution of tags and location error, a pose and position estimation algorithm using WLSM (weighted least square method) is designed. Further, it works together with the interactive algorithm by the designed switching strategy to adapt the singular condition. A favorable balance has been achieved between the algorithm complexity and location accuracy by the presented techniques. The effectiveness of the location system designed herein is validated by both simulation and experiments, and compared with UWB, the location error is reduced greatly. The main contributions of this study are summarized as follows:

(1) A fast query and matching algorithm is designed to manage the nonunique problem of tags, which cannot be distinguished directly by their texture features.

(2) A cooperative strategy is proposed to combine WLSM and the interactive algorithm together to realize the estimation of pose and position with high accuracy under both general and singular conditions.

(3) An indoor location system only using simple visual tags is developed, which has the advantages of low consumption of computation resources, high accuracy, easy deployment, and robustness to environmental disturbances.

(4) The effectiveness of the proposed indoor location system is validated by both simulation and experiment tests.

The remainder of this paper is organized as follows: Section 2 describes the indoor location system design. The tag-matching strategy and the position calculation algorithm are introduced in Sections 3 and 4, respectively; the effectiveness of the designed location system is validated by numerical simulations and comparative tests in Section 5; and Section 6 concludes the paper.

2. Indoor Location System Design

The visual tags and calculation algorithm of position are the two main parts in the designed system.

2.1. Simple Visual Tag Design

Based on the following reasons, the tags shown in Figure 1 are designed [20]:

(1) It is difficult to control the natural texture, which is also easily influenced by environmental factors;

(2) Tagging with a code is unique, but the detection algorithm is more complex and its sophisticated texture is also easily degraded;

(3) Cameras have strong capability to measure colors, and RGB are the three primary colors;

(4) It is easy to detect a circle, which also has the advantage of fine detection robustness and invariance from different views.

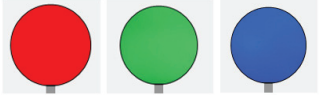


Figure 1. Simple visual tags.

In this study, the tag is detected by the color and shape features [21], and the azimuth is measured by the tag's center. In this way, better anti-interference capability can be achieved with less computation complexity. Only using these simple tag features, however, we cannot discriminate them directly. Accordingly, the pose and position cannot be calculated directly from the tag coordinates.

2.2. Location Algorithm with High Accuracy

To manage the nonunique problem of the simple visual tags, the location algorithm shown in Figure 2 is proposed after considering the advantages of the geometrical constraint method, i.e., high location accuracy and low algorithm complexity.

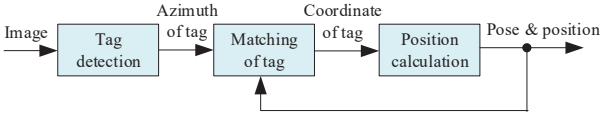


Figure 2. Location algorithm with high accuracy.

Considering that most intelligent platforms move indoors within the horizontal plane, the angle measurement method by vision is used to measure the tag position, as shown in Figure 3.

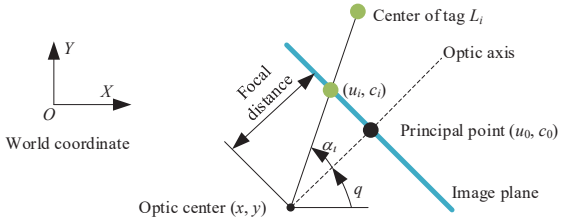


Figure 3. Angle measurement by vision.

The optic center of camera (x, y), azimuth θ, and the tag position (x_i, y_i) satisfy [18,19]:

$$\tan(\theta + \alpha_i) = \frac{y_i - y}{x_i - x}, \ \alpha_i = \arctan\frac{u_i - u_0}{f}, \ i \in \Omega \tag{1}$$

where f is the focal distance and Ω is composed of the tag identifier, which can be observed by the camera. If the tag feature is unique and its coordinate is known, the pose and position can be solved from (1) directly, when the number of observed tags is no smaller than three. Information from more tags is beneficial to location accuracy and robustness [18]. Unfortunately, the designed tag only has color and shape features, which are nonunique. In the next section, a fast query and matching algorithm is designed by using the FOV of the camera and the tag azimuth. In this way, the matched tag can be found without searching the entire database. Then, with the coordinate of observed tags, a hybrid algorithm is presented in Section 3 to estimate the pose and position by combing WLSM and the interactive algorithm to create a favorable balance between algorithm complexity and location accuracy.

3. Matching of Tag

Considering the following reasons, the matching algorithm shown in Figure 4 is designed:

(1) The camera only can detect the tags in its FOV, which can be used to quickly search the database to determine the candidate tags for better efficiency;

(2) When the observed tag is consistent with a candidate tag, their azimuths should be exactly equal. This fact can be used to realize tag matching.

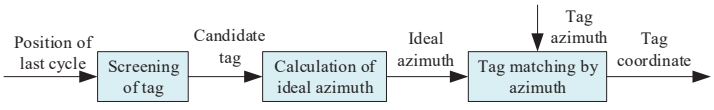


Figure 4. Matching process for observed and stored tags.

3.1. Determination of Candidate Tag

To determine the candidate tag quickly, the location result from the last cycle is used to estimate the current pose and position by DR (dead rocking) [22]. Then, according to the FOV model shown in Figure 5, the possible area where the candidate tag may exist can be determined. On the contrary, a tag locating outside this area cannot be the candidate tag.

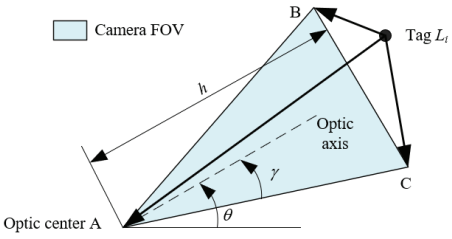


Figure 5. FOV model of camera.

In Figure 5, the coordinate of A is calculated by DR [22], and the coordinates of B and C, i.e., (x_B, y_B) and (x_C, y_C), are calculated by their geometrical relationship:

$$\begin{bmatrix} x_B \\ y_B \end{bmatrix} = \begin{bmatrix} x_A \\ y_A \end{bmatrix} + \frac{h}{\cos\gamma}\begin{bmatrix} \cos(\theta+\gamma) \\ \sin(\theta+\gamma) \end{bmatrix} \text{ and } \begin{bmatrix} x_C \\ y_C \end{bmatrix} = \begin{bmatrix} x_A \\ y_A \end{bmatrix} + \frac{h}{\cos\gamma}\begin{bmatrix} \cos(\theta-\gamma) \\ \sin(\theta-\gamma) \end{bmatrix} \quad (2)$$

where h is the detection range and γ is the half horizontal view angle.

To improve the efficiency, the whole location area is rasterized to ensure that only one tag is allowed in one grid at most, and the stored tag information can be indexed by the grid number directly. According to the FOV model shown in Figure 5, the candidate tags are determined by the following steps:

Step 1: The grids covered by the camera FOV are determined by the coordinates of A, B, and C, which are calculated by (2).

Step 2: According to the grid number covered by the camera FOV, the possible tags are roughly selected from the database.

Step 3: For each tag L_i selected in Step 2, whether it lies in FOV is judged accurately by the sign of pairwise dot products between the vectors, $\overrightarrow{L_iA}$, $\overrightarrow{L_iB}$, and $\overrightarrow{L_iC}$.

Step 4: The set of candidate tags Ω_c is composed of all tags whose pairwise dot products have the same sign. Additionally, the ideal azimuth of tags β_i, $i \in \Omega_c$ is calculated by (1) according to the camera's pose and position derived by DR.

3.2. Tag Matching by Azimuth

Considering the fact that when the candidate tag is consistent with the observed tag, their azimuths should be equal, we design the following strategy for tag matching:

Step 1: For any observed and unmatched tag, the error e_i between its azimuth α and that of the candidate tag with the same color and shape feature is calculated by

$$e_i = |\alpha - \beta_i|,\ i \in \Omega_c. \quad (3)$$

Step 2: If the minimum e_i is smaller than the threshold, the candidate tag L_k, $k = \underset{i}{\operatorname{argmin}} e_i$ matches the observed tag. Otherwise, it is considered as a disturbance to be deleted.

Compared with the method using all feature information to perform the match by total optimization, this strategy can effectively avoid the matching error caused by the incomplete information of unobserved tags, which may be shaded by other objects, etc. With the aforementioned process, the coordinate of observed tags can be retrieved from the tag database quickly and accurately.

4. Pose and Position Calculation with High Accuracy

Theoretically, the pose and position can be calculated by (1) when the number of effective tags is not smaller than three [18]. In practical application, the following factors will degrade the location accuracy:

(1) The pixel error caused by factors such as environmental disturbances, camera pose, and tag position, is unfavorable for location accuracy.

(2) When solving the pose and position by (1), there exists matrix inversion. Under some special conditions, singularity causes the matrix to be invertible.

To manage these problems, a WLSM is designed to estimate the pose and position with high accuracy by analysis of the error transfer characteristics using (1), and furthermore, a cooperation strategy is presented to schedule WLSM and the interactive algorithm to adapt to the singular condition.

4.1. Analysis of Error Transfer Characteristic

Considering the fact that the pose and position can be calculated with three effective tags, the following error transfer characteristic is obtained by the sensitive analysis of (1):

$$\begin{bmatrix} \mathrm{d}\alpha_i \\ \mathrm{d}\alpha_j \\ \mathrm{d}\alpha_k \end{bmatrix} = G \begin{bmatrix} \mathrm{d}x \\ \mathrm{d}y \\ \mathrm{d}\theta \end{bmatrix}, \quad G = \begin{bmatrix} \frac{\sin(\alpha_i+\theta)}{r_i} & \frac{-\cos(\alpha_i+\theta)}{r_i} & -1 \\ \frac{\sin(\alpha_j+\theta)}{r_j} & \frac{-\cos(\alpha_j+\theta)}{r_j} & -1 \\ \frac{\sin(\alpha_k+\theta)}{r_k} & \frac{-\cos(\alpha_k+\theta)}{r_k} & -1 \end{bmatrix}. \tag{4}$$

where α_i and r_i are the measured azimuth of the tag L_i and its distance to the camera. When the camera and the tags are on the same circle, as shown in Figure 6, we have the following equations according to Ptolemy theorem and law of sines:

$$r_i d_{jk} + r_k d_{ij} = r_j d_{ik}, \quad \frac{d_{ij}}{\sin(\alpha_i - \alpha_j)} = \frac{d_{jk}}{\sin(\alpha_j - \alpha_k)} = \frac{d_{ik}}{\sin(\alpha_i - \alpha_k)}. \tag{5}$$

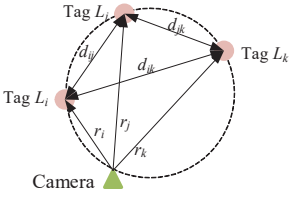


Figure 6. Schematic for cocircular condition.

Under this condition, it is known from (5) that $|G| = 0$ and the position equations become ill-conditioned. This implies that a small error in the measured azimuth will result in significant deviation in the solved pose and position. Accordingly, the following cooperation strategy is designed. When the condition number of G is smaller than the threshold, the pose and position is calculated by WLSM as it is given in Section 4.2. Otherwise, the following interactive searching algorithm is used:

Step 1: The possible azimuth range is discretized according to the requirement of accuracy.

Step 2: For each discretized azimuth, the camera coordinate can be calculated by the following equation according to the information of any two tags:

$$\begin{aligned} x &= \frac{(t_i t_\theta - 1)(t_j t_\theta - 1)(y_i - y_j) + (t_j t_\theta - 1)(t_\theta + t_i)x_i - (t_i t_\theta - 1)(t_\theta + t_j)x_j}{(t_\theta^2 + 1)(t_j - t_i)} \\ y &= \frac{(t_\theta + t_i)(t_\theta + t_j)(x_i - x_j) + (t_i t_\theta - 1)(t_\theta + t_j)y_i - (t_j t_\theta - 1)(t_\theta + t_i)y_j}{(t_\theta^2 + 1)(t_i - t_j)} \\ t_i &= \tan\alpha_i, \quad t_j = \tan\alpha_j, \quad t_\theta = \tan\theta \end{aligned} \tag{6}$$

where (x_i, y_i) and α_i are the stored position of tag L_i and its measured azimuth, respectively, and θ is the discretized azimuth.

Step 3: For each discretized azimuth, the variance of camera's coordinate is calculated. The coordinate with the smallest scattered degree and the corresponding azimuth are then selected as the effective pose and position value.

4.2. Position Estimation by Weighted Least Square Method

Under the noncircular conditions, the pose and position can be solved by every three tags. It has the advantage of fine real-time performance and good robustness to noise to solve the aforementioned overdetermined equations by LSM [23]. However, the location error from different tag groups is influenced by many factors, such as environments, image resolution, etc. To further improve the location accuracy, a weighted least square estimation algorithm is presented by designing the evaluation index of solution quality.

Taking the estimation of θ as an example, the following equation is derived for any group of tags, L_i, L_j, and L_k, from the geometrical constraint described by (1) [18]:

$$\begin{aligned} \sigma_{t_\theta}^2 &= \frac{\begin{bmatrix} \frac{\partial t_\theta}{\partial t_i} & \frac{\partial t_\theta}{\partial t_j} & \frac{\partial t_\theta}{\partial t_k} \end{bmatrix} \begin{bmatrix} \frac{\partial t_\theta}{\partial t_i} & \frac{\partial t_\theta}{\partial t_j} & \frac{\partial t_\theta}{\partial t_k} \end{bmatrix}^{\mathrm{T}}}{f^2} \sigma_u^2, \\ t_\theta &= \frac{(t_k - t_i)\left(-y_j + t_j x_k + y_i - t_i x_i\right) - \left(t_j - t_i\right)\left(-y_k + t_k x_k + y_i - t_i x_i\right)}{(t_k - t_i)\left(-x_j - t_j y_j + x_i + t_i y_i\right) - \left(t_j - t_i\right)\left(-x_k + t_k y_k + x_i - t_i y_i\right)} \end{aligned} \tag{7}$$

where $\sigma_{t_\theta}^2$ and σ_u^2 are the variances of the tangent value of θ and the pixel coordinate u. Equation (7) establishes the relationship between the pixel error of the tag center and the azimuth error.

When using WLSM, the maximum likelihood solution can be obtained by setting the weight as the reciprocal of error variance [23]. From the error relationship described by (7), the WLSM used to optimally estimate the azimuth is designed as:

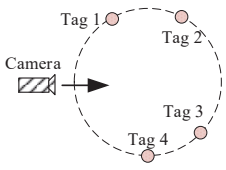
$$\begin{aligned} \theta &= \operatorname{atan} t_\theta, \;\; t_\theta = \left(A^{\mathrm{T}} W A\right)^{-1} A^{\mathrm{T}} W B, \\ A &= \begin{bmatrix} a_1 & a_2 & \cdots \end{bmatrix}^{\mathrm{T}}, B = \begin{bmatrix} b_1 & b_2 & \cdots \end{bmatrix}^{\mathrm{T}}, W = \operatorname{diag}(1/\omega_1, 1/\omega_2, \cdots), \\ a_n &= (t_k - t_i)\left(x_i - x_j + \begin{bmatrix} t_i & -t_j \end{bmatrix} \begin{bmatrix} y_i \\ y_j \end{bmatrix}\right) - \left(t_j - t_i\right)\left(x_i - x_k + \begin{bmatrix} t_i & -t_k \end{bmatrix} \begin{bmatrix} y_i \\ y_k \end{bmatrix}\right), \\ b_n &= (t_k - t_i)\left(y_i - y_j + \begin{bmatrix} -t_i & t_j \end{bmatrix} \begin{bmatrix} x_i \\ x_j \end{bmatrix}\right) - \left(t_j - t_i\right)(y_i - y_k + \begin{bmatrix} -t_i & t_k \end{bmatrix} \begin{bmatrix} x_i \\ x_k \end{bmatrix}, \\ \omega_n &= \begin{bmatrix} \frac{\partial t_\theta}{\partial t_i} & \frac{\partial t_\theta}{\partial t_j} & \frac{\partial t_\theta}{\partial t_k} \end{bmatrix} \begin{bmatrix} \frac{\partial t_\theta}{\partial t_i} & \frac{\partial t_\theta}{\partial t_j} & \frac{\partial t_\theta}{\partial t_k} \end{bmatrix}^{\mathrm{T}}. \end{aligned} \tag{8}$$

5. Validation and Analysis

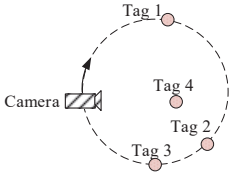
In this section, following validation, the effectiveness of the proposed method by numerical simulations is further compared with UWB by experiments.

5.1. Simulation Validation and Analysis

The location scenarios shown in Figure 7 are designed to indicate the effectiveness of the designed cooperation strategy and WLSM. Under condition (a), all tags locate on the same circle, which is used to validate the cooperative strategy. Under condition (b), tags 1, 2, and 3 are on the same circle, but tag 4 is not, and the camera moves along the circle. This condition is used to validate that the designed WLSM can attenuate the negative influence from the unfavorable tag group. The scenario is simulated in Prescan and the designed location algorithm runs in Matlab. Prescan provides the integrated interfaces for cosimulation with Matlab [24]. During the simulation, the time step is set to be 0.03 s, the radius of the circular trajectory is 5 m, and the moving speed of the camera is about 0.5 m/s.



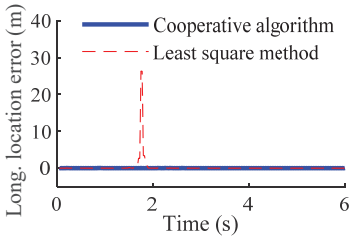
(**a**) Cocircular condition



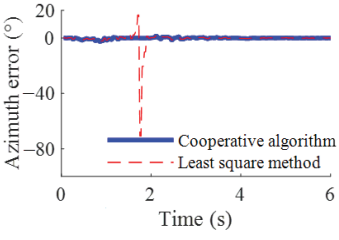
(**b**) Noncircular condition

Figure 7. Simulation conditions.

Shown in Figure 8 are the location results when the camera moves from the outside of the circle along the arrow direction shown in Figure 7a. At about 1.3~1.9 s, the camera and tags are on the same circle. The overdetermined Equation (1) becomes singular and the location error of LSM increases quickly to a large value. On the contrary, the proposed cooperative algorithm can detect the singularity and switch to the interactive algorithm as soon as possible. Accordingly, the location accuracy can be ensured during the whole simulation process.



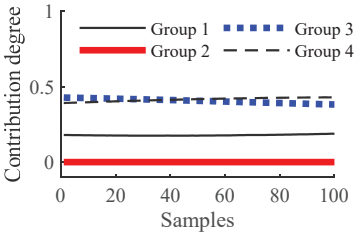
(**a**) Longitudinal location error



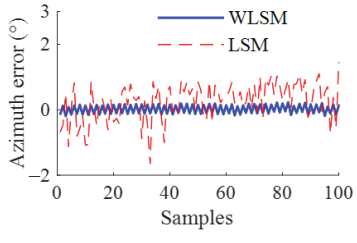
(**b**) Azimuth error

Figure 8. Location results under condition (a).

Since pose and position can be solved with any 3 tags, under condition (b), 4 tags can generate 4 combinations with every 3 tags. The combination of tags 2, 3, and 4 is denoted by group 1, the combination of tags 1, 2, and 3 is denoted by group 2, the combination of tags 1, 3, and 4 is denoted by group 3, and the combination of tags 1, 2, and 4 is denoted by group 4. Among them, the tags in group 2 and the camera are always on the same circle. The solution quality for group 2 is worst. It is found from Figure 9a that the contribution from group 2 on the location results is almost zero. This shows that the designed index to evaluate the solution quality given in Section 4.2 can measure the confidence level of results from different tag groups. Accordingly, the presented WLSM using this index as the weight can successfully attenuate the negative influence from unfavorable tag groups. Compared with LSM, it can be found from Figure 9c,d that the maximum errors of azimuth, longitudinal, and lateral location results are reduced by 89.47%, 51.61%, and 88.78%, respectively.

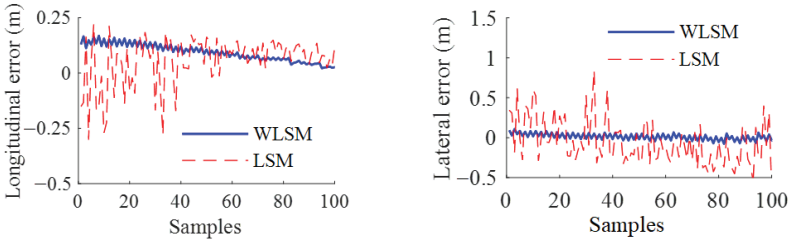


(**a**) Contribution degree



(**b**) Azimuth error

Figure 9. *Cont.*



(**c**) Longitudinal location error (**d**) Lateral location error

Figure 9. Location results under condition (b).

5.2. Comparative Test of Location Accuracy

To further test the location accuracy, the designed location system is compared with UWB under the indoor condition shown in Figure 10 considering the following reasons: (1) Methods such as ORB-SLAM have a high requirement for computation resources. (2) The visual methods using complex texture features can hardly work under such disturbed conditions because some main features are covered or light obviously changes. (3) The average error of the methods using location fingerprint of wireless signals is up to several meters. (4) Comparatively, UWB has the advantages of low computation complexity, high accuracy, and robustness to environmental disturbances such as light and dynamic covering of objects. The room size is about 4.1 m × 5.2 m, where there are several desks, which may degrade the location performance of UWB. About 20 tags are stuck on the walls around the room, and 4 UWB stations are used. To simulate the environmental disturbances, the light is turned on/off and the tags are covered randomly by some objects during the test.

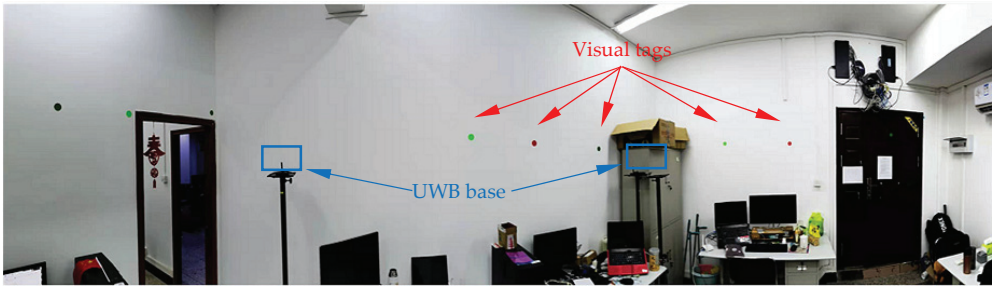


Figure 10. Indoor location scenario.

During the test, the location algorithm runs in a Raspberry 4b board with the rate 30 fps. The board with the camera is held by hand and moves along the defined trajectory as best we can with the speed of about 0.5 m/s. The main test equipment with their technical parameters is listed in Table 1. The camera is used to detect the tag, and in addition to the algorithm, the location results from both the proposed method and UWB are recorded by the Raspberry 4b.

Table 1. Test equipment and technical parameters.

Equipment	Parameter	Value
Camera	Resolution	1920 × 1080
	Frame rate	30 fps
	Field of view	78 degree
	Focal length	747.1 pix
Raspberry 4b	CPU	BCM2837B0SOC
	Number of cores	4
	Basic frequency	1.4 GHz
	RAM	1 G
UWB	Location precision	10 cm
	Communication range	≤130 m
	Update frequency	≤50 Hz

The location results are shown in Figure 11 when the camera moves along the defined trajectory.

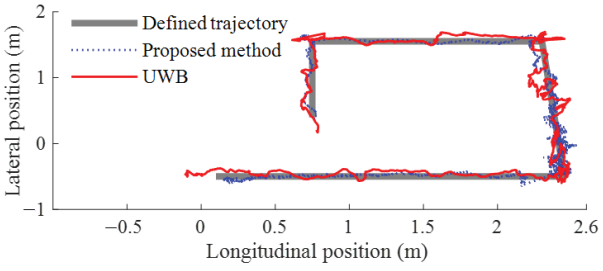


Figure 11. Comparative test results.

The average location error of the results is shown in Table 2. Compared with UWB, the longitudinal and lateral location error of the designed system is reduced by 65.93% and 69.75%, respectively.

Table 2. Statistical location errors.

Average Error	Long. Location (cm)	Lat. Location (cm)	Azimuth (°)
UWB	36.1	32.4	/
Proposed location system	12.3	9.8	0.88

Since it is difficult to control the node to move along the trajectory accurately and strictly, the statistical error in Table 2 is larger than their actual values. Ten test points are selected where the location results are measured statically for accuracy, and the results are shown in Figure 12. On the whole, the average location error of UWB is more than 13.5 cm, reaching a maximum of 49 cm at position (1.61 cm, −0.50 cm). The wireless signal of UWB is reflected by the objects in the room, which degrades the location performance of UWB by multipath effect. Even though with the simulated environmental disturbances of light and covering, the average error of the proposed location system is about 5.5 cm and the maximum error is only 9 cm. The location accuracy is obviously improved by the proposed techniques in this study.

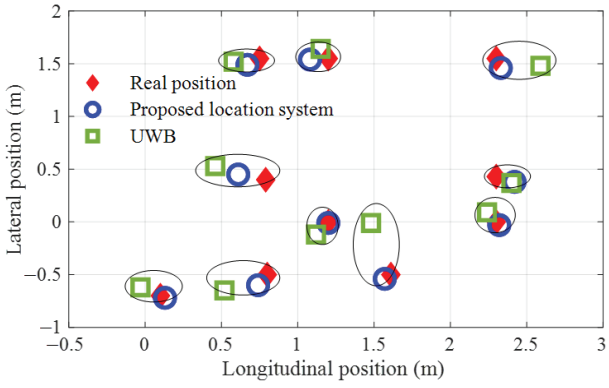


Figure 12. Comparative test results under statistical conditions.

6. Conclusions

To realize indoor locations with high accuracy, low cost, and robustness, an indoor location system is designed in this study by presenting a fast tag-matching strategy and cooperative-solving algorithm. The system locates according to simple visual tags, which are robust to environmental disturbances and easily deployed. The cooperative-solving algorithm for estimation of pose and position can work under cocircular conditions, and compared with LSM, the maximum errors of azimuth, longitudinal, and lateral location results are reduced by 89.47%, 51.61%, and 88.78%, respectively. The average location error for the designed location system under disturbed environments is about 5.5 cm, and the location accuracy is higher than UWB by 62%.

Author Contributions: Conceptualization and methodology, F.G.; validation J.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the State Key Laboratory of Automotive Safety and Energy under grant KFY2209.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data are available upon request to the corresponding author.

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

References

1. Vladareanu, L.; Vlădareanu, V.; Pop, N.; Migdalovici, M.; Boșcoianu, M.; Pop, S.; Ciocîrlan, A. Robot extenics control developed by versatile, intelligent and portable robot Vipro platform applied on firefighting robots. *Int. J. Online Biomed. Eng.* **2020**, *16*, 99–107. [CrossRef]
2. Gao, F.; Dang, D.F.; He, Y.D. Robust coordinated control of nonlinear heterogeneous platoon interacted by uncertain topology. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 4982–4992.
3. Li, S.; Rashidzadeh, R. Hybrid indoor location positioning system. *IET Wirel. Sens. Syst.* **2019**, *9*, 257–264. [CrossRef]
4. Ding, J.; Wang, Y.; Fu, S.; Si, H.; Zhang, J.; Gao, S. Multiview features fusion and Adaboost based indoor localization on Wifi platform. *IEEE Sens. J.* **2022**, *22*, 16607–16616. [CrossRef]
5. Youm, S.; Shin, K.S. Improved method of Bluetooth low energy based location tracking using neural networks. *Sens. Mater.* **2021**, *33*, 2083–2093. [CrossRef]
6. Liu, F.; Zhang, J.; Wang, J.; Han, H.; Yang, D. An UWB/vision fusion scheme for determining pedestrians' indoor location. *Sensors* **2020**, *20*, 1139. [CrossRef] [PubMed]
7. Billa, A.; Shayea, I.; Alhammadi, A.; Abdullah, Q.; Roslee, M. An overview of indoor localization technologies: Toward IoT navigation services. In Proceedings of the International Symposium on Telecommunication Technologies, Shah Alam, Malaysia, 9–11 November 2020.

8. Kordi, K.A.; Alhammadi, A.; Roslee, M.; Alias, M.Y.; Abdullah, Q. A review on wireless emerging IoT indoor localization. In Proceedings of the International Symposium on Telecommunication Technologies, Shah Alam, Malaysia, 9–11 November 2020.
9. Alhammadi, A.; Hashim, F.; ARasid, M.F.; Alraih, S. A three-dimensional pattern recognition localization system based on a Bayesian graphical model. *Int. J. Distrib. Sens. Netw.* **2020**, *16*, 1–12. [CrossRef]
10. Song, J.; Kook, J. Visual SLAM based spatial recognition and visualization method for mobile AR system. *Appl. Syst. Innov.* **2022**, *5*, 11. [CrossRef]
11. Kayhani, N.; Zhao, W.; McCabe, B.; Schoellig, A.P. Tag-based visual-inertial localization of unmanned aerial vehicles in indoor construction environments using an on-manifold extended Kalman filter. *Autom. Constr.* **2022**, *135*, 104112. [CrossRef]
12. Li, M.; Chen, R.; Liao, X.; Guo, B.; Zhang, W.; Guo, G. A precise indoor visual positioning approach using a built image feature database and signle user image from smartphone. *Remote Sens.* **2020**, *12*, 869. [CrossRef]
13. Hong, W.; Xia, H.; An, X.; Liu, X. Natural landmarks based localization algorithm for indoor robot with binocular vision. In Proceedings of the Chinese Control and Decision Conference, Chongqing, China, 28–30 May 2017.
14. Li, N.; Tu, W.; Ai, H.; Deng, H.; Tao, J.; Hu, T.; Sun, X. VISEL: A visual and magnetic fusion-based large scale indoor localization system with improved high-precision sematic maps. *Int. J. Intell. Syst.* **2022**, *37*, 7992–8020. [CrossRef]
15. Park, D.; Lee, S.J.; Kwon, J.; Choi, S. Localization of mobile robot based on artificial landmark. *J. Korean Inst. Commun. Inf. Sci.* **2020**, *45*, 2150–2153. [CrossRef]
16. Li, M.; Zhao, L.; Tan, D.; Tong, X. BLE fingerprint indoor localization algorithm based on eight-neighborhood template matching. *Sensors* **2020**, *19*, 4859. [CrossRef]
17. Shao, F.; Lin, W.; Gu, S.; Jiang, G.; Srikanthan, T. Perceptual full-reference quality assessment of stereoscopic images by considering binocular visual characteristics. *IEEE Trans. Image Process.* **2013**, *22*, 1940–1953. [CrossRef]
18. Xie, H.L.; Zhu, Q.D.; Liu, P. Self localization method of panoramic stereo vision robot based on maximum likelihood estimation. *Appl. Technol.* **2017**, *44*, 40–45.
19. Jiang, L.Z.; Lei, W.; Wang, J. Radar angle measurement accuracy evaluation system based on machine vision. *Comput. Meas. Control.* **2014**, *22*, 2842–2843, 2866.
20. Lee, J.K.; Wang, Y.M.; Lu, C.S.; Wang, H.C.; Chou, T.R. The enhancement of graphic QR code recognition using convolutional neural networks. In Proceedings of the International Conference on Innovation, Communication and Engineering, Zhenzhou, China, 25–30 October 2019.
21. Gao, F.; Wang, C. Hybrid strategy for traffic light detection by combining classical and self-learning detectors. *IET Intell. Transp. Syst.* **2020**, *14*, 735–741. [CrossRef]
22. Nazemzadeh, P.; Fontanelli, D.; Macii, D. Optimal placement of landmarks for indoor localization using sensors with a limited range. In Proceedings of the International Conference on Indoor Positioning and Indoor Navigation, Alcala de Henares, Spain, 4–7 October 2016.
23. Chen, Y.; Hu, J.; Wang, J.H. Data driven weighted least square localization algorithm for UAV. *Command. Inf. Syst. Technol.* **2020**, *11*, 76–80.
24. Gao, F.; Han, Z.; Yang, Y. Performance Limit Evaluation by Evolution Test with Application to Automatic Parking System. *IEEE Trans. Intell. Veh.* 2022. Available online: https://ieeexplore.ieee.org/document/9645203 (accessed on 19 January 2023).

sensors

MDPI

Article

DCP-SLAM: Distributed Collaborative Partial Swarm SLAM for Efficient Navigation of Autonomous Robots

Huma Mahboob [1,*], Jawad N. Yasin [1,2], Suvi Jokinen [1], Mohammad-Hashem Haghbayan [1], Juha Plosila [1] and Muhammad Mehboob Yasin [3]

[1] Autonomous Systems Laboratory, Department of Future Technologies, University of Turku, Vesilinnantie 5, 20500 Turku, Finland
[2] ABB Oy, 00380 Helsinki, Finland
[3] Department of Computer Networks, College of Computer Sciences & Information Technology, King Faisal University, Hofuf 31982, Saudi Arabia
* Correspondence: humahb@utu.fi

Abstract: Collaborative robots represent an evolution in the field of swarm robotics that is pervasive in modern industrial undertakings from manufacturing to exploration. Though there has been much work on path planning for autonomous robots employing floor plans, energy-efficient navigation of autonomous robots in unknown environments is gaining traction. This work presents a novel methodology of low-overhead collaborative sensing, run-time mapping and localization, and navigation for robot swarms. The aim is to optimize energy consumption for the swarm as a whole rather than individual robots. An energy- and information-aware management algorithm is proposed to optimize the time and energy required for a swarm of autonomous robots to move from a launch area to the predefined destination. This is achieved by modifying the classical Partial Swarm SLAM technique, whereby sections of objects discovered by different members of the swarm are stitched together and broadcast to members of the swarm. Thus, a follower can find the shortest path to the destination while avoiding even far away obstacles in an efficient manner. The proposed algorithm reduces the energy consumption of the swarm as a whole due to the fact that the leading robots sense and discover respective optimal paths and share their discoveries with the followers. The simulation results show that the robots effectively re-optimized the previous solution while sharing necessary information within the swarm. Furthermore, the efficiency of the proposed scheme is shown via comparative results, i.e., reducing traveling distance by 13% for individual robots and up to 11% for the swarm as a whole in the performed experiments.

Keywords: swarm robotics; collaborative sensing; multi-agent systems; energy efficient; swarm intelligence; leader–follower; collision avoidance

Citation: Mahboob, H.; Yasin, M.M.; Yasin, J.N.; Haghbayan, M.-H.; Plosila, P. DCP-SLAM: Distributed Collaborative Partial Swarm SLAM for Efficient Navigation of Autonomous Robots. *Sensors* **2023**, *23*, 1025. https://doi.org/10.3390/s23021025

Academic Editors: Yuanlong Xie, Shiqi Zheng and Zhaozheng Hu

Received: 24 December 2022
Revised: 13 January 2023
Accepted: 14 January 2023
Published: 16 January 2023

1. Introduction

Swarm robotics is a field inspired by natural self-organizing swarms, such as birds, bees, ants, and fish [1]. The aim of researchers is to create swarms of autonomous robots that can mimic such self-organizing behavior in different situations to carry out their collective mission in an energy-efficient manner [2]. Each robot in the swarm executes relatively simple control routines to accomplish its task. It uses its onboard sensors for awareness of the surrounding environment, whereas it relies on wireless messaging for coordination with other robots of its formation, akin to behavior of individual agents in a multi-agent system [3]. In the following text, the terms agent and robot are used interchangeably. Due to their small size, robustness, and ability to reach difficult or hazardous environments, there has been exponential increase in research for the development of novel techniques for the integration of autonomous robots in various applications, such as surveillance [4], infrastructure inspection [5], military applications [6], GPS-denied envi-

ronments [7], transportation [8], hazardous environments [9], and mapping or atmospheric research [10].

Of particular interest is the problem of navigation of a swarm of autonomous robots in an unknown environment, such as a GPS-denied area or where no prior map information exists. Work in this field presents a wide array of research challenges, such as the ability to maintain formation, self-localization, collision avoidance, and path finding [3]. Simultaneous localization and mapping (SLAM) is a classical and fundamental technique for mapping and localizing in the field of autonomous robots in environments with no prior map information. Until relatively recently, SLAM development was mainly concerned with a single agent or robot, hence leaving a significant gap for the development of SLAM techniques with multiple agents or a swarm of robots [11,12]. Various collaborative SLAM (C-SLAM) methodologies and techniques have been researched, where the primary objective has been to integrate the data gathered by individual robots to build a global map that is shared between all robots [13]. In collaborative navigation, the actions of one agent may impact those of other agents in the system; therefore, it is vital to realize how the collaborative navigation can be of assistance. The coordination strategies to attain autonomy for a swarm of agents can be categorized into the following three types [14,15]: (1) centralized coordination, in which either a central node or server or an established agent, often labeled as a dedicated leader, provides the vital parameters to the rest of the agents in the swarm, such as maneuver information, trajectory planning, and coordination; these are often described as virtual structure-based approaches or leader–follower-based approaches [16]; (2) decentralized with coordination-based approaches, in which all agents in the swarm can directly interact with other agents within their communication range; (3) decentralized without coordination-based approaches, in which the agents do not interact with each other to interchange the individually gathered data; in this approach, the agents work on the observe and react principle [17].

The sensing modes utilized for C-SLAM can be divided into two broad categories, namely,vision-based and laser/LiDAR-based. Due to their high resolution, robustness to varying weather conditions, and resilience against lighting conditions, several LiDAR C-SLAM techniques have been investigated [18–22]. Approaches presented in [18,20] work by transmitting the locally generated maps of all the robots to a central base station/server for optimization and stitching purposes. This process requires high availability of the central server, the failure of which results in the failure of the entire mission. Furthermore, this approach is very dependent on the communication channel between each robot and the server, requiring it to be lossless and have high bandwidth. To alleviate the dependence on a lossless communication channel, ref. [21] presented a three-dimensional LiDAR-based data-driven descriptor approach to optimize the required transmission bandwidth. The presented methodology functions on a completely centralized system; it is not suitable for larger swarms. Most of the present C-SLAM-based approaches are either highly centralized or the database is split into several segments for assessment, have high computational costs, or require higher communication bandwidth.

Minimization of the energy consumption of a swarm as a whole is another important research area with core emphasis on a varied set of themes, such as dealing with external influences [23,24], optimization of consumption due to ranging sensors [25], efficient inter-robot communication [26], optimization of distance to be traveled [27,28], or recharging optimization [29,30]. In this respect, we present DCP-SLAM, a LiDAR-based distributed collaborative partial SLAM framework for swarm robotics. In the proposed DCP-SLAM technique, we focus on energy efficiency of the swarm as a whole rather than trying to accurately map individual obstacles. We argue that the requirement of maintaining a safe distance from obstacles allows us to conduct trajectory planning even with partially discovered obstacles. Moreover, a cluster of obstacles where interobstacle distance does not allow a robot to pass between them can be fuzzified into one large obstacle without causing excessive elongation of the collision-free path. Furthermore, this fuzzification reduces communication cost as less data need to be transmitted. Thus, in order to have a

low-bandwidth low-computation navigation setup for the swarm, an exact replication of the map is not required to be communicated between the robots. Furthermore, the safe distance requirement, also referred to as inflation, since each obstacle's dimensions are inflated by this amount, covers up the sacrificed error while performing active obstacle detection and avoidance maneuvers. This is handled in three phases: (i) The first is the detection and map building phase, in which the point cloud returned by LiDAR after each scan is utilized to update the obstacle list. Here, we adapt the Euclidean-distance-based incremental region growing technique of [31] by adding fuzzification while calculating distances. The concept of fuzzification and merging of objects is explained in Figure 1. The resulting obstacle list is broadcast to the rest of the swarm. (ii) In the second phase, the robot generates waypoints for bypassing all known obstacles in the updated obstacle list. (iii) Finally, the robot will align itself in order to move towards the nearest waypoint as a temporary goal. Since the obstacle list only contains information about obstacles that have been fully or partially discovered so far, hitherto unknown obstacles may still be present and in the way of a robot. Therefore, the robots are required to continually perform scanning and obstacle detection during navigation. Simulations of various scenarios show that the DCP-SLAM technique results in swarms where later robots show marked improvement in their trajectory by utilizing partial maps discovered by their earlier fellows. Even in complex scenarios, the optimal path is established relatively quickly.

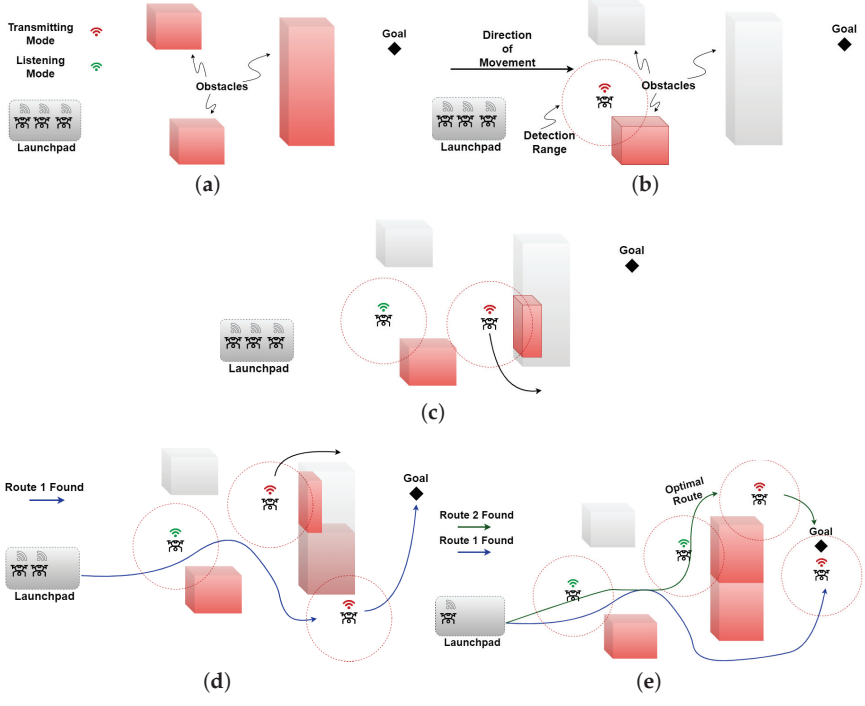


Figure 1. Illustration of DCP-SLAM technique. Unknown obstacles are illustrated in grey. Once an obstacle (or a part of it) is detected by the onboard sensors of the robot(s), it is illustrated in red. (**a**) Initial setup, launchpad where robots are launched from and unknown obstacles are shown. (**b**) Robot 1 detects an obstacle while navigating towards goal. Robot 1 broadcasts the information to following robots. (**c**) Partial detection of second obstacle by Robot 1, Robot 2 navigating towards the waypoint dropped by Robot 1. (**d**) Robot 1 finds unobstructed path to goal, Robot 2 navigates from the other side of the obstacle, while broadcasting the information. (**e**) Robot 2 finds unobstructed path to goal. Followers opt for the optimal path.

The organization of the rest of the paper is as follows: Section 2 provides the related work and motivation for DCP-SLAM. Section 3 explains working of the proposed algorithm in detail. Simulation results are provided in Section 4. Finally, concluding remarks, discussion, and future work is presented in Section 5.

2. Related Work

Navigation in environments with no prior map information raises several research challenges. The autonomous robots or agents, while navigating in unknown environments and not having to bank on acquiring information from central or remote servers, must utilize their onboard sensors to observe, analyze, and perform necessary actions based on the information at hand for collision-free navigation and successful completion of the mission [25]. At the same time, reducing the power consumed by the autonomous agents, i.e., effectively exploiting the available resources, in order to increase the mission duration is of paramount importance [32]. With reference to the existing literature, the proposed approach relates closely to the detection, avoidance, and energy-efficient path development for the swarm as a whole.

In SLAM, exploration is fundamental, and its importance becomes even more vital in swarm SLAM systems (or multi-robot SLAM). Various exploration methodologies are utilized to facilitate the exploration, such as random walk exploration [33,34], potential field-based or frontier-based exploration; however, path planning is the most commonly utilized scheme [35]. Authors in [34] utilized the random walk exploration scheme for a robot swarm mapping event, and illustrated that an occupancy grid can be produced in a closed indoor environment. However, in practical situations, the proposed methodology struggles to work efficiently due to the poor quality of the close-range sensors. Furthermore, by employing high-precision sensors, this issue can be rectified to virtually produce any kind of map, as shown in the experiment performed by the authors in [36]. Ideally, the robots in the swarm should be as simple as possible, and should employ simple algorithms with low overhead, due to practicality. Therefore, swarm SLAM schemes that are able to make use of low-cost and low-precision sensors and generating relatively abstract maps for navigational purposes are of greater importance. Authors in [37] demonstrated such an approach for generating semantic maps; however, the computational complexity of the proposed approach is significantly high for real-world scenarios for swarms. Such an issue, with focus on map retrieval without centralization, is an open problem in the field of swarm SLAM. As shown in [34], a natural technique for merging the maps to be utilized by individual robots of the swarm is to accumulate the individual maps on a single central system. Furthermore, another approach proposed by [11] works by merging individually generated maps in all robots in the swarm. In this manner, every robot or agent in the swarm has access to the map; however, for the successful deployment of such an approach, an external framework or central node is necessary.

Literature differentiates collaborative SLAM (C-SLAM) as centralized or decentralized architectures, with centralized schemes gathering all the data into a central server (central station or a robot) and assessing the trajectories for all the robots. Different sensing arrangements, such as laser or LiDAR-based [18,38] and vision-based [40**?**], are examined in the centralized C-SLAM. In [38], locally generated maps constructed via utilizing LiDARs are sliced, and all the segments of the sliced data are accumulated for the detection of loop closure. The authors of [18] introduced a large-scale autonomous mapping and positioning system (LAMP) utilizing a 3D LiDAR scanner mounted on single or multiple robots to scan the surroundings, and an RGB-D camera for the detection and localization of known obstacles in the environment. In the proposed scheme, all the robots in the swarm are connected to a central server or node, and in case of loss of communication with the base station, the algorithm switches back to classical single-robot SLAM-based navigation. For vision-based techniques, e.g., the works presented in [40**?**], the odometry key frames obtained by utilizing onboard visual sensing mechanisms are offloaded to a

central server. Moreover, to obtain an optimized solution by discovering overlaps, the bag-of-words approach is utilized.

On the other hand, for swarms to function effectively, especially in conditions where communication range is limited or there are poor communication channels, distributed SLAM is a more appropriate approach. The authors of [12] presented a decentralized visual SLAM approach, utilizing readily accessible datasets, and analyzed how to reduce the transmission data. Further investigation is required in order to achieve significant reduction in data transmission to make the whole system energy-efficient. Another recent approach [41] proposed simplification of data representation and utilizing maximum clique outlier rejection for distributed place recognition and pose graph optimization. The algorithm is able to build 3D metric semantic meshes accurately, and is able to handle loop closure errors that may occur. Ref. [42] utilized the Gauss–Siedel technique, and presented a two-stage distributed approach, where they used object-based models to reduce the communication costs between the robots. Furthering the work in [42], authors in [43] presented DOOR-SLAM, a distributed SLAM technique based on peer-to-peer communication. The presented methodology utilizes a pose graph optimizer to reject false inter-robot loop closures. In [22], the authors employed a lightweight Scan Context descriptor to facilitate swarm SLAM, and presented a two-stage global and locally distributed optimization framework. The presented approach runs on a dataset collected by one robot.

Collaborative Sensing for Map Building

The approach presented in [44] results in significant improvements in swarm SLAM field while utilizing partial feedback from the leading agents in order to find an efficient path to the destination. However, there is no methodology to detect whether the path to the destination discovered by the agent is the most efficient one. This is due to the fact that in the aforementioned approach, as soon as an agent successfully finds an unobstructed path to the destination goal, it broadcasts the tracker points for the rest of the agents in the swarm to follow to be able to reach the destination. Therefore, the presented methodology of partial swarm SLAM is further enhanced in this work by the constant sharing of findings between agents in the swarm in order to build a map in a collaborative manner.

We now present a collaborative sensing approach whereby individual agents form partial maps of known obstacles discovered by other agents, and utilize this information to find the shortest path to their respective goals. It is to be noted that all agents continue sensing their environment to avoid possible collisions with hitherto unknown obstacles or hidden parts of partially discovered obstacles, continuously passing on any such observation to all other agents to update their world view. Figure 1a shows the general map. Grey objects are unknown obstacles, whereas red objects or red parts of the objects imply detected/partially detected obstacles, respectively. As illustrated in Figure 1b, Robot 1 detects an obstacle while navigating towards its goal. Figure 1c shows a scenario in which Robot 1 partially detected a large obstacle and decides to bypass it, while Robot 2 is following the first waypoint dropped by Robot 1. Upon reaching waypoint 2 or the partially detected obstacle Figure 1d, Robot 2 navigates from the other side of the obstacle to complete the detection. This approach also facilitates finding an alternate route that may be superior to the previously discovered path. Upon detection of the optimal path, the rest of the robots will utilize the available information to locally evaluate the options and select the route that is optimal for themselves, as illustrated in Figure 1e.

3. Proposed Approach

Algorithm 1 provides the general pseudocode for navigation and obstacle detection. All agents execute this top-level algorithm locally by utilizing their onboard processing units. In the beginning of the mission, all the agents are assigned IDs and connection between them is set up in a leader–follower manner. A global leader is declared, and in a hierarchical manner, the respective leaders are connected to the immediate respective followers, in the initialization phase of the algorithm. Afterwards, the respective agent

starts navigation if it has not reached the designated goal, i.e., $\mid \Gamma_S^i - \Gamma_G \mid > \rho$, where Γ_S^i and Γ_G are the coordinates of the i th agent and the goal, respectively, and ρ is the radius defined around the goal point to handle any errors that may occur while acquiring the coordinates. If the agent has not yet reached the desired goal, and any obstacle(s) is(are) detected by the onboard ranging sensors, the attributes of the detected obstacle are then stored in an obstacle list ($\S_{[i][j:0\rightarrow n]}$), where i stands for the i th agent and $j : 0 \rightarrow n$ stands for the j th obstacle in the list. The respective agent then checks whether this particular object has already been detected by any of the agents that may have navigated through the same path by calling the Merge Objects() function. This Merge Objects() function will perform the following actions if the detected obstacle is present in the obstacle list or if the detected obstacle is not present in the obstacle list. If the latter is true, then based on the attributes of the detected obstacles in the obstacle list and the newly detected obstacle, it is determined whether the newly detected obstacle is a completely different obstacle or part of a previously partially detected obstacle. These attributes are then passed onto the Shortest Path Obstacle Avoidance() function in order for the agent to be able to perform collision avoidance actively.

Algorithm 1 Navigation

Input: Self.ID: S_{id}; Leader is Alive: L_{alive}; Self.Coordinates: Γ_S^i;
Output: Obstacle list: $\S_{[i][j:0\rightarrow n]}$;
Constant: Goal.Coordinates: Γ_G; Goal.Radius: ρ;

```
S_id ← Initialization, ID allocation;
if S_id == 1 then
    Leader ← Self;
    L_alive ← False;
else
    Leader ← Self;                                  ▷ Connection to respective leaders
    L_alive ← True;
end if
while | Γ_S^i − Γ_G | > ρ do
    if §_[i][j:0→n] ← Obstacle Detection() then
        Merge Objects and Update Map(§_[i][j:0→n]);
        Shortest Path Obstacle Avoidance(§_[i][j:0→n], Γ_S^i, Γ_G);
    end if
    Continue Navigation;
end while
```

3.1. Collision Avoidance and Map Building

When an agent is launched, it executes the obstacle avoidance algorithm to find the shortest path to its destination while avoiding known obstacles. It selects the nearest inflection point as its temporary goal or waypoint, and moves towards this waypoint while continuously sensing its environment for hitherto unseen obstacles. In the case that an obstacle is sensed on its way to the waypoint, the agent performs collision avoidance while simultaneously broadcasting the coordinates of the newly discovered parts of the obstacle. All agents update their individual maps by adding the newly gathered information and run the Merge Objects algorithm to merge multiple sections of the same obstacle into one obstacle. After the agent has moved clear of the obstacle, the process of selecting the waypoint towards its goal is repeated until it completes its mission.

3.2. Merge Objects and Update Map

Algorithm 2 starts by checking whether the detected obstacle exists in the obstacle list ($\S_{[i]}$). Then, for each existing obstacle in the list, the current detected obstacle's distance is

compared with a defined margin of error (λ_e). If the distance or gap between the existing obstacle in the list and the current detected obstacle is equal to or less than λ_e, both obstacles are merged together and considered to be a single obstacle. Otherwise, the current detected obstacle is added to the obstacle list. The updated list is then broadcast by the agent to rest of the agents.

Algorithm 2 Merge Objects and Update Map

Input: Obstacle list: $\S_{[i][j:0\rightarrow n]}$; Self.ID: S_{id}; Leader is Alive: L_{alive}; Self.Coordinates: Γ^i_S;

Constant: Temporary obstacle variable: Temp_{obj}; Obstacle dimensions: $\mathbb{D}^{[i][j]}_{o+\alpha}$;Margin of error: λ_e

```
for i in §[i] do
    Temp_obj ← §[i];
    for j in §[i][j] do
        if | Temp_obj − − §[i][j] | ≤ λ_e then
            Merge → Temp_obj ∪ §[i][j];
        else
            Add Temp_obj to §[i];
        end if
        Broadcast();
    end for
end for
```

3.3. Shortest Path with Obstacle Avoidance

Algorithm 3 shows the pseudocode of the shortest path calculation, as illustrated in Figure 2, while utilizing the collision avoidance algorithm. The algorithm starts by analyzing the attributes of the detected obstacle(s) from the obstacle list, i.e, $\S_{[j:0\rightarrow n]}$ (Line 1), where D^j_o is the j th obstacle's distance from the agent, $\angle^j_o$ is the angle at which the obstacle lies, and $\mathbb{D}^j_{o+\alpha}$ represents the obstacle's dimensions (o: $0 \rightarrow \alpha$). The agent in question then updates the Euclidean distance ($\mathbb{E}$) to the goal from its current position, i.e., Γ_G and Γ^i_S, respectively (Line 2). Afterwards, for each detected obstacle, it is checked whether the obstacle poses a potential collision risk, i.e., lies within the planned trajectory of the agent. This is achieved by analyzing whether the obstacle's detected dimensions intersect $\mathbb{E}$ at any point (Lines 3–5). It is important to note here that the agent may or may not be able to detect the complete obstacle, as the obstacle may be larger than the detection range of the onboard sensor system of the agent. In either case, the extreme edges of the obstacle that are visible to the agent are recorded. If the straight line passes through any of the detected obstacle(s), then based on the available information, the agent calculates new waypoints for collision-free navigation (Line 6). Finally, the waypoint closest to the calculated shortest path is selected as a temporary goal for the agent to navigate towards (Line 12).

Here, we present two methods of path selection with obstacle avoidance, namely, shortest path planning with obstacle avoidance (SP-OA) and immediate waypoint selection with obstacle avoidance (IWp-OA). First, we explain shortest path planning with the obstacle avoidance algorithm with reference to Figure 3. Consider that a robot at position R is planning its path towards its goal G. Furthermore, its map of the world is populated by three obstacles, A, B, and C, that have been discovered by earlier robots. The steps taken by the said robot, as described in the algorithm Y, are as follows:

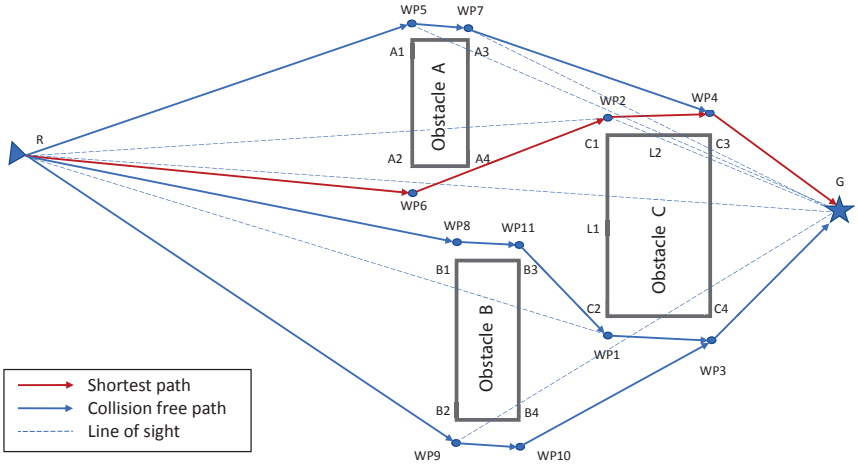


Figure 2. Shortest path planning with obstacle avoidance.

Algorithm 3 Shortest path with Obstacle Avoidance

Input: Obstacle list: $\S_{[i][j:0\rightarrow n]}$; Self.Coordinates: Γ_S^i; Goal.Coordinates: Γ_G;

Output: Obstacle's distance: $D_o^{[i][j]}$; Obstacle's angle: $\angle_o^{[i][j]}$; Obstacle dimensions: $\mathbb{D}_{o+\alpha}^{[i][j]}$; Temporary Goal: Γ_{temp};

Constant: Safe Distance: s_d;

$D_o^{[i][j]}, \angle_o^{[i][j]}, \mathbb{D}_{o+\alpha}^{[i][j]} \leftarrow \S_{[i][j:0\rightarrow n]}$;
Order § based on increasing $D_o^{[i][j]}$;
$\overline{\mathbb{E}} \leftarrow \overline{\Gamma_G \Gamma_S^i}$; ▷ line segment between Self and Goal
for § in $\S_{[i][j:0\rightarrow n]}$ **do**
  for each edge $\overline{\mathbb{D}_o \mathbb{D}_{o+\alpha}}$ of § **do** ▷ one edge to other of the obstacle
    if edge $\overline{\mathbb{D}_o \mathbb{D}_{o+\alpha}}$ Intersects $\overline{\mathbb{E}}$ at Point $\mathbb{I}$ **then**
      $\mathrm{WP}(i) = (\mathbb{D}_o - s_d)$, $\mathrm{WP}(i+1) = (\mathbb{D}_{o+\alpha} + s_d)$;
    end if
  end for ▷ End for all edges of the object §
  $\Gamma_{temp} = \mathrm{WP}(i)$ if $(|\mathrm{WP}(i) - \mathrm{I1}| < |\mathrm{WP}(i+1) - \mathrm{I1}|$ in $\mathrm{WP}_i)$ else $\mathrm{WP}(i+1)$;
end for ▷ End of all objects in the object list $\S_{[i][j:0\rightarrow n]}$
Move to Γ_{temp};

1. The robot draws a straight line from its current position R to its goal G to determine whether it intercepts any edge of the known obstacles. In the instant case, the line $\overline{RG}$ is intercepted by the edge $\overline{C_1C_2}$ of the obstacle C at point I1.
2. Next, it chooses two waypoints that are a minimum safe distance away from vertices C1 and C2; let us call these W1 and W2. Possible paths are $\overline{RW_1}$ followed by $\overline{W_1G}$;, and $\overline{RW}$ followed by $\overline{W_2G}$.
3. The robot iterates steps 1 and 2 above for all possible path segments, always progressing from left to right, creating a new waypoint for each edge that intercepts any path, and repeating steps 1 and 2.
4. We now have a directed acyclic graph (DAG) with multiple obstacle-free paths from the robots current position to the goal.

5. The robot now selects the first waypoint on shortest obstacle-free path as its temporary goal and starts moving towards it.

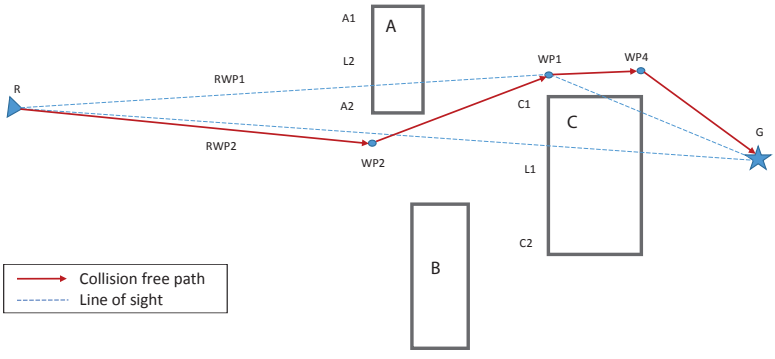


Figure 3. Immediate waypoint selection.

This algorithm will be re-run every time a new obstacle is discovered, a hidden part of a known obstacle is discovered by the robot's onboard sensors, or such a finding is communicated by other robots in the swarm. As an alternate approach in the interest of saving computation time and energy, we propose a simplified version of the above algorithm. In this approach, the robot works backwards from the goal, choosing the shorter of two paths at each intersection upon reaching an obstacle. The simplified approach is explained below with reference to Figure 3.

1. The robot draws a straight line from its current position R to its goal G and determines whether it intercepts any edge of the known obstacles. In the instant case, the line $\overline{RG}$ is intercepted by the edge $\overline{C_1C_2}$ of the obstacle C at point I1.
2. Since the top left corner C1 is nearer to I1 than the bottom left corner C2, it decides to circumvent the obstacle from the top left corner C1. It chooses a waypoint that is a minimum safe distance away from C1, let us call it W1, and sets this waypoint as its temporary goal.
3. It repeats steps 1 and 2 above; this time, the line $\overline{RW_1}$ is intercepted by the edge $\overline{A_1A_2}$ of obstacle A at point I2. The robot avoids obstacle A from bottom left corner A2 and chooses a waypoint WP2 that is a minimum safe distance away from vertex A2. The robot now sets its temporary goal to WP2.
4. The robot repeats step 1 by drawing a straight line $\overline{RW_2}$. Since this line is not intercepted by any known obstacle, the algorithm finishes, and the robot starts moving towards its temporary goal WP2.

The concept of fuzzification of objects is exemplified in Figure 4. In this example, three points, A, B and C, are returned by the LiDAR sensor. The level of inflation or minimum safe distance s_d is shown by the red line. The three points are fuzzified in all directions by the level of inflation, shown as red lines in the four dimensions around each point. Next, rectangular objects are created that encompass all edges of the fuzzified points. The objects that overlap, or are adjacent, are merged into one object. Thus, in Figure 4, points A and B are assumed to belong to one object, while point C probably belongs to a second object.

Since there may be other obstacles in the way that are yet to be discovered, planning a complete path to the final goal may be an unnecessary effort at this stage. Additionally, for this reason, the robots continually scan their immediate environment for any hitherto unseen obstacles. They also continue listening to broadcasts from other robots for information about a discovery of a new obstacle or the discovery of further sections of a known obstacle. In either case, the path planning algorithm is executed again to ensure that the path followed by the robot is obstacle-free.

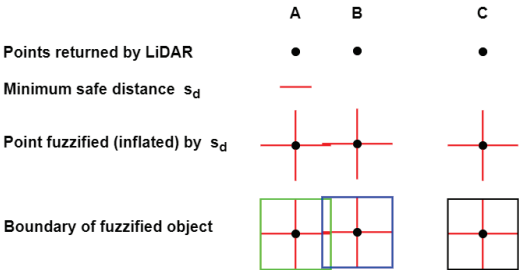


Figure 4. Concept of fuzzification.

4. Simulation Results

A two-dimensional XY plane, i.e., all the objects (robots and obstacles) are at the same altitude, of 12 km × 6 km is used, with unknown obstacles randomly scattered in space. The number of agents are set to ten and are launched from the same coordinates one after the other. Robots move with a maximum speed of 72 km per hour, or 20 m per second. Python graphics are utilized for simulation purposes. For simulation and testing, we utilized the mathematical models of differential drive robots, and further equipped them with the output of a simulated LiDAR sensor in a two-dimensional plane. The following is a kinematic model of a differential drive robot:

$$\begin{aligned} \dot{x} &= v cos\theta, \\ \dot{y} &= v sin\theta, \\ \dot{\theta} &= \frac{v_\Delta}{W} \end{aligned} \tag{1}$$

where $\dot{x}$ and $\dot{y}$ are the x and y positions of the robot, v is the velocity, and $\dot{\theta}$ is the heading angle of the robot.

The following equation is utilized to calculate the turning curve of the robot:

$$\Delta V = \frac{v_r - v_l}{W} \tag{2}$$

where ΔV is the difference between the left and the right wheel speed, v_r and v_l are the right and the left speeds, respectively, and W is the width of the robot.

In order to show the progress of the swarm graphically, the field was scaled to fit the available screen resolution, each pixel representing 10 m. The following are the initial conditions and assumptions defined for our work:

1. The robots pass through a vast passage area between the launch zone and delivery zone.
2. There are randomly placed multiple obstacles in the passage area.
3. The communication channel between the robots is considered to be ideal and lossless.
4. Utilizing onboard localization methods, the robots obtain their position vector.
5. The range of LiDAR sensors is 100 m.

Figure 5 shows the effectiveness of utilizing the fuzzy nature of the proposed technique. As can be seen in Figure 5a, Robot #1, on the left of the figure moves along the straight line towards the goal, shown by red circle in the bottom right, and discovers obstacle #1 using a LiDAR sensor; the point cloud is indicated by red dots on the left edge of the obstacle. Using the immediate waypoint with Obstacle Avoidance (IWp-OA) algorithm, it chooses to circumvent the obstacle from below since the the length of edge discovered so far is smaller on this side. As shown in Figure 5b, Robot #2 finds that an already discovered part of obstacle #1 is in the way of the goal, and it decides to circumvent obstacle #1 from above since, by then, most of the left edge is discovered and the top corner seems the shorter way. In another instance, presented in Figure 5c, Robot #1 has discovered, and communicated

to other robots, obstacle #4 and the bottom part of obstacle #5. Robot #2 has discovered obstacle #3 and avoids this obstacle by going below it. It has also discovered the top part of obstacle #5. Utilizing the already available information, Robot #3 initially aims to go over the top of obstacle #3, and later discovers obstacle #2; however, since the gap between obstacle #2 and obstacle #3 is narrower than the minimum safe distance, the two are merged and recorded as one obstacle, shown by the blue grid encompassing both obstacles in Figure 5d. As robot #3 is already near the top edge of the merged obstacle, as shown in the simulation screenshot in Figure 5e, it decides to circumvent it from above. Robot #4, after clearing obstacle #1 from above, finds the merged obstacle in its way and decides to bypass it from below. Similarly, Figure 5f shows that Robot #5, or any other following robots, benefits from the information gathered by earlier robots and chooses to pass over the top right corner of obstacle #5, resulting in the shortest path.

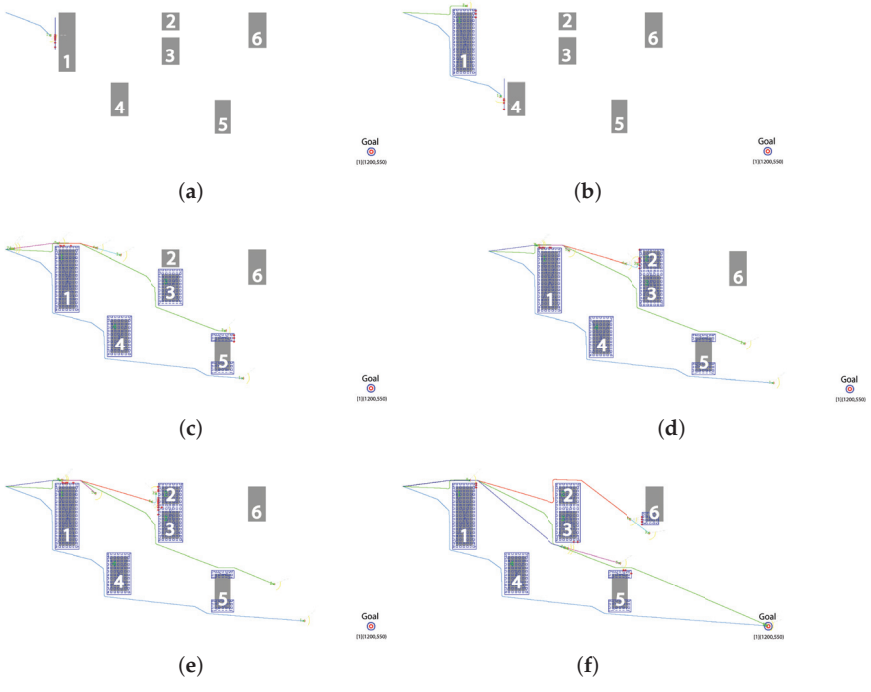


Figure 5. Simulation snapshots: showing the operation of the IWp-OA algorithm. (**a**) Robot 1 avoiding the first detected obstacle from the bottom edge. (**b**) Robot 2 circumvents the obstacle from above to discover an alternate route. (**c**) Robots 1 and 2 have partially discovered obstacle 5. (**d**) Robot 3 discovers obstacle 2, and utilizing fuzzification, combines it with the already discovered obstacle 3. (**e**) Robot 4 chooses the trajectory, by utilizing the broadcast information, to bypass obstacle 3 from below. (**f**) Robot 5 utilizes information provided by all other robots and chooses the optimal trajectory.

Similarly, Figure 6 shows the trends of the routes taken by robots from different experimental setups with different launch and goal coordinates. Figure 6a,b show the overall traces of the robots in the swarm from launch to goal, where the launch and goal are slightly towards the center of the initial obstacles. Figure 6c shows the detection and resultant trend when the launch and goal are both moved towards one side of the obstacles. All the robots take similar routes. The robots 1 and 2 performed the optimization for finding an efficient path; afterwards, the rest of the robots followed the discovered optimal path, as shown. Figure 6d shows the trend when the goal is moved diagonally to the other side with same placements of the obstacles, and Figure 6e shows the optimization trend when

the obstacles are randomly relocated. Figure 6f,g show the optimization trend over time by the robots when the goal is moved from one point to another while keeping the launch coordinates the same. Similarly, Figure 6h–j show the overall trend while covering all possible scenarios for testing the efficiency of the proposed DCP-SLAM technique.

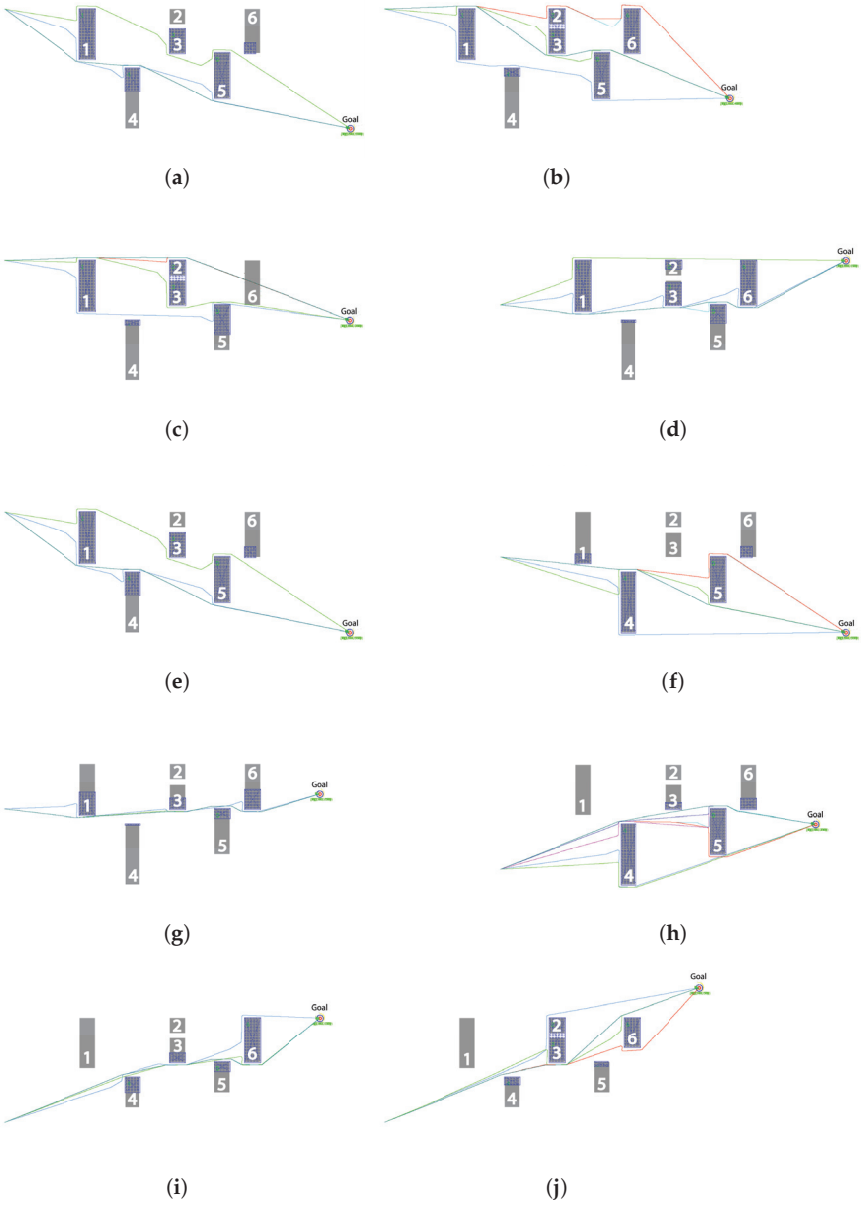


Figure 6. Simulation results: different experimental scenarios. (**a**) Experimental scenario 1. (**b**) Experimental scenario 2. (**c**) Experimental scenario 3. (**d**) Experimental scenario 4. (**e**) Experimental scenario 5. (**f**) Experimental scenario 6. (**g**) Experimental scenario 7. (**h**) Experimental scenario 8. (**i**) Experimental scenario 9. (**j**) Experimental scenario 10.

Figure 7 shows the distances traveled by individual robots from mission start until they reach the goal of the experiments performed in Figure 6. As reflected, the proposed algorithm manages to find the optimal path to the goal relatively effectively and aggressively.

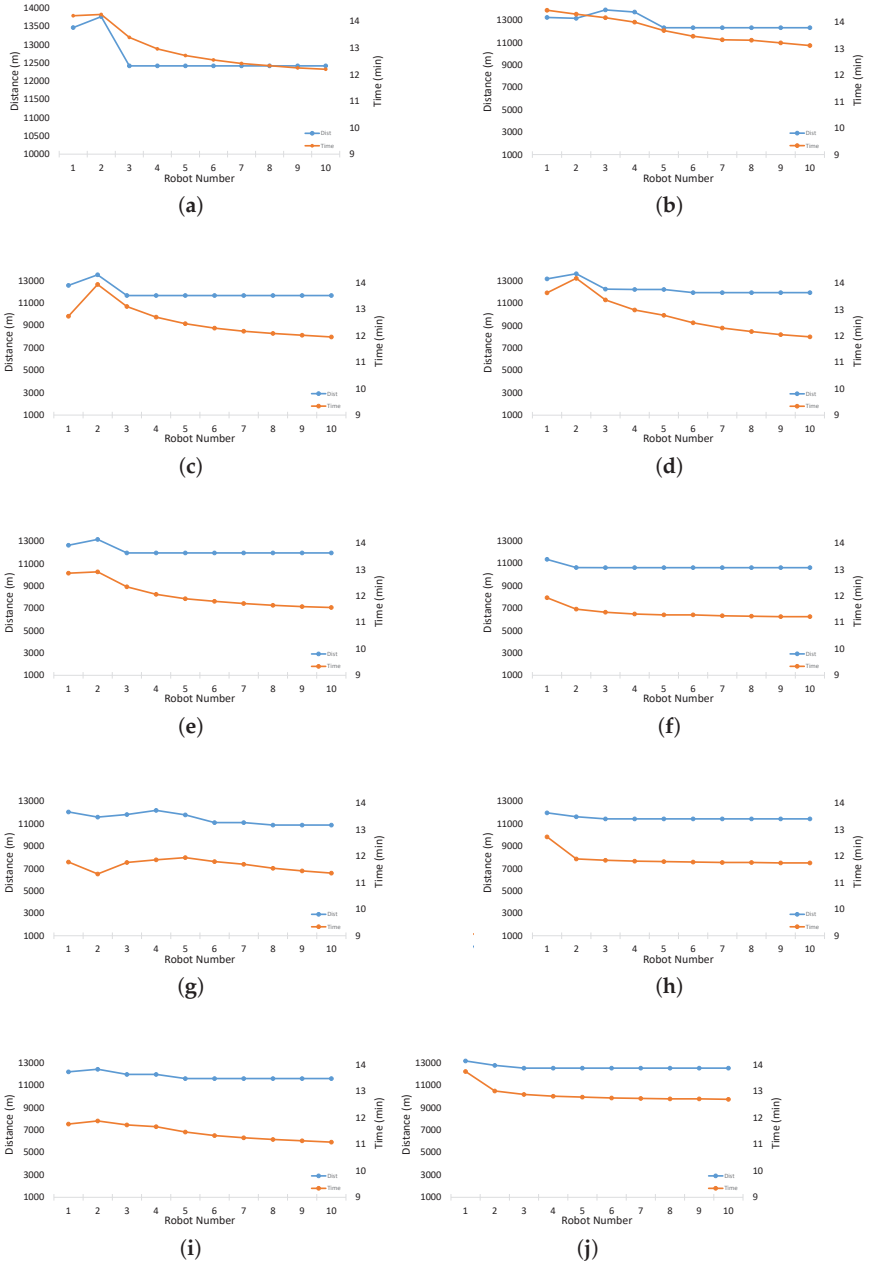


Figure 7. Simulation results: results for respective experiments, showing the total distance traveled by each robot along with the time it took to reach the goal. (**a**) Experimental scenario 1. (**b**) Experimental scenario 2. (**c**) Experimental scenario 3. (**d**) Experimental scenario 4. (**e**) Experimental scenario 5. (**f**) Experimental scenario 6. (**g**) Experimental scenario 7. (**h**) Experimental scenario 8. (**i**) Experimental scenario 9. (**j**) Experimental scenario 10.

Figure 8 shows the maximum and minimum distance traveled by a single robot in each experimental scenario, along with the average distance the whole swarm traveled. As is evident from the results, utilizing the proposed approach optimizes the traveling distance relatively quickly as compared to the previously proposed PS-SLAM technique [44] or non-collaborative methods. The results shown in Figure 9 clearly indicate the efficiency in terms of distance traveled by individual robots and, consequently, the swarm as a whole, of the proposed DCP-SLAM technique as compared to the PS-SLAM technique. Evidently, significant consumption is reduced while utilizing the proposed approach, as the distance the swarm traveled is reduced on average by around 10 km. This amounts to an efficiency increase by approximately 10% over the previously developed algorithm [44].

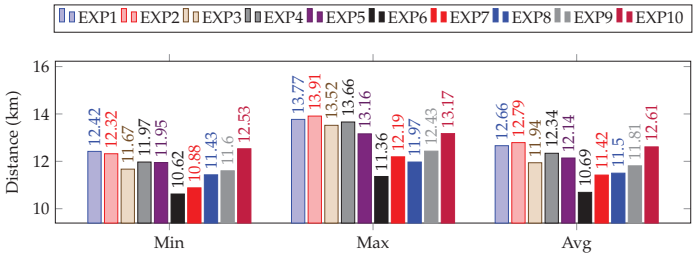


Figure 8. Distance (minimum, maximum, and on average) traveled by the swarm in all experimental scenarios.

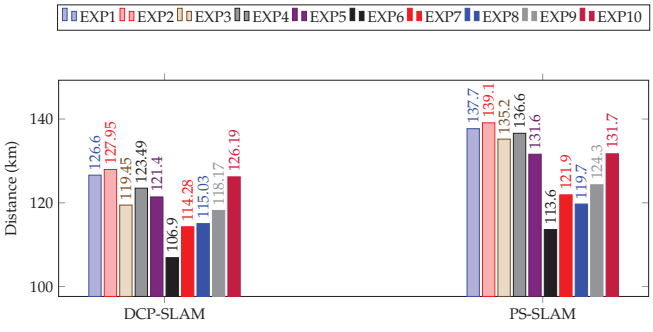


Figure 9. Comparative results of distance traveled by the swarm as a whole

Tables 1–3 show total communication cost of each robot in number of messages, Table 1, in number of bytes transmitted, Table 2, and energy consumed in transmission, Table 3. We compare two scenarios for all experiments, firstly, where a robot transmits the point cloud every time it detects an obstacle, resulting in high communication cost for each robot and secondly, where a robot transmits its list of vertices of all obstacles detected so far only when it discovers a new object or hitherto unseen part of an object. As can be seen from the three tables, the latter scheme results in significant saving in communication cost. Since obstacles are gradually discovered and communicated by leading robots, the robots towards the tail end of the swarm have almost zero communication cost.

Table 1. Number of message transmissions.

Exp	1		2		3		4		5		6		7		8		9		10	
Robot	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj
1	29,340	105	32,353	154	29,907	245	29,187	185	22,097	101	30,606	167	20,900	140	26,582	160	16,420	131	27,225	140
2	27,393	89	30,766	105	35,216	80	35,197	107	20,523	35	21,868	42	15,338	0	19,078	11	17,867	32	18,770	10
3	12,182	0	23,914	38	18,781	0	15,495	0	11,987	0	22,112	1	26,824	82	18,704	1	14,270	56	18,717	1
4	12,182	0	22,402	0	18,781	0	15,854	5	11,987	0	22,112	0	22,022	10	18,700	0	14,271	0	18,717	0
5	12,182	0	19,030	0	18,781	0	15,854	0	11,987	0	22,112	0	23,951	1	18,700	0	10,052	0	18,717	0
6	12,182	0	19,030	0	18,781	0	11,092	0	11,987	0	22,112	0	16,933	56	18,700	0	10,052	0	18,717	0
7	12,182	0	19,030	0	18,781	0	11,092	0	11,987	0	22,112	0	16,933	0	18,700	0	10,052	0	18,717	0
8	12,182	0	19,030	0	18,781	0	11,092	0	11,987	0	22,112	0	15,726	2	18,700	0	10,052	0	18,717	0
9	12,182	0	19,030	0	18,781	0	11,092	0	11,987	0	22,112	0	15,726	0	18,700	0	10,052	0	18,717	0
10	12,182	0	19,030	0	18,781	0	11,092	0	11,987	0	22,112	0	15,726	0	18,700	0	10,052	0	18,717	0

Table 2. Number of kB (kilobytes) transmitted.

Exp	1		2		3		4		5		6		7		8		9		10	
Robot	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj	Raw	Obj
1	82.15	5.46	90.59	8.01	83.74	12.74	81.72	9.62	61.87	5.25	85.69	8.68	58.52	7.28	74.43	8.32	45.98	6.81	76.23	7.28
2	76.70	4.63	86.15	5.46	98.61	4.16	98.55	5.56	57.46	1.82	61.23	2.18	42.95	0	53.42	0.57	50.03	1.66	52.56	0.52
3	34.11	0	66.96	1.98	52.59	0	43.39	0	33.56	0	61.91	0.05	75.11	4.26	52.37	0.05	39.96	2.91	52.41	0.05
4	34.11	0	62.73	0	52.59	0	44.39	0.26	33.56	0	61.91	0	61.66	0.52	52.36	0	39.96	0	52.41	0
5	34.11	0	53.28	0	52.59	0	44.39	0	33.56	0	61.91	0	67.06	0.05	52.36	0	28.15	0	52.41	0
6	34.11	0	53.28	0	52.59	0	31.06	0	33.56	0	61.91	0	47.41	2.91	52.36	0	28.15	0	52.41	0

Table 2. Cont.

Exp	1		2		3		4		5		6		7		8		9		10	
Robot	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**
7	34.11	0	53.28	0	52.59	0	31.06	0	33.56	0	61.91	0	47.41	0	52.36	0	28.15	0	52.41	0
8	34.11	0	53.28	0	52.59	0	31.06	0	33.56	0	61.91	0	44.03	0.10	52.36	0	28.15	0	52.41	0
9	34.11	0	53.28	0	52.59	0	31.06	0	33.56	0	61.91	0	44.03	0	52.36	0	28.15	0	52.41	0
10	34.11	0	53.28	0	52.59	0	31.06	0	33.56	0	61.91	0	44.03	0	52.36	0	28.15	0	52.41	0

Table 3. Energy consumed in transmission (mJ).

Exp	1		2		3		4		5		6		7		8		9		10	
Robot	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**	**Raw**	**Obj**
1	82.15	0.55	90.59	0.80	83.74	1.27	81.72	0.96	61.87	0.53	85.69	0.87	58.52	0.73	74.43	0.83	45.98	0.68	76.23	0.73
2	76.70	0.46	86.14	0.55	98.61	0.42	98.55	0.56	57.46	0.18	61.23	0.22	42.95	0	53.42	0.06	50.03	0.17	52.56	0.05
3	34.11	0	66.96	0.19	52.59	0	43.39	0	33.56	0	61.91	0.01	75.11	0.43	52.37	0.01	39.96	0.29	52.41	0.01
4	34.11	0	62.73	0	52.59	0	44.39	0.03	33.56	0	61.91	0	61.66	0.05	52.36	0	39.96	0	52.41	0
5	34.11	0	53.28	0	52.59	0	44.39	0	33.56	0	61.91	0	67.06	0.01	52.36	0	28.15	0	52.41	0
6	34.11	0	53.28	0	52.59	0	31.06	0	33.56	0	61.91	0	47.41	0.29	52.36	0	28.15	0	52.41	0
7	34.11	0	53.28	0	52.59	0	31.06	0	33.56	0	61.91	0	47.41	0	52.36	0	28.15	0	52.41	0
8	34.11	0	53.28	0	52.59	0	31.06	0	33.56	0	61.91	0	44.03	0.01	52.36	0	28.15	0	52.41	0
9	34.11	0	53.28	0	52.59	0	31.06	0	33.56	0	61.91	0	44.03	0	52.36	0	28.15	0	52.41	0
10	34.11	0	53.28	0	52.59	0	31.06	0	33.56	0	61.91	0	44.03	0	52.36	0	28.15	0	52.41	0

5. Conclusions and Future Work

In this article, we present a methodology for finding an optimal solution for the navigation of a swarm of autonomous robots in environments with no prior map information, utilizing only local onboard sensors for observational purposes and inter-robot communication for the sharing of observations in a concise manner. In the presented technique, robots utilize their onboard ranging sensors to detect and avoid close-range obstacles while navigating towards their goal. The approach works by utilizing the information regarding maps built by leading robots for optimizing the routes for rest of the robots in the swarm. Interestingly, this partial map building approach is robust against communication failure of one or more robots, since the followers build upon whatever information is at hand and utilize onboard sensors to augment it. Thus, any gaps owing to lost information are quickly filled by followers. We have simulated various situations with multiple objects obstructing the straight path to the destination, and show that the proposed approach results in the swarm learning its environment in sufficient detail with the passage of few leading robots so that the remaining members of the swarm can find an optimal path.

Thus, considering the arrival of the whole swarm at the destination as the mission to be carried out with optimal path traversal, some of the leading robots end up taking longer routes and discovering hidden parts of partially discovered obstacles. This additional information is shared with the rest of the swarm, and results in a more informed choice of route by following robots. This results in optimal path selection for the followers and up to 13% saving in the travel path for individual robots. Furthermore, utilizing the proposed approach results in efficiency of up to 11% in traveling distance for the swarm as a whole.

Another contribution of this work is that communication cost is limited to transmitting only significant new discoveries rather than sending the whole point cloud with each detection event of LiDAR by each member of the swarm. This is a direct result of the fuzzification and merging of detected obstacles that reduces the number of obstacles. As long as an obstacle is within the sensing range of a robot's LiDAR sensor, it receives several detection points in the point clouds for each scan. If this detection data were to be communicated to the rest of the swarm, it would result in high communication cost. Instead, the robot performs several preprocessing steps before transmission. Firstly, it evaluates whether a detection point belongs to an already detected object or not. Here, our fuzzification of the detected objects helps in returning affirmative answers for very close points. If the answer of the first step is negative, it is assumed that the said point belongs to an undetected object. The aforementioned point is fuzzified to form an object with dimensions equal to the minimum clearance distance, and the resulting object is added to the obstacle list. Next, the whole obstacle list is scanned to determine whether any two objects are adjacent or overlapping, merging such objects into one larger object. Since the obstacle list is maintained in a sorted order, only one scan is sufficient to merge all adjacent or overlapping objects. After this final step of merging, the updated obstacle list is broadcast to the rest of swarm. This scheme results in the saving of communication cost and economy of associated communication energy.

In our future work, we aim to further develop the proposed methodology by extending the approach to dynamic environments with moving obstacles. Furthermore, other interesting aspects to analyze will be the effects of communication delays, IMU drifts, and other environmental disturbances in dynamic environments.

Author Contributions: Conceptualization, H.M. and M.M.Y.; methodology, H.M. and M.M.Y.; software, H.M. and M.M.Y.; validation, J.N.Y. and S.J.; writing—original draft preparation, H.M., M.M.Y., J.N.Y., S.J., M.-H.H. and J.P.; writing—review and editing, M.-H.H., M.M.Y. and J.P.; supervision, M.M.Y., M.-H.H. and J.P.; funding acquisition, M.M.Y., M.-H.H. and J.P; project administration, M.M.Y. and J.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been financially supported by the Academy of Finland funded projects 335512—ADAFI (Adaptive Fidelity Digital Twins for Robust and Intelligent Control Systems) and 330493—AURORA (Autonomous Performance Management in Digital Manufacturing).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Hinchey, M.G.; Sterritt, R.; Rouff, C. Swarms and Swarm Intelligence. *Computer* **2007**, *40*, 111–113. [CrossRef]
2. Brambilla, M.; Ferrante, E.; Birattari, M.; Dorigo, M. Swarm robotics: A review from the swarm engineering perspective. *Swarm Intell.* **2013**, *7*, 1–41. [CrossRef]
3. Yasin, J.N.; Haghbayan, M.H.; Yasin, M.M.; Plosila, J. Swarm formation morphing for congestion-aware collision avoidance. *Heliyon* **2021**, *7*, e07840. . [CrossRef] [PubMed]
4. Grocholsky, B.; Keller, J.; Kumar, V.; Pappas, G. Cooperative air and ground surveillance. *IEEE Robot. Autom. Mag.* **2006**, *13*, 16–25. [CrossRef]
5. Besada, J.A.; Bergesio, L.; Campaña, I.; Vaquero-Melchor, D.; López-Araquistain, J.; Bernardos, A.M.; Casar, J.R. Drone Mission Definition and Implementation for Automated Infrastructure Inspection Using Airborne Sensors. *Sensors* **2018**, *18*, 1170. [CrossRef] [PubMed]
6. Yasin, J.N.; Mohamed, S.A.S.; Haghbayan, M.H.; Heikkonen, J.; Tenhunen, H.; Plosila, J. Unmanned Aerial Vehicles (UAVs): Collision Avoidance Systems and Approaches. *IEEE Access* **2020**, *8*, 105139–105155. [CrossRef]
7. Madridano,Á.; Al-Kaff, A.; Martín, D.; de la Escalera., de la, A. 3D Trajectory Planning Method for UAVs Swarm in Building Emergencies. *Sensors* **2020**, *20*, 642. [CrossRef]
8. Tagliabue, A.; Kamel, M.; Verling, S.; Siegwart, R.; Nieto, J. Collaborative transportation using MAVs via passive force control. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 5766–5773. [CrossRef]
9. Udroiu, R.; Deaconu, A.M.; Nanau, C. Data Delivery in a Disaster or Quarantined Area Divided into Triangles Using DTN-Based Algorithms for Unmanned Aerial Vehicles. *Sensors* **2021**, *21*, 3572. [CrossRef]
10. Shakhatreh, H.; Sawalmeh, A.H.; Al-Fuqaha, A.; Dou, Z.; Almaita, E.; Khalil, I.; Othman, N.S.; Khreishah, A.; Guizani, M. Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges. *IEEE Access* **2019**, *7*, 48572–48634. [CrossRef]
11. Kegeleirs, M.; Grisetti, G.; Birattari, M. Swarm SLAM: Challenges and Perspectives. *Front. Robot. AI* **2021**, *8*, 618268. [CrossRef]
12. Cieslewski, T.; Choudhary, S.; Scaramuzza, D. Data-Efficient Decentralized Visual SLAM. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 2466–2473. [CrossRef]
13. Lajoie, P.Y.; Ramtoula, B.; Wu, F.; Beltrame, G. Towards Collaborative Simultaneous Localization and Mapping: A Survey of the Current Research Landscape. *Field Robot.* **2022**, *2*, 971–1000. [CrossRef]
14. Mertens, J.C.; Knies, C.; Diermeyer, F.; Escherle, S.; Kraus, S. The Need for Cooperative Automated Driving. *Electronics* **2020**, *9*, 754. [CrossRef]
15. Malik, S.; Khan, M.A.; El-Sayed, H. Collaborative Autonomous Driving—A Survey of Solution Approaches and Future Challenges. *Sensors* **2021**, *21*, 3783. [CrossRef] [PubMed]
16. Kerner, B.S. Failure of classical traffic flow theories: Stochastic highway capacity and automatic driving. *Phys. A Stat. Mech. Its Appl.* **2016**, *450*, 700–747. [CrossRef]
17. Knies, C.; Hermansdorfer, L.; Diermeyer, F. Cooperative Maneuver Planning for Highway Traffic Scenarios based on Monte-Carlo Tree Search. In Proceedings of the AAET 2019—Automatisiertes und vernetztes Fahren, Montreal, QC, Canada, 13–17 May 2019.
18. Ebadi, K.; Chang, Y.; Palieri, M.; Stephens, A.; Hatteland, A.; Heiden, E.; Thakur, A.; Funabiki, N.; Morrell, B.; Wood, S.; et al. LAMP: Large-Scale Autonomous Mapping and Positioning for Exploration of Perceptually-Degraded Subterranean Environments. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 80–86. [CrossRef]
19. Wang, Y.; Sun, Z.; Xu, C.Z.; Sarma, S.E.; Yang, J.; Kong, H. LiDAR Iris for Loop-Closure Detection. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 25–29 October 2020; pp. 5769–5775. [CrossRef]
20. Chang, Y.; Ebadi, K.; Denniston, C.E.; Ginting, M.F.; Rosinol, A.; Reinke, A.; Palieri, M.; Shi, J.; Chatterjee, A.; Morrell, B.; et al. LAMP 2.0: A Robust Multi-Robot SLAM System for Operation in Challenging Large-Scale Underground Environments. *arXiv* **2022**. [CrossRef]
21. Dubé, R.; Cramariuc, A.; Dugas, D.; Sommer, H.; Dymczyk, M.; Nieto, J.; Siegwart, R.; Cadena, C. SegMap: Segment-based mapping and localization using data-driven descriptors. *Int. J. Robot. Res.* **2020**, *39*, 339–355. [CrossRef]
22. Huang, Y.; Shan, T.; Chen, F.; Englot, B. DiSCo-SLAM: Distributed Scan Context-Enabled Multi-Robot LiDAR SLAM With Two-Stage Global-Local Graph Optimization. *IEEE Robot. Autom. Lett.* **2022**, *7*, 1150–1157. [CrossRef]

23. Bartashevich, P.; Koerte, D.; Mostaghim, S. Energy-saving decision making for aerial swarms: PSO-based navigation in vector fields. In Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI), Honolulu, HI, USA, 27 November–1 December 2017; pp. 1–8. [CrossRef]
24. Al-Sabban, W.H.; Gonzalez, L.F.; Smith, R.N. Wind-energy based path planning for Unmanned Aerial Vehicles using Markov Decision Processes. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, German, 6–10 May 2013; pp. 784–789. [CrossRef]
25. Yasin, J.N.; Mahboob, H.; Haghbayan, M.H.; Yasin, M.M.; Plosila, J. Energy-Efficient Navigation of an Autonomous Swarm with Adaptive Consciousness. *Remote Sens.* **2021**, *13*, 1059. [CrossRef]
26. Narayanan, K.; Honkote, V.; Ghosh, D.; Baldev, S. Energy Efficient Communication with Lossless Data Encoding for Swarm Robot Coordination. In Proceedings of the 2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID), Delhi, India, 5–9 January 2019; pp. 525–526. [CrossRef]
27. Majd, A.; Loni, M.; Sahebi, G.; Daneshtalab, M. Improving Motion Safety and Efficiency of Intelligent Autonomous Swarm of Drones. *Drones* **2020**, *4*, 48. [CrossRef]
28. Yasin, J.N.; Mohamed, S.A.S.; Haghbayan, M.H.; Heikkonen, J.; Tenhunen, H.; Yasin, M.M.; Plosila, J. Energy-Efficient Formation Morphing for Collision Avoidance in a Swarm of Drones. *IEEE Access* **2020**, *8*, 170681–170695. [CrossRef]
29. Tseng, C.M.; Chau, C.K.; Elbassioni, K.M.; Khonji, M. Flight tour planning with recharging optimization for battery-operated autonomous drones. *arXiv* **2017**. [CrossRef]
30. Alyassi, R.; Khonji, M.; Karapetyan, A.; Chau, S.C.K.; Elbassioni, K.; Tseng, C.M. Autonomous Recharging and Flight Mission Planning for Battery-Operated Autonomous Drones. *IEEE Trans. Autom. Sci. Eng.* **2022**, 1–13. [CrossRef]
31. Dubé, R.; Gollub, M.G.; Sommer, H.; Gilitschenski, I.; Siegwart, R.; Cadena, C.; Nieto, J. Incremental-Segment-Based Localization in 3-D Point Clouds. *IEEE Robot. Autom. Lett.* **2018**, *3*, 1832–1839. [CrossRef]
32. Han, Q.; Li, T.; Sun, S.; Villarrubia, G.; de la Prieta, F. "1-N" Leader-Follower Formation Control of Multiple Agents Based on Bearing-Only Observation. In Proceedings of the Advances in Practical Applications of Agents, Multi-Agent Systems, and Sustainability: The PAAMS Collection, Salamanca, Spain, 3–4 June 2015; Demazeau, Y.; Decker, K.S.; Bajo Pérez, J.; de la Prieta, F., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 120–130.
33. Dimidov, C.; Oriolo, G.; Trianni, V. Random Walks in Swarm Robotics: An Experiment with Kilobots. In Proceedings of the Swarm Intelligence, Brussels, Belgium, 7–9 September 2016; Dorigo, M.; Birattari, M.; Li, X.; López-Ibáñez, M.; Ohkura, K.; Pinciroli, C.; Stützle, T., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 185–196.
34. Kegeleirs, M.; Garzón Ramos, D.; Birattari, M. Random Walk Exploration for Swarm Mapping. In Proceedings of the Towards Autonomous Robotic Systems, London, UK, 3–5 July 2019; Althoefer, K.; Konstantinova, J.; Zhang, K., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 211–222.
35. Rone, W.; Ben-Tzvi, P. Mapping, localization and motion planning in mobile multi-robotic systems. *Robotica* **2013**, *31*, 1–23. [CrossRef]
36. Allen, J.M.; Joyce, R.; Millard, A.G.; Gray, I. The Pi-puck Ecosystem: Hardware and Software Support for the e-puck and e-puck2. In Proceedings of the Swarm Intelligence, Barcelona, Spain, 26–28 October 2020; Dorigo, M.; Stützle, T.; Blesa, M.J.; Blum, C.; Hamann, H.; Heinrich, M.K.; Strobel, V., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 243–255.
37. Rosinol, A.; Abate, M.; Chang, Y.; Carlone, L. Kimera: An Open-Source Library for Real-Time Metric-Semantic Localization and Mapping. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020, pp. 1689–1696. [CrossRef]
38. Dubé, R.; Gawel, A.; Sommer, H.; Nieto, J.; Siegwart, R.; Cadena, C. An online multi-robot SLAM system for 3D LiDARs. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 1004–1011. [CrossRef]
39. Schmuck, P.; Chli, M. CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams. *J. Field Robot.* **2019**, *36*, 763–781 [CrossRef]
40. Karrer, M.; Schmuck, P.; Chli, M. CVI-SLAM—Collaborative Visual-Inertial SLAM. *IEEE Robot. Autom. Lett.* **2018**, *3*, 2762–2769. [CrossRef]
41. Chang, Y.; Tian, Y.; How, J.P.; Carlone, L. Kimera-Multi: A System for Distributed Multi-Robot Metric-Semantic Simultaneous Localization and Mapping. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 11210–11218. [CrossRef]
42. Choudhary, S.; Carlone, L.; Nieto, C.; Rogers, J.; Christensen, H.I.; Dellaert, F. Distributed mapping with privacy and communication constraints: Lightweight algorithms and object-based models. *Int. J. Robot. Res.* **2017**, *36*, 1286–1311. [CrossRef]

43. Lajoie, P.Y.; Ramtoula, B.; Chang, Y.; Carlone, L.; Beltrame, G. DOOR-SLAM: Distributed, Online, and Outlier Resilient SLAM for Robotic Teams. *IEEE Robot. Autom. Lett.* **2020**, *5*, 1656–1663. [CrossRef]
44. Yasin, J.N.; Mahboob, H.; Jokinen, S.; Haghbayan, H.; Yasin, M.M.; Plosila, J. Partial Swarm SLAM for Intelligent Navigation. In Proceedings of the Advances in Practical Applications of Agents, Multi-Agent Systems, and Complex Systems Simulation. The PAAMS Collection, L'Aquila, Italy, 13–15 July 2022; Dignum, F.; Mathieu, P.; Corchado, J.M.; De La Prieta, F., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 435–446.

sensors

MDPI

Review

A Review of Sensing Technologies for Indoor Autonomous Mobile Robots

Yu Liu, Shuting Wang, Yuanlong Xie *, Tifan Xiong and Mingyuan Wu

School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China; yu_liu_d@hust.edu.cn (Y.L.); wangst@hust.edu.cn (S.W.); Xiongtf@hust.edu.cn (T.X.); mingyuan_wu@hust.edu.cn (M.W.)

* Correspondence: yuanlongxie@hust.edu.cn

Abstract: As a fundamental issue in robotics academia and industry, indoor autonomous mobile robots (AMRs) have been extensively studied. For AMRs, it is crucial to obtain information about their working environment and themselves, which can be realized through sensors and the extraction of corresponding information from the measurements of these sensors. The application of sensing technologies can enable mobile robots to perform localization, mapping, target or obstacle recognition, and motion tasks, etc. This paper reviews sensing technologies for autonomous mobile robots in indoor scenes. The benefits and potential problems of using a single sensor in application are analyzed and compared, and the basic principles and popular algorithms used in processing these sensor data are introduced. In addition, some mainstream technologies of multi-sensor fusion are introduced. Finally, this paper discusses the future development trends in the sensing technology for autonomous mobile robots in indoor scenes, as well as the challenges in the practical application environments.

Keywords: sensing technology; SLAM; obstacle avoidance; sensor fusion

Citation: Liu, Y.; Wang, S.; Xie, Y.; Xiong, T.; Wu, M. A Review of Sensing Technologies for Indoor Autonomous Mobile Robots. *Sensors* **2024**, *24*, 1222. https://doi.org/10.3390/s24041222

Academic Editor: Andrey V. Savkin

Received: 3 January 2024
Revised: 4 February 2024
Accepted: 12 February 2024
Published: 14 February 2024

1. Introduction

In recent times, there has been a rapid development in autonomous mobile robots (AMRs). Due to their excellent work ability, AMRs are being used to replace humans in some indoor scenarios. Mobile robots can help reduce the burden of human work and improve production efficiency. They have been widely used in many industries and services, such as warehousing, logistics, healthcare, restaurant service, and personal services [1–3].

The basics of mobile robotics consist of the fields of locomotion, perception, cognition, and navigation [4]. The locomotion problem is mainly concerned with the motion system of the mobile robot. The motion system design is based on the requirements of the provided services, such as the motion environment, controllability, efficiency, stability, and other relevant indicators. Perception involves acquiring information about the mobile robot's working environment and itself. Cognition involves analyzing and processing data from the perception system and providing control solutions to the motion system to accomplish the mobile robot's tasks. Mobile robot navigation relies on perception, cognition, and motion control to move from a starting point to a task goal point in a work environment, whether known or unknown. Perception is a crucial part of mobile robotics research. For mobile robots to safely and efficiently perform mobile tasks, they must sense their environment and make decisions based on that information. If a mobile robot cannot perceive its environment accurately and efficiently, then it will not be able to perform even simple tasks [5].

The perception system of the mobile robot utilizes relevant sensing techniques to provide information about the environment, the robot itself, and the relationship between the two. This information is then used for path planning, controlling the robot's movement, and ultimately completing the navigation task. The main tasks of the perceptual system include localization, map building, object detection, etc.

Localization is the process of estimating a mobile robot's position relative to the surrounding work area during its movement. This information serves as the foundation for the mobile robot's navigation [6–8]. Mobile robot localization techniques include inertial navigation, global positioning systems (GPS), active beacon-based navigation, landmark navigation, and model matching. Inertial navigation is a navigation method that utilizes data from sensors responding to inertial forces. The global positioning system (GPS) is a satellite-based technology that provides accurate information on longitude, latitude, and altitude for any location on Earth. It is a highly effective technique for locating mobile robots in outdoor environments. However, GPS is of limited use for localizing in indoor environments, due to non-line-of-sight and signal-blocking problems in complex environments [9]. Active beacon-based navigation and landmark navigation use relative positional relationships to beacons or landmarks with known locations in the environment for localization, as shown in Figure 1a. Model matching is a method of positioning by matching sensor data with specific map model data. For instance, in 2D LiDAR localization, the LiDAR measurement obtains 2D laser point cloud data, and the map model is an occupancy grid map. The mobile robot localization is performed by matching the point cloud data with the occupancy grid map, as shown in Figure 1b.

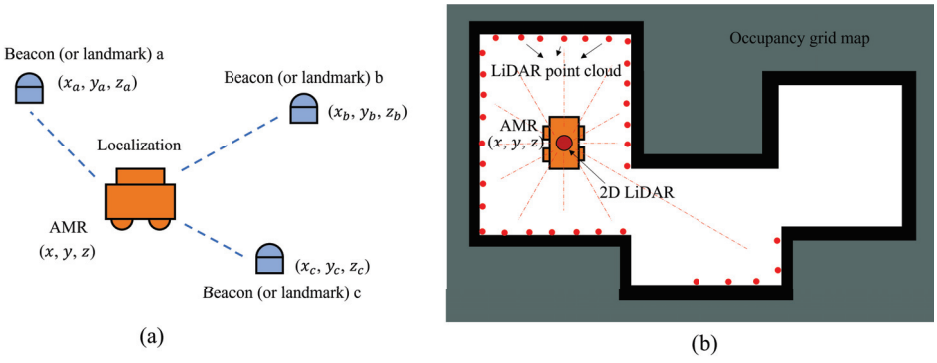


Figure 1. Schematic diagram of the localization method: (**a**) the localization method base on active beacon or landmark, and (**b**) the localization method base on model matching of 2D LiDAR.

Map building typically involves Simultaneous Localization and Mapping (SLAM), which is a crucial research field for mobile robots. SLAM is a technique used by mobile robots to determine their location in an unknown environment while simultaneously constructing an accurate map of the environment. The localization component estimates the robot's position on an existing map, while the mapping component constructs the environment's map [10–12]. The two components are interdependent in completing the map building process and enhancing accuracy through continuous iteration. The two most frequently used sensors in SLAM are LIDAR and vision cameras. SLAM techniques can be classified into two main categories based on short- and long-time optimization processing: filter-based methods and graph optimization-based methods. Filter-based SLAM algorithms typically use various types of filters to estimate and optimize the robot's trajectory and map, as shown in Figure 2. The two most common filter-based algorithms are the Kalman filter SLAM algorithm and the particle filter SLAM algorithm.

The graph optimization algorithm mainly consists of two processes, front-end and back-end, as shown in Figure 3a. The graph optimization algorithm models the robot's trajectory and the environment's topology as a graph, and minimizes the error by optimizing the nodes and edges in the graph to estimate the robot's position and obtain a mao of the environment, as shown in Figure 3b. Nodes generally represent the variables to be optimized in the SLAM process of the sensor, such as sensor position, feature point location, estimated trajectory, etc. Edges represent the errors or constraints between the

variables. The optimization algorithm is iterated to minimize the error of the edges to find the optimal solution [13].

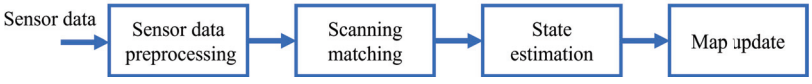


Figure 2. Schematic diagram of the SLAM method based on filter.

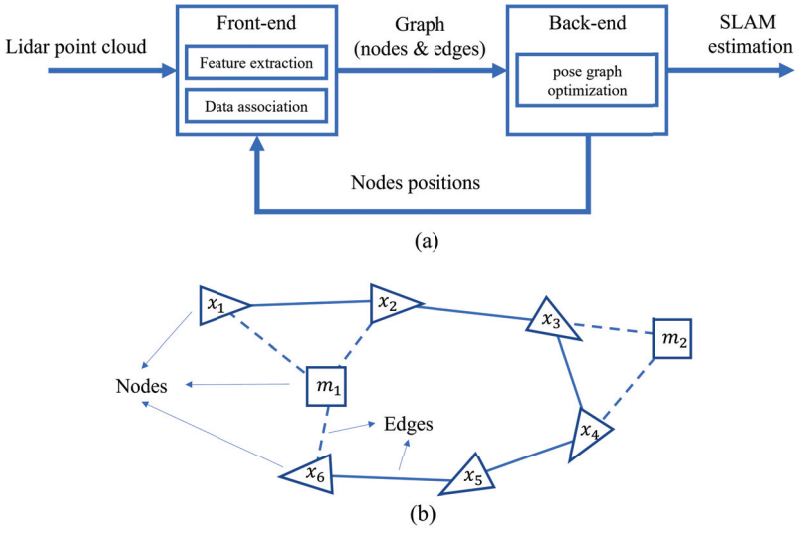


Figure 3. Schematic diagram of the SLAM method based on graph optimization: (**a**) representation of a graph optimization SLAM process, and (**b**) schematic representation of the graph optimization algorithm.

With the development of artificial intelligence, the emergence of reinforcement learning gradually allows mobile robots to gradually discard the limitations associated with a priori maps, and achieve the judgement of navigation actions only through image ingestion [14,15]. The main logic is: to set up a series of exploration reward mechanisms around the mobile robot exploration problem, such as: arrival reward, collision reward, curiosity reward, etc., and to establish the relevant neural network through reinforcement learning methods, and to train the network using the data set. Finally, the navigation decision of the autonomous mobile robot is realised.

To accomplish the navigation task efficiently and accurately, the mobile robot must avoid colliding with obstacles. Obstacle avoidance is a crucial task for autonomous mobile robots moving in uncertain environments [16,17]. Obstacle avoidance focuses on the problem by obtaining the global position information of the robot relative to the surrounding environment to the mobile robot only needs to obtain the local position information of the close objects around the robot relative to itself. Obstacle detection is crucial for perception systems as well. The mobile robot's cognition system performs path planning to avoid obstacles based on the distance and direction information about obstacles obtained from the perception system. In principle, range sensors (ultrasonic sensors and LiDAR) are highly suitable for obstacle detection tasks. In addition, vision cameras can recognize obstacles in the environment and even predict the trajectory of dynamic obstacles.

The perception system achieves these functions through the use of sensors and extracting relevant information from their measurements. Sensors used for data collection are categorized into two major aspects: proprioceptive/exteroceptive sensors and active/passive sensors [3]. Proprioceptive sensors gather data about the interior of the mobile robot, such as motor speed, wheel rotation angle, etc. Currently, encoders [18], accelerom-

eters, and gyroscopes [19] are widely used in this type of sensor. Exteroceptive sensors acquire information about the environment around the mobile robot, such as images, sound waves, and wireless signals. Examples of exteroceptive sensors include vision sensors [20], ultrasonic sensors [21], and LiDAR [22]. Active sensors send energy to the external environment and then measure the environmental feedback, such as sonar sensors, LIDAR, etc. Passive sensors measure the environmental energy coming into the sensor, such as vision sensors (Charge Coupled Device (CCD) or Complementary Metal Oxide Semiconductor (CMOS) cameras).

This paper describes the functions and principles related to the perception system of mobile robots. Further, this paper analyzes the application of various types of sensors in sensing technology. Additionally, the research results for multi-sensor fusion are presented here. The rest of this article is organized as follows: Section 2 describes some extensively adopted sensors; Section 3 presents the mainstream multi-sensor fusion schemes; and Section 4 discusses perception techniques and potential technology trends.

2. Overview of Single Sensor Sensing Technologies

2.1. Inertial Measurement Unit (IMU)

An IMU is an electronic device that integrates an accelerometer, gyroscope, and magnetometer to measure an object's acceleration, angular velocity, and the direction of the geomagnetic field [23]. The IMU is the proprioceptive sensor. The Inertial Navigation System (INS) is a dead-reckoning navigation system that uses the Inertial Measurement Unit (IMU) and is commonly used for mobile robot navigation. Magnetometer-measured orientation parameters have low accuracy and are challenging to use for precise navigation tasks. Therefore, the most frequently used parameters in IMUs are acceleration and angular velocity. The principle of dead reckoning is demonstrated in Figure 4. By using the considered IMU, the trajectory of an indoor AMR is determined by integrating the acceleration and angular velocity based on the initial position, direction, and velocity.

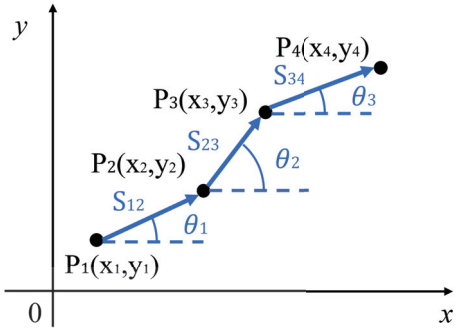


Figure 4. IMU dead reckoning.

An IMU has two typical advantages: a high output update rate and immunity to external interference. These advantages are irreplaceable when a high data update rate or high reliability are required, or when the external signal acquired by the mobile robot system is unreliable.

However, IMUs have limitations when applied to mobile robots. IMUs have two typical disadvantages:

1. The calculation process must depend on the initial conditions.
2. The navigation error increases with time, which is due to the need for double integration of its measurements.

It should be mentioned that its duration and sensor accuracy have a significant impact on the accuracy of IMU-based navigation [23]. To overcome these disadvantages, an IMU is frequently used in fusion with other sensors and as auxiliary measurements in other

navigation methods. L. Luo et al. [24] designed a positioning method that combines an IMU, Bluetooth, and a magnetometer for the mobile robot positioning problem. In the above-mentioned method, the inertial navigation system (INS) serves as the core, and the Bluetooth AOA positioning base station observes the position, while the magnetometer observes the heading angle. Alatise M. B. et al. [25] fused IMU and vision sensors in order to achieve accurate positioning of a mobile robot. The accelerometer data are used as inputs to the control, while gyroscope and image data are used as observations for position pose estimation using extended Kalman filtering. Zhang et al. [26] developed an algorithmic framework for the indoor localization of mobile robots by combining a wheel odometer, an inertial navigation unit (IMU), and Ultra-Wideband (UWB). The framework uses a fusion method with extended Kalman filtering, where the wheel odometer provides localization information as a prediction and the IMU and UWB provide attitude angle and position information as observations.

2.2. Ultrasonic Sensor/Sonar

Ultrasound refers to sound waves that vibrate at frequencies above 20 kHz, which are beyond the upper limit of human hearing. Sonar also employs ultrasound. Ultrasonic sensors are typically distance measurement sensors. Distance measurement sensors can be classified into two types of measurement methods: the reflection-type distance measuring method and the unidirectional distance measuring method, as shown in Figure 5. In the reflection-type method, the sensor sends a signal to the surrounding environment, which is then reflected when it encounters an obstacle. The sensor then receives the reflected signal to measure the distance between the sensor and the obstacle. The unidirectional distance-measuring method involves placing the transmitter and receiver in different locations. The receiver measures the distance between the two by accepting the signal from the transmitter. Ultrasonic ranging measures distance using three main criteria: time, phase, and acoustic vibration.

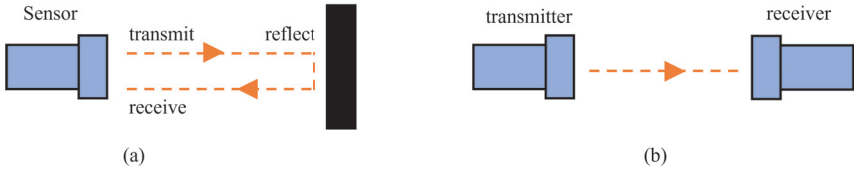


Figure 5. Sensor distance measuring method: (**a**) reflection-type distance measuring method. (**b**) unidirectional distance measuring method.

Ultrasonic sensors typically use a reflection-type distance measuring method to determine the distance to an object, which is used for mobile robot positioning and navigation. However, the sonar sensor's ultrasound beam width is too wide to precisely determine the object's direction. Additionally, specular reflection on object surfaces often results in inaccurate distance calculations, reducing the reliability of distance measurements from sonar sensors. It is not possible to construct a complete grid map using erroneously measured sonar data when the frequency of erroneous distance measurements is too high. To overcome this problem, S.-J. Lee et al. [27] designed a method for constructing occupancy grid maps for mobile robots using sonar data. This method assigns weights to each sonar data point based on its geometric reliability to minimize the impact of incorrect sonar data on the grid map construction. Anomalous sonar data are identified and assigned a lower geometric reliability value through a morphological comparison with neighboring sonar data. H. Liu et al. [28] used sonar sensors to overcome the limitations of sensor performance by accumulating sonar data to achieve room-level localization. Y. Liu et al. [29] presented a mobile robot on the basis of LEGO MINDSTORM NXT, in which an ultrasonic sensor was mounted on a 360° rotating motor and a particle-filtering approach was used to implement its localization function. Mingqi Shen et al. [30] provided a localization

method for autonomous mobile robots based on multiple ultrasonic sensors. The method calculates the target's coordinates based on the ratio of three flight times measured by three ultrasonic sensors. This improves the localization accuracy by avoiding the effect of ambient temperature on localization.

In addition, ultrasonic sensors can also be used for the unidirectional distance measuring method, which has been applied in the localization of indoor mobile robots. R. Li et al. [21] set up an ultrasonic receiver on the robot and had multiple ultrasonic generators in the scene. The three-sided localization principle is used to achieve the localization of the mobile robot. Chen-Chien Hsu et al. [31] presented a localization method based on omnidirectional ultrasonic sensing. The proposed method utilizes reflective cones to generate a 360° propagating ultrasonic beam to address the detection angle limitation of conventional ultrasonic sensors. Ultrasonic waves are emitted directly onto a cone, which reflects them in all directions. The ultrasonic signal is received by the installed receiver in the environment. The mobile robot's position is determined through the ToF (time-of-flight) method and the trilateration localization method.

Ultrasonic sensors are susceptible to environmental noise and long-range strength degradation [32] and are not accurate enough to measure detailed features of the environment. There are relatively few cases where ultrasonic sensors are used for positioning in high-precision indoor navigation applications. However, ultrasonic sensors have a noticeable advantage in obstacle detection. They are highly accurate in detecting highly transmissive or reflective objects, such as glass and mirrors, which can be challenging for many optical sensors. H. Takai et al. [33] used ultrasonic stereo sonar and a single image sensor to detect obstacles in the workspace. J.-H. Jean et al. [34] mainly used the environmental distance information collected by ultrasonic sensors to guide the robot along the wall baseline using a steering controller based on the potential field method. Grami T. et al. [35] designed a mobile robot equipped with seven ultrasonic sensors and used a particle-filtering approach to implement its localization function. The mobile robot can move freely while avoiding obstacles. Maryna Derkach et al. [36] utilized four ultrasonic sensors for obstacle avoidance in mobile robots. Their proposed algorithm uses a linear recursive Kalman filter to process the sensor data, allowing the robot to avoid additional obstacles.

2.3. Infrared Sensor

Infrared is an electromagnetic wave with a wavelength between visible light and microwaves. Infrared sensors use an emitter to emit an infrared beam and a receiver to measure the intensity of the beam. The intensity of the infrared beam decreases as the distance traveled increases. Infrared sensors can be used to measure distance by assessing the intensity of the beam, in addition to the ToF (time-of-flight) method. Similar to ultrasonic sensors, infrared sensors use reflection-type and unidirectional distance measuring methods.

There are many early studies applying infrared sensors to mobile robot localization. Eric Brassart et al. [37] designed a mobile robot localization system based on infrared beacons. In this system, a ceiling-mounted infrared beacon sends an encoded infrared signal, which is then received by a CCD camera mounted on the mobile robot. The robot's location is determined using triangulation and trilateration methods. J. Krejsa et al. [38] presented an interior localization method for mobile robots that uses infrared beacons. The beacons are placed at known locations in the environment, and a ring beacon scanning device with 16 receivers is used to receive the infrared signals. The scanner is mounted on the robot to measure the relative angle between the robot and the beacon. The position of the mobile robot is estimated by fusing orientation information with motion controller commands or an odometer through an extended Kalman filter. Infrared sensors can be used to track targets when the transmitter and receiver are mounted on different robots. Tzuu-Hseng S. Li et al. [39] have designed a tracking control scheme for an autonomous mobile robot by using infrared sensors. The scheme involves a target mobile robot with

an infrared reflector and a tracking mobile robot with an infrared receiver and a reflective sensor. The latter was designed to track the target robot. Juang J. G. et al. [40] designed wheeled mobile robots with integrated ultrasonic and infrared sensors that can move along walls and avoid obstacles. Ultrasonic and infrared sensors were used to detect obstacles and identify unknown environments. Infrared-based localization technology is currently mature. However, it is difficult for it to penetrate objects and is susceptible to its surroundings. Additionally, obstacles can directly block infrared rays in the infrared beacon-based localization method. As a result, the application of infrared sensors in mobile robot localization is limited.

In obstacle detection, infrared sensors are frequently used because of their rapid response and compact size [41–43]. Jianwei Zhao et al. [44] designed a multi-sensor mobile robot that uses infrared sensors mounted in front of the two front wheels at the bottom of the robot to detect ground pits.

2.4. LiDAR

LiDAR (Light Detection and Ranging) uses laser beams to measure the position and distance of target objects. Multiple laser beams are sent out from transmitters, and when they hit objects, they reflect back. The reflected beams are received by the LiDAR receivers. LiDAR measures the distance traveled by each beam using time-of-flight (ToF) and combines the distance information from all the laser beams to obtain information about the surroundings. LiDAR operates similarly to radar, but with a higher accuracy, measurement speed, and measurement distance compared to conventional radar.

In indoor mobile robot navigation, mobile robots mainly use 2D LIDAR to sense environmental information, while 3D LIDAR is more often used in the field of intelligent driving because it needs more information to process [45]. LiDAR is widely used for mobile robot localization due to its excellent distance measurement accuracy. Localization with LiDAR is based on an environment map [46]. Dirk Hähnel et al. [47] presented an algorithm for acquiring 3D models with mobile robots. Xipeng Wang et al. [48] devised a method for global localization using only schematic floor plans as prior maps. The method achieves global localization by matching features observed from LiDAR with features in the floor plan. The method performs comparably to a baseline system using a conventional LiDAR-based prior map. Haofei Kuang et al. [49] proposed a global localization method for mobile robots using 2D LiDAR. In this method, the neural occupancy field uses a neural network to implicitly represent the scene, and 2D LiDAR scans of arbitrary robot poses are synthesized by using a pre-trained network. The MCL system integrates the similarity between the synthesized scans and the actual scans as an observation model to perform accurate localization based on the implicit representation.

As LiDAR distance measurement is based on the principle of light reflection, the measurement has a large error in environments with glass obstacles or mirror obstacles. To overcome the limitations, J. Kim et al. [50] proposed a method for LIDAR-based localization schemes in glass wall environments. In this method, all candidate distances from the glass wall are pre-calculated based on the occupancy grid map and the reflection characteristics of the laser beam. The type of reflection phenomenon on the glass surface is then estimated on the basis of actual measurements. The robot's local position tracking is performed by using a scan-matching method that considers the estimated results. Additionally, the fusion of ultrasonic sensors and LiDAR during obstacle detection can achieve a higher detection accuracy as ultrasonic sensors are not affected by glass or mirror obstacles.

In the absence of a prior environment map, LiDAR SLAM uses sensors to measure the surrounding environment and obtain a LiDAR point cloud, which is then used to estimate its position and construct a map based on the position information.

Typical 2D LiDAR SLAM methods include Gmapping [51], Hector SLAM [52], Karto SLAM [53], and Cartographer [54]. The Gmapping algorithm framework is based on the RBPF (Rao-Blackwellized Particle Filters) algorithm, and Gmapping makes two main improvements, including improving proposal distribution and selectively resampling.

The Hector SLAM algorithm framework is based on the EKF (extended Kalman filter) algorithm. Hector SLAM can be used without an odometer and is suitable for airborne or uneven road environments, but it is prone to drift errors when the LiDAR is rotating too fast. Karto SLAM is the first open-source algorithm based on the PGO (pose graph optimization) algorithm. It relies on efficient linear matrix construction and sparse non-iterative Cholesky decomposition to efficiently represent and solve large sparse pose graphs. The most important point of Karto SLAM is the introduction of back-end optimization and loop closure detection. Compared with Gmapping and Hector SLAM, Karto SLAM is more advantageous for mapping in large environments. Cartographer is based on the PGO (pose graph optimization) algorithm. It eliminates the cumulative errors generated during the map building process mainly through loop closure detection, in which the submap is the basic unit. It accelerates loop closure detection by combining local and global data and has high real-time performance, which makes it a reliable 2D LIDAR SLAM method.

In recent years, many improvements have been made based on these SLAM methods. Xiang Y. et al. [55] designed a LIDAR-based localization and mapping method for indoor mobile robots. The method makes improvements in distribution and limiting the number of resamplings based on the original RBPF-SLAM algorithm. Han Wang et al. [56] proposed an intensity-assisted SLAM framework for LiDAR-based localization systems. The proposed SLAM includes intensity-based front-end odometry estimation and intensity-based back-end optimization. Yingzhong Tian et al. [57] proposed a method to improve the traditional ICP method by incorporating intensity into the calculation. This method reduces the number of iterations and arithmetic required. The method uses an objective function that considers the distance and intensity residuals of the LiDAR point cloud to determine the optimal initial transformation estimation. Weiwei Hu et al. [58] proposed keyframe extraction by clustering 2D LiDAR point clouds in indoor environments. In this method, firstly, the dimension of the scan is reduced to a histogram, and the key frames are extracted using the histogram. Then, the laser points in the key frames are divided into different regions using a region-segmentation method. Next, the points are clustered in separate regions, and the point sets from neighboring regions are merged. Finally, the sets with laser points below a threshold are discarded as anomalous clusters. Saike Jiang et al. [59] designed an autonomous navigation method for indoor mobile robots by fusing LiDAR, IMU, and encoder data. The method optimizes and combines the 3D point cloud information from multi-line LiDAR into a 2D LiDAR point cloud. This reduces computational power consumption in the SLAM process and improves map building efficiency. Additionally, this method can recognize obstacles at varying heights, overcoming the limitations of 2D LiDAR in acquiring height-related information and enhancing the safety of mobile robot movement. Weipeng Guan et al. [60] proposed a loosely coupled multi-sensor fusion method for VLP and LiDAR SLAM. In this method, the LiDAR sensor detects the surroundings to avoid obstacles during navigation and compensates for the cumulative error of the odometer. And VLP is used to provide high-precision pose initialization and correction for the LiDAR SLAM and odometer.

For obstacle detection, compared to ultrasonic sensors, LiDAR has significant advantages in distance measurement accuracy and efficiency for obstacles. J. H. Lee et al. [61] proposed a method for tracking multiple walking humans based on 2D LiDAR. The algorithm considers the fact that 2D LiDAR is often positioned at the height of human legs and uses the geometric characteristics of the legs to detect their position. The method utilizes a pendulum model of the angle between the two legs as well as an extended Kalman filter to extract the frequency and phase of the walking motion. Mozos O. M. et al. [62] used a multilayer 2D LiDAR for person detection. The multilayer LiDAR method scans different parts of the human body using different layers. A supervised learning approach is used to obtain a classifier for each layer of LiDAR data, which is then used to detect and recognize specific body parts. The results of each classifier are ultimately combined to achieve human body detection. Ángel Manuel Guerrero-Higueras et al. [63] devised a 2D LiDAR-based method for tracking people. The 2D LiDAR is mounted at knee height in

a mobile robot, and the method is based on an offline trained full Convolutional Neural Network that can track pairs of legs in the environment. Yan, Z. et al. [64] proposed an online learning framework based on 3D LiDAR scanning for human detection. Ailing Zou et al. [65] proposed a nearest neighbor clustering algorithm based on the adaptive threshold to detect obstacles in the path of indoor robots by using 2D LiDAR.

2.5. Vision-Based Sensor

Visual sensors are devices that mimic the function of the human visual system. They capture, perceive, and interpret visible light information in the environment and convert it into digital signals or other forms of output for analysis, processing, and decision-making by robots or other systems. Compared to other sensors, visual sensors have the ability to represent the surrounding environment in the form of an image, making them capable of describing complex texture features in the environment. Moreover, vision sensors are widely utilized in the field of mobile robotics due to their relatively low cost. The main types of vision sensors are monocular cameras, binocular cameras, RGBD cameras [66], and event cameras [13].

The imaging principle of a monocular camera is similar to that of a normal camera and is usually represented by the pinhole camera model. Light enters through the camera's aperture and is projected onto an imaging plane, typically a CMOS or CCD sensor [67]. The projected image is de-distorted and aligned with pixels to present a two-dimensional image similar to what is observed by the human eye. Despite its low cost, there are still significant technical challenges in research and development due to the difficulty of directly computing depth information. A stereo camera is a camera sensing system consisting of two monocular cameras, similar to the structure of the human eye. The depth information can be calculated from the baseline distance between the optical centers of the two cameras [68]. Depth estimation of a pixel point can be achieved by the underlying geometric model, as shown in Equation (1).

$$z = \frac{fb}{d} \tag{1}$$

where z is the actual depth estimation, f is the camera focal length, b is the actual baseline distance, and d is the parallax between the pixel points in the left and right screen projections. From this equation, it can be seen that, due to the influence of the baseline distance b, a truncation error will occur if the depth z is too large, resulting in inaccurate depth estimation.

RGBD cameras are sensors obtained by integrating a depth sensor with a monocular sensor. The depth camera actively provides depth measurements at each pixel. Currently, the main depth sensors for RGBD cameras are infrared structured-light sensors and ToF (time-of-flight) sensors [69]. The main difference between the two cameras is the depth estimation principle. The infrared structured-light sensor calculates the depth information by recognizing the structured light pattern. The Time of Flight (ToF) sensor calculates the depth information of each pixel point in the camera by sending pulses of light and measuring the time it takes for the light to return. The depth map is then mapped to the RGB image using the relative position relationship between the sensors, resulting in a composite image of color and depth. However, due to the limitations of the depth sensor, searching for depth information is not feasible in large-scale and large-range spatial environments.

Event cameras are biologically inspired to work, so their imaging principles are not the same as those cameras described above. In traditional cameras, the main camera unit is a monocular camera, where the sensor captures the surrounding image at a constant frame rate to obtain environmental information. However, considering the frame rate and aperture factors, it will appear shadowy or blurred when facing environmental factors such as high dynamic and strong exposure. The event camera changes the recording method of the captured image to the recording of pixel changes. This means that the change in the pixels on the screen is recorded as an "event", and the environmental information is read by the change in pixel brightness in the image on the output screen. As a result, the event

camera can calculate quickly, with a frame rate of up to 1 KHz, and will not overexpose in high light intensity environments.

The light field camera is based on a monocular camera. A microlens array is added to the monocular camera to achieve a composite collection of environmental information. Due to the addition of the microlens array, the light sensor is able to sense light information in different directions at the same pixel point. The light field can be solved by the light information in different directions to achieve depth estimation in the surrounding environment. Similar to the binocular camera, it is able to estimate the depth of the surroundings from the line of sight in one frame. However, in contrast to the binocular camera's simple two-frame camera, the light field camera can provide richer depth information for depth estimation or scene reconstruction at a later stage. The table below (Table 1) shows the pros and cons of each visual sensor.

Table 1. The advantages and disadvantages of each sensor and the related representative SLAM work applied for the first time to autonomous mobile robot perception.

Camera Type	Application Time	Characteristic	Advantages and Disadvantages
Monocular Camera	2007 [70]	Contains only one lens, the basic unit of the visual sensor	Pros: easy to integrate, low cost Cons: cannot directly provide in-depth information
Stereo Camera	2008 [71]	Simultaneous generation of left and right images for stereoscopic visual	Pros: produces two images (left and right), and can be used for stereo visual Cons: calibrates camera, computationally complex, limited depth estimation range
RGBD Camera	2011 [72]	RGB images and depth images can be provided directly	Pros: Provides accurate depth information Cons: Depth information is susceptible to environmental influences
Event Camera	2017 [73]	Uses pixel changes as events. Captures luminance changes with microsecond time resolution	Pros: able to cope with high dynamics and scenes with large lighting variations, low power consumption Cons: requires image pre-processing to obtain traditional image information
Light Field Camera	2017 [74]	Adding microlens arrays to camera lenses to add scale information to monocular cameras	Pros: provides scale information for monocular cameras Cons: high computational volume, difficult to achieve real-time computing

In indoor environments, the effects of lighting, dynamic objects, and unstructured scenes can cause images that do not accurately represent the indoor environment. Additionally, before performing localization, the screen must be depicted as a scene that can be modified by the mobile robot. Therefore, it is necessary to pre-process the images to facilitate the extraction of environmental information by the camera.

Image pre-processing is mainly divided into the following aspects:

1: Eliminating image distortion.
2: Assigning semantic labels to specific objects. The extraction of specific objects or features in the picture can be achieved by neural networks or traditional feature extraction algorithms. Assigning semantic labels to them is more suitable for human understanding of environmental features.
3: Image Partial Reconstruction. In indoor environments, deep learning techniques like generative adversarial networks (GAN) and diffusion should be considered for reconstructing specific images due to the significant impact of light and dynamic objects on environmental representation. This reduces or reconstructs interfering factors in the image for the map construction process, such as dynamic objects, visible light, and shadows from occlusions. The goal is to acquire suitable images for scene localization.

Liu et al. [75] solved the issue of depth estimation between day and night by using the DCNN neural network. Considering the similarities and differences of the images between the environment with and without lighting, the images are divided into a private domain (lighting conditions) and a public domain (texture features). The reconstruction of a night image from a day image is achieved by the GAN neural network. This reduces the effect of the day/night boundary on the position and image construction errors. Bescos et al. [76] utilized CNN neural networks to implement multi-category semantic segmentation for the removal of dynamic objects and their accompanying shadows. GANs were employed along with a loss function based on image steganalysis techniques to achieve an absolutely static environment in the scene.

The conventional method for localizing mobile robots using visual sensors involves capturing image information with a camera and then comparing the images to determine the robot's current position [77]. It is mainly divided into two major steps: feature extraction and data association. The decision of whether to extract features is divided between the feature point method and the direct method. For the data association part, the main goal is to achieve a real-time estimation of the camera's own position change through the image comparison between frames. As the monocular camera cannot detect depth information directly, the relative positional relationship between the front and back frames of the camera corresponds to the rotation and translation in 3D space, which is achieved by establishing the rotational position between the front and back frames. That is, the pair of pole geometry problem, as shown in the following equation:

$$E = t^{\wedge}R \quad x_2^T E x_1 = 0 \quad E = \begin{bmatrix} e_1 & e_2 & e_3 \\ e_4 & e_5 & e_6 \\ e_7 & e_8 & e_9 \end{bmatrix} \tag{2}$$

where E is the essential matrix, which represents the positional transformation of the robot's translation and rotation, and x_1 and x_2 denote the normalized coordinates of the corresponding pixel points of both on the projection plane. Since e_9 can be directly designated as one, it is possible to solve the polar geometry problem for eight pairs of pixel points corresponding to the eight pairs in the frame, thus realizing the position estimation of the camera. The depth information of the environment in the camera positioning scene is solved by triangulation, that is, the positional transformation relationship between the two frames and the corresponding pixel projection coordinates in the two frames are known, so as to calculate the corresponding depth information of the pixel points. This can be shown in the following equation:

$$s_1 {x_1}^{\wedge} x_1 = 0 = s_2 {x_1}^{\wedge} R x_2 + {x_1}^{\wedge} t \tag{3}$$

Although the monocular camera can realize the position estimation of the robot from eight mutually matching pixel points, it still has problems such as the large initialization error and degradation of position estimation due to pure rotation. For the stereo camera and depth camera, they can directly glean the pixel information from the environment and its corresponding depth information. Therefore, the position calculation is relatively simple, i.e., through the PNP algorithm [78]. As shown in the following equation:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} t_1 & t_2 & t_3 & t_4 \\ t_5 & t_6 & t_7 & t_8 \\ t_9 & t_{10} & t_{11} & t_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4}$$

where u, v are the pixel points in normalized coordinates and X, Y, Z are the 3D coordinates corresponding to the pixel point.The T matrix represents the camera's positional transformation. From the above, it can be concluded that the positional estimation of the camera can be achieved by six pairs of well-matched pixel points. Due to the addition of depth information, other constraints can be added to the PNP algorithm to achieve fewer points

for the pose estimation, such as the P3P algorithm, the EPNP algorithm, etc., [68]. The event camera serves as the basic unit of the camera. Unlike monocular camera processing, the event camera performs feature matching by matching time, pixel coordinates, and light intensity changes in the event stream to generate environmental features in the frame that can be correlated with the data [79].

The position estimation of mobile robots can be implemented using a neural implicit approach due to the excellent generalization and iterative capabilities of neural networks. The traditional explicit representation of position estimation is to represent the relevant objects in the frame and the relevant features as the exact six-degree-of-freedom relevant point, line, and surface features. Neural implicit methods favor the representation of the features in the frame over functions or neural networks, such as voxels, meshes, distance fields, etc. The advantage of implicit representation is that it can effectively represent complex texture features and more effectively represent complex surfaces such as voids and surfaces as opposed to explicit point clouds or grids. Furthermore, neural implicit methods have the advantage of compact spatial representation and the ability to store a large amount of information, allowing them to represent a larger map space with less memory compared to explicit networks. By using neural networks to predict and update maps, neural implicit methods have end-to-end learning capabilities that allow for greater flexibility in updating maps and estimating poses in real-time applications. Compared to traditional feature extraction and matching-based methods, neural implicit methods have better adaptability and robustness when dealing with dynamic scenes or fast movements. Zhu et al. [80] employs a neural implicit representation to capture the geometric and appearance attributes of a scene in terms of hierarchical network features. They utilize a pre-trained neural implicit decoder encoded at different spatial resolutions. By rendering the generated depth and colour images, they optimise the feature mesh within the field of view cone by minimising the re-rendering loss function.

There are many types of visual map building, but map functions generally have the following characteristics:

1: The ability to effectively represent the characteristics of the surrounding environment.
2: The amount of computation and storage required can match the hardware carried by the robot.
3: The ability to achieve the functionality required by the mobile robot.

As for map types, the main ones are point cloud maps [81,82], 3D occupancy grid maps [83], TSDF maps [84], semantic maps [85], etc.

A point cloud map generally consists of a camera frame that represents the camera's position, the camera's historical trajectory, and a pixel point cloud. It accurately depicts the surrounding environment and the camera's position transformation. However, the redundant point cloud features result in large storage space and construction costs. Vijayanarasimhan et al. [86] introduced the SfM-Net network, which retrieves depth maps and performs camera pose estimation using a single video stream. The network generates a corresponding depth image from the input single image and fuses the depth point cloud to create a comprehensive representation. Finally, the positional relationship between the resulting images is computed based on the input pair of consecutive frames. Furthermore, the method identifies and segments any moving objects within the scene, presenting them in the form of a mask.

The 3D occupancy grid map divides space into occupied subspaces based on a certain volume. If an object is present in the occupied subspace, it is considered occupied. While it sacrifices clarity in environmental information representation compared to the point cloud map, it significantly reduces computation and map information storage requirements. This is advantageous for constructing large-scale maps. Cao et al. [87] proposed a 3D semantic occupancy raster map completion method for inferring indoor and outdoor scenes in a single RGB image. The method is applicable to a wide range of scenes and environments and does not require additional depth sensors or specific scene types. The article introduces the FLoSP mechanism to correlate 2D and 3D networks. Additionally,

a new loss function is proposed to optimize inference results by taking into account the semantic relationships and local structure of voxel groups. This improves the quality and consistency of scene completion.

TSDF maps are maps represented by the TSDF notation (Truncated Signed Distance Function). Similar to three-dimensional occupancy grid maps, TSDF divides the space equally into an infinite number of voxel representations denoted by (x, y, z, v). The map is iteratively updated by a mobile robot in a continuous positioning process. Since updating the voxels directly is required, RGBD cameras are generally used to implement TSDF map construction. Kim et al. [88] predicted 3D Truncated Signed Distance Function (TSDF) voxels from a single RGB image by integrating depth prediction and TSDF generation processes seamlessly. This approach enables the generation of highly accurate TSDF maps with greater efficiency compared to TSDF conversion from depth prediction alone. The comprehensive TSDF model is utilized to improve the precision and robustness of camera pose estimation, resulting in more effective scene facilitation.

Semantic maps differ from the above three maps. They are created by using image processing or deep learning network methods to extract the physical properties of points, lines, surfaces, or object features from an image and assign corresponding semantic information during the map construction process. Thus, we can create maps that align with human thinking and are more conducive to human–machine interactions based on the aforementioned maps. Yu et al. [89] proposed a methodology for extracting semantic segmentation information from the conventional SLAM framework. This process is executed as an independent thread within the image preprocessing module. The approach effectively identifies and eliminates dynamic objects present in each frame by combining semantic data with the concurrent detection of motion feature points. A dense semantic octree map is generated to meet the specific task requirements associated with advanced robot decision-making and path planning.

The development of mapless navigation can improve the computational speed of indoor mobile robots, which is conducive to autonomous decision making and path planning in dynamic environments. However, how to set up an effective reward mechanism and network structure is a major research challenge in mapless navigation. Kulhanek et al. [90] used auxiliary tasks consisting of reward prediction, pixel control, and semantic segmentation prediction to achieve effective training of neural net structure, and introduced LSTM as a memory module in the process of the network structure to cope with the before and after state correlation problem in the POMDP problem. Xiao et al. [91] addressed the problem of how to make a mobile robot learn to find the target point by passing through corners and such places when the initial target is not visible, creating arrival rewards, collision penalties, and proximity rewards. An "action consistency" mechanism was designed to achieve consistency between the learned behaviour of the mobile robot and its accumulated experience. Xie et al. [92] proposed a framework for a mapless navigation method for mobile robots based on a hierarchical approach with DL and DRL. It is assumed that the mobile robot can be provided with navigation instructions and images of key locations before task execution. The upper layer "watches" the navigation instructions and the current state to determine whether it is time to reach the place where it needs to act according to the navigation instructions. The lower layer is responsible for completing the navigation function of the mobile robot in between the upper layer's updated decisions.

Vision cameras have a significant advantage in obstacle recognition compared to other sensors, making them a popular choice for obstacle avoidance in mobile robots. The obstacle avoidance system designed by Tai et al. [93] does not provide real-time estimations of the robot's position during obstacle avoidance but builds a convolutional neural network for the original depth image. Subsequently, it establishes a fully connected network with five decisions on obstacle avoidance for the mobile robot, which enables the mobile robot to autonomously realize obstacle avoidance planning during indoor movement. Cristóforis et al. [94] achieved autonomous obstacle avoidance by building a

"teach-learning" system that only requires texture information about the path of the screen rather than the position estimated since localization.

The perception of indoor mobile robots covers a variety of map representations. Point cloud maps are discrete point cloud data acquired by sensors such as LIDAR or camera sensors that provide highly accurate information about the structure of the environment. Three-dimensional occupancy raster maps divide the environment into a three-dimensional grid that is used to represent the location and occupancy of objects.TSDF maps provide highly accurate geometric information by modelling the surface of an object. Semantic maps further incorporate semantic information to enable the robot to understand the semantic meaning of different areas in the environment. In contrast to traditional navigation techniques, the mapless navigation function based on image information is a lightweight and efficient navigation method. It acquires environmental images through sensors such as cameras, and uses deep reinforcement learning and other methods to achieve end-to-end perception and decision making by learning the environmental features in the images, so as to realize autonomous navigation for robots. As the method does not need to pre-construct maps, it alleviates the dependence on expensive sensors, while showing a strong adaptability in dynamic environments. Therefore, mapless navigation based on image information is robust, with a small computation and memory footprint, making it perform well on resource-constrained mobile robot systems.

2.6. Radio Frequency Technologies

Radio frequency (RF)-based sensing technology is mostly used for indoor mobile robot localization. The RF-based localization system comprises beacon stations (BS) that transmit radio signals and mobile robots that receive them. Unlike infrared and ultrasound-based beacon location systems, RF waves can penetrate doors and walls, providing ubiquitous coverage of buildings. RF-based localization systems applied to indoor mobile robots typically include: WiFi [95], Bluetooth [96], Zigbee [97], Ultra-Wideband (UWB) [98], and Radio Frequency Identification (RFID). The advantages and disadvantages of the RF techniques [99] are shown in Table 2.

Table 2. The characteristics of different RF technologies used for indoor localization.

RF	Range (m)	Cost	Advantages	Disadvantages
WiFi	250	high	Implementation simplicity, large coverage, high transmission rate	High power consumption, meterlevel accuracy, vulnerable to NLOS path
Zigbee	100	medium	Extremely low energy consumption, low system cost	Low transmission rate, high latency, vulnerable to NLOS path
Bluetooth	100	low	Implementation simplicity, low energy consumption	Small coverage, low transmission rate, vulnerable to NLOS path
UWB	100	high	High accuracy, extremely high transmission rate, low latency, immune to interference, robustness to NLOS path	High energy consumption and system cost
RFID	5	low	Extremely low energy consumption, implementation simplicity, low system cost, high accuracy with specific approach	Small coverage, vulnerable to NLOS path

2.6.1. Radio Frequency Technology Localization Algorithms

RF-based indoor localization technologies use two main methods: geometry and fingerprinting. The geometry-based localization method determines the location of the mobile robot using geometric knowledge, either by measuring the distance to fixed beacon stations (BS) or by calculating the received signal angle. Several typical RF-based positioning algorithms are Time of Arrival (TOA), Time Difference of Arrival (TDOA), Angle of Arrival (AOA), and Received Signal Strength Indication (RSSI), as shown in Figure 6.

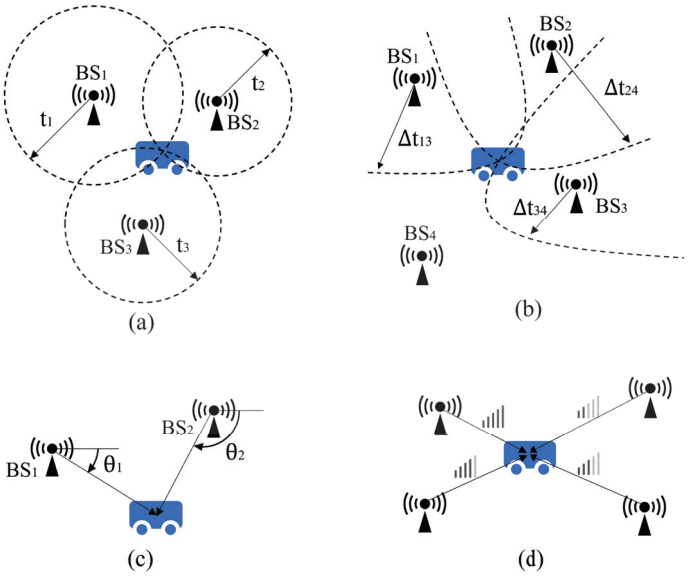


Figure 6. RF localization technologies: (**a**) Time of Arrival (TOA). (**b**) Time Difference of Arrival (TDOA). (**c**) Angle of Arrival (AOA). (**d**) Received Signal Strength Indication (RSSI).

TOA is a localization method that measures signal propagation time. A signal sent from a transmitter on a mobile robot travels through the air at approximately the speed of light. When the signal reaches receivers mounted at multiple beacon stations (BSs), the receivers record their respective arrival times [100]. The distance of signal propagation is calculated by measuring the propagation time. This allows for the localization of the mobile robot using the triangulation method, as shown in Figure 6a. The signals emitted by the transmitters are time-stamped to calculate the propagation time. It is crucial to synchronize the time of the beacon stations with the time of the mobile robot [101].

TDOA is a method used to determine the location of a signal transmitter by measuring the time difference between the signal's arrival at different receivers [102]. The arrival time of the same signal is recorded by multiple signal receivers, and the time differences between the different receivers are calculated. These time differences are then used to construct a system of equations. Each equation represents the positional constraint of the corresponding receiver on the transmitter. These constraints are reflected in the coordinate graph as hyperbolas. The position of the signal transmitter, i.e., the mobile robot, is determined by solving these equations, as shown in Figure 6b. Unlike the TOA method, the TDOA method only requires time synchronization between the beacon sites, not between them and the mobile robot [103].

AOA is a method used to determine the position of a signal transmitter by measuring the angle at which the signal arrives at different receivers [104]. The signal receiver consists of multiple signal-receiving elements, and the angle of arrival is calculated by comparing the arrival times or phase differences of the signals between the different receiving elements. The angle of arrival of a signal is recorded by a plurality of signal receivers. Geometric relationships are then used to calculate the position of the signal transmitter, which is the mobile robot, as shown in Figure 6c.

The received signal strength indicator (RSSI) is a measurement of the power of a radio signal that has been received. Signal strength decays during propagation, which is characterized by the propagation power loss model. Therefore, the strength of the received signal can also be used to calculate the signal propagation distance. Triangulation is then used to measure the position of the mobile robot [105].

Fingerprint localization is a technique used to determine location by comparing received signals with a pre-established fingerprint database [106]. It consists of fingerprint database construction and location estimation. The process involves deploying a set of RF sensor receiver stations throughout the area to collect RF signals at each sampling point and record their characteristic parameters. The sampling point locations are correlated with these characterization parameters to construct a fingerprint database. This database contains pre-measured and recorded correspondences between signal features and locations. The most commonly used signal characteristic parameter is RSSI. Once the fingerprint database is established, the RF sensor stations collect real-time wireless signal parameters, which are then matched and compared with the fingerprint information in the database [107]. Accurate target location is estimated through the use of comparison and inference algorithms. Compared to the geometry-based method, the fingerprint localization method is more accurate. However, constructing the fingerprint database requires significant effort, and it must be re-collected and constructed if the signal receiver station's location changes [103].

2.6.2. WiFi

WiFi is a widely used wireless technology that follows the IEEE 802.11 standards. WiFi utilizes wireless communication frequencies of 2.4 GHz and 5 GHz. It can support multiple device connections simultaneously and is widely used in indoor settings such as factories and offices. While WiFi technology offers advantages such as convenience, flexibility, and high speed, it has limitations when used for indoor localization. Factors such as the volatility of the WiFi signal and the non-line-of-sight (NLOS) path may affect the accuracy of localization.

For WiFi-based positioning methods, RSSI is most commonly used because of its high accuracy. Hemin Ye et al. [108] optimized the traditional WiFi fingerprint positioning method. Compared to traditional methods for fingerprint localization, this approach reduces the distance between sampling points and improves fingerprint matching accuracy by collecting and normalizing WiFi signals from different time periods. Additionally, the method uses Mahalanobis distance as a similarity reference and filters out noise using an improved adaptive K-value WKNN algorithm to enhance location estimation accuracy. Zhang L. et al. [109] optimized WiFi-based RSSI localization with Deep Fuzzy Forest. In their work, a deep fuzzy random forest is used as a mapping function, which estimates the corresponding coordinates of fingerprint vectors from an offline database. The deep fuzzy forest can inherit the merits of decision trees and deep neural networks within an end-to-end trainable architecture. Guangbing Zhou et al. [110] combine WiFi, vision, and LIDAR for the indoor localization of mobile robots.The WiFi-based RSSI fingerprinting localization method was used for coarse area estimation.

2.6.3. Zigbee

Zigbee is a wireless communication technology based on the IEEE 802.15.4 standard. It uses a distributed network topology that supports multiple devices communicating with each other, with all terminators communicating directly with the coordinator. ZigBee has a low cost, low power consumption, and low data transfer rate compared to WiFi standards, as well as a longer latency time.

Zhang Lei et al. [111] conducted an experimental analysis of Zigbee location using RSSI technology for the influence of reference node layout and number on the location accuracy. The experimental results indicate that Zigbee reference nodes with triangular layouts achieve higher localization accuracy than those with rectangular and prototypical layouts. Anbalagan Loganathan et al. [97] combined Zigbee-based RSSI and odometry for the indoor localization of mobile robots. Their method fuses the results of Zigbee-based RSSI triangulation with odometer data through an adaptive filtering method to find the robot's position. Zhu Wang et al. [112] designed a Zigbee-based localization algorithm for mobile robots. The method corrects the RSSI ranging results by least squares and uses

triangulation to calculate the robot's position. The coordination nodes and mobile nodes of the ZigBee network are embodied in mobile robots, thus enhancing the mobility and flexibility of the network.

2.6.4. Bluetooth

Bluetooth is a standardized wireless communication technology based on the IEEE 802.15 standard. It is commonly used for short-range data transmission due to its low power consumption, low cost, high security, and ease of use. Bluetooth technology is widely used in various electronic products, automobiles, and medical devices. However, the accuracy of indoor positioning is affected by the non-line-of-sight (NLOS) path due to obstacles affecting Bluetooth signal propagation.

Aswin N. Raghavan et al. [113] proposed a method for the localization of mobile robots using Bluetooth. This method is based on RSSI for distance estimation and uses trilateration measurements for localization. The localization accuracy is within 1 m. Y. Yamami et al. [114] designed a Bluetooth-based AOA method for indoor mobile robot localization. The method makes use of Bluetooth 5.1 with the Constant Tone Extension (CTE) at the end of a packet, enabling the Bluetooth module to receive signals from multiple antennas simultaneously. In the study conducted by Katrina Weinmann et al. [115], a mobile robot controller employs Bluetooth as a sensor to measure the localization of the target object. The Bluetooth beacon is mounted on the target object, while the mobile robot is equipped with a Bluetooth antenna array. The mobile robot achieves target tracking by estimating the direction and distance of the target object. The direction is estimated using the AoA method, while the distance is estimated using RSSI. Astafiev Alexandr et al. [116] experimentally analyzed indoor localization based on a sensor network using Bluetooth Low Energy (BLE) beacons. The experiments demonstrate that the transmission of BLE signals can be affected by the presence of other radio signals in the localization area. The level of signal interference directly impacts BLE-based localization accuracy, resulting in greater errors and a lower accuracy. Additionally, obstacles such as people interfere with the RSSI measurement of BLE, further affecting positioning accuracy. To enhance the anti-interference capability of the BLE positioning system, it can be combined with other sensors, such as the IMU.

2.6.5. Ultra-Wideband (UWB)

UWB (Ultra-Wideband) technology is a wireless communication technology that uses short, non-continuous pulse signals to transmit data. It has a wide frequency bandwidth and offers advantages such as low power consumption, fast transmission rate, and strong anti-interference capability. Additionally, UWB is theoretically less affected by non-line-of-sight (NLOS) paths, making it highly accurate and stable for indoor localization.

Dongqing Shi et al. [117] designed a positioning system based on UWB sensors. To optimize localization results in the presence of nonlinearity and noise in UWB ranging, the system employs the gradient descent method and the least squares method. Jiajun Leng et al. [118] designed a two-stage UWB positioning algorithm for indoor mobile robots. First, the distance measured by the TDOA method was optimized by Gaussian filtering. Then, the final location coordinates were obtained by weighting the corrected range values with coordinates.

To reduce the influence of the LOS/NLOS environment on positioning accuracy, Peisen LI et al. [119] designed a positioning method combining the inertial navigation system (INS) with Ultra-Wideband (UWB). In this method, an interactive multiple model (IMM) algorithm is proposed to integrate the different distance error characteristics of LOS and NLOS states. The algorithm uses two Kalman filter models for different states and transforms them using the Markov chain. The filter results are then fused using weighting for position estimation. The localization method corrects the UWB's estimated position with the INS position estimation results, resulting in the final localization estimation. Tao Xu et al. [120] proposed a Weighted Adaptive Kalman Filter (WAKF) positioning method

based on UWB andodometry. To minimize the effect of LOS/NLOS scenarios on UWB positioning accuracy, a Kalman filter is used to integrate the odometry data. Additionally, the method uses power differences to distinguish between LOS and NLOS environments, and it adjusts the Kalman filter weights accordingly. Haoyu Zhou et al. [121] designed an online multi-robot SLAM system based on Lidar/UWB fusion. The system fuses distance measurements provided by UWB sensors with Lidar data provided by different mobile robots and constructs a globally consistent map based on the UWB coordinate system.

2.6.6. Radio Frequency Identification (RFID)

RFID (Radio Frequency Identification) technology uses radio waves for data transmission and identification. It typically consists of three components: a tag, a reader, and a data processing system. RFID tags are small devices that contain a chip and an antenna and are used for data storage and transmission. The reader communicates with the tag via radio and reads or writes the tag's information. The RFID data are processed and managed by the data processing system. RFID technology offers the benefits of low energy consumption and low system costs. However, it has a short signal transmission distance of about 1–5 m, which requires the presetting of numerous beacon sites for large-area localization. RFID is advantageous for short-range target tracking and identification.

Haibing Wu et al. [122] proposed a method for mobile robot navigation based on RFID technology to track targets. The method uses a particle filter to measure the real-time relative position between the mobile robot and the RFID-tagged object based on the RFID phase difference observation model. Jun Wang et al. [123] proposed a SLAM method for mobile robots based on high-frequency band RFID by using the particle smoother for landmark mapping and the particle filter for the self-localization of the mobile robot. F. Shamsfakhr et al. [124] proposed a mobile robot localization method based on passive UHF radio frequency identification technology. The method combines odometry and RSSI-based position estimation results to determine the robot pose by the linear least squares method.

2.7. Summary and Discussion

IMU sensors have a high output frequency and are extensively adopted in mobile robot localization and SLAM fields. As a proprioceptive sensor, it is not influenced by the environment, but it does not have a role in obstacle detection. To mitigate error accumulation, an IMU is often used in conjunction with encoders as a short-term odometer and is part of the sensor fusion method in mobile robot positioning and SLAM. Ultrasonic sensors are less accurate in distance detection than infrared sensors and LiDAR. Therefore, they are rarely used as primary sensors in mobile robot localization and SLAM in recent studies. However, they have significant advantages over other sensors in obstacle detection and are widely used in mobile robot obstacle avoidance. Infrared sensors, as optical sensors, are susceptible to other light sources or reflections in the environment. However, infrared sensors are very low cost and are often used as auxiliary sensors in mobile robots, responsible for obstacle or target distance detection. LiDAR and vision sensors are the most commonly used sensors in SLAM for mobile robots, and they can both be used for navigation in unknown environments. In contrast, LiDAR has a higher distance measurement accuracy, and vision sensors can collect richer information about the working scenes. However, the accuracy of LiDAR measurements can be affected by transparent or reflective objects in the environment. Vision sensors are advantageous for detecting obstacles or targets due to their excellent object recognition technology. They can efficiently filter out obstacle data, improving localization precision during robot navigation. Radio frequency (RF) sensors are widely adopted in mobile robot localization, and the localization accuracy will not be influenced by environmental factors such as low-visibility conditions. However, RF sensor-based localization requires additional anchor nodes with prior position information and cannot provide orientation information for mobile robots. A comparison of results of the relating sensors is demonstrated in Table 3.

Table 3. The characteristics of the sensors.

Sensors	Advantages	Disadvantages
IMU	High output frequency, free from environmental interference.	Initial conditions required, error accumulation, accelerometer is susceptible to vibration.
Ultrasonic sensor	Low cost, not influenced by transparent or reflective objects	long-range intensity attenuation, vulnerable to ambient noise.
Infrared sensor	Low cost.	Susceptible to environmental interference (light source, reflection).
LiDAR	High distance measurement accuracy.	Restricted in environments surrounded by transparent or reflective objects.
Vision-based sensor	Rich collection of information	Large computational volume, vulnerable to light.
Radio Frequency Sensor	Environment and obstacles have less effect on positioning.	Not suitable for unknown or unpredictable environments, unable to measure orientation.

3. Overview of Multi-Sensor Fusion Sensing Technologies

Various sensors can gather diverse forms of measurement data. Sensor fusion allows for a more comprehensive acquisition of sensory information, thereby enhancing localization efficiency in task completion. It is important to note that each sensor has its own superiorities, limitations, and potential scenarios. It is noted that sensor information fusion is able to enhance mobile robot positioning accuracy, obstacle recognition efficiency, and other characteristics by leveraging strengths and compensating for weaknesses. This approach can meet the requirements of diverse working environments. For instance, visual sensors provide a vast amount of image data that is valuable for recurrent closed-loop detection and optimization of the same scene in large-scale environments. This method enables the elimination of cumulative sensor errors. However, visual sensors in indoor environments are also limited by their image information. They suffer from long image processing times (generally 30 fps is required for the vision SLAM image in real-time), and environmental information is susceptible to occlusion. Therefore, improving visual sensor performance through other sensors is also a hot topic for mobile robot perception. As for multi-sensors, sensors that are often mixed with visual sensors are sonar [125,126], laser rangefinders [127,128], radio frequency identification (RFID) [128,129], inertial sensors [130,131], GPS [132,133]. However, in the case of GPS sensors, they are based on satellite systems to achieve global positioning. In indoor environments, objects are affected by buildings as well as other signals, causing a significant reduction in their positioning accuracy, implying that GPS may be inappropriate for mobile robot location positioning in indoor obscured environments [134,135].

Sensor data fusion algorithms mainly include improved algorithms based on Kalman filters, particle filters, and neural networks.

3.1. Multi-Sensor Fusion Algorithm Based on Kalman Filter

Kalman filtering methods are mainly used for fusing high-frequency and high-precision dynamic data and have been widely applied in multi-sensor fusion [51,136,137]. When assuming that the system state model and observation model are linear and follow a Gaussian distribution and that the noise is also Gaussian, the Kalman filtering method only requires the mean and variance of the noise to iteratively solve for the state. Kalman filter mainly includes prediction and update steps to realize real-time state estimation of the mobile robot from the previous frame to the current one, as shown in Figure 7. However, the traditional Kalman filter cannot meet the requirements of nonlinear problems. Therefore, researchers have developed improved versions of the Kalman filter, such as the Extended Kalman Filter method (EKF) and the Untraceable Kalman Filter method (UKF), to deal with nonlinear problems.

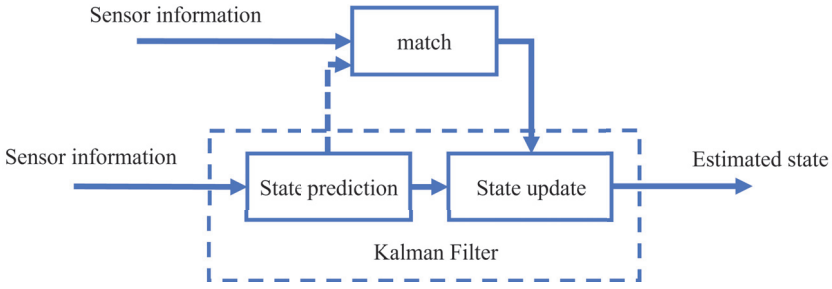


Figure 7. Multi-sensor fusion algorithm based on Kalman filter.

Daniel Magree et al. [138] proposed a modified navigation system by integrating visual SLAM and LiDAR SLAM, thus achieving an EKF-based inertial navigation system. In this vision SLAM approach, the robot's position can be determined by analyzing the environment's image features. The position is estimated and updated using the EKF method. LiDAR SLAM is based on the Monte Carlo method for scan matching. Its results are used to correct the results of visual SLAM and reduce the effect of ambiguous geometry on visual SLAM. Chengguo Zong et al. [139] presented an EKF multi-sensor information fusion method for obstacle avoidance in mobile robots. This method utilizes the previous moment's pose and encoder data to estimate the predicted position. The observed position can be obtained on the basis of observations from ultrasonic sensors, infrared sensors, and an electronic compass. The EKF updates the pose estimation by incorporating the predicted and observed positions, resulting in the optimal estimate of the required pose. LILI MU et al. [140] offered a SLAM approach that fuses LIDAR, an RGBD camera, an encoder, and an IMU. This method utilizes UKF to fuse the data from the four sensors for localization. In this method, the current position of the mobile robot is obtained from the IMU and wheel encoder, and it is input into the UKF for updating the rough position, which is used as the initial position for scanning matching by LIDAR and depth camera point clouds. This speeds up the matching speed of the point cloud and improves the matching accuracy. The optimized pose obtained by point cloud matching is sent to UKF as the measured value. Ping Jiang et al. [141] designed a rank Kalman filtering method for obtaining the robot motion trajectory by utilizing the IMU and LiDAR observations. The proposed method improves the localization accuracy of indoor mobile robots under nonlinear and non-Gaussian noise models. Lin et al. [142] proposed an iterative Kalman filtering scheme by using the error states. It utilizes the IMU information as an information carrier to discretize the continuous motion model and integrate the three sensors to generate more accurate trajectories. Wei Xu et al. [143] proposed a LiDAR–inertial odometry framework that contains a state estimation module and a mapping one. In this way, the IMU and LiDAR data are input into the iterative Kalman filter for optimal estimating position.

3.2. Multi-Sensor Fusion Algorithm Based on Particle Filter

The Kalman filter and its improved algorithms are only capable of handling Gaussian distributions. When dealing with arbitrary distributions, the adoption of Kalman filter-related methods may result in more errors. The particle filter-based approach, known as the Monte Carlo algorithm, is able to deal with the arbitrary distribution of multiple samples [144]. In this method, the probability density function of the indoor AMR's position can be approximated using a set of "samples" or "particles", and regions with a larger number of particles have a higher probability. Each particle denotes a hypothetical pose of the considered mobile robot, while its weight indicates the extent of the match between the hypothesis and the true state. Typically, larger samples are used for global localization, and smaller samples are used for pose tracking.

When using the Monte Carlo algorithm for global localization, particles must be distributed throughout the entire map due to the unknown initial position. However,

this can result in a significant amount of computation and reduced localization efficiency. To enhance global localization efficiency, researchers have conducted numerous studies on sensor fusion. This involves rough localization followed by accurate localization, as shown in Figure 8. Song Xu et al. [145] designed a global localization method using the camera and LiDAR. In this method, global localization includes two processes: coarse localization and fine localization. In coarse localization, existing features from a pre-trained convolutional neural network (CNN) are used for determining the candidate positions based on images extracted from a monocular camera. In fine localization, the precise robot position is estimated by adaptive Monte Carlo localization based on LiDAR, where the candidate position is used as a seed for initial random sampling. Gengyu Ge et al. [146] proposed a method for localizing mobile robots by fusing camera and LiDAR data. The method first obtains coarse localization from the camera's text-level semantic information, and then enhances the localization by using the Monte Carlo localization (MCL) method with LiDAR data. Huang Y et al. [147] presented a global localization approach using LiDAR and dual AprilTag for enhancing the global localization of mobile robots. This method utilizes AprilTag-based localization results as a rough localization for the adaptive Monte Carlo global localization algorithm. Based on this rough localization, precisely located particles are generated, which enhances the efficiency, as well as the success rate during global localization.

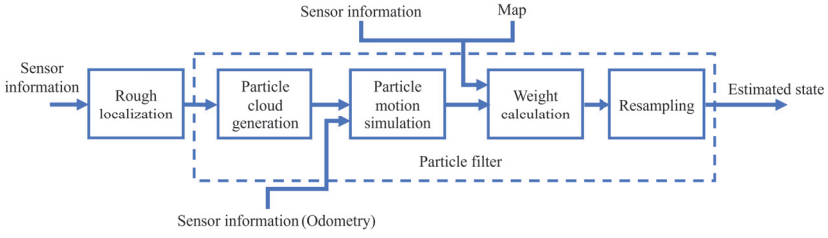


Figure 8. Multi-sensor fusion algorithm based on particle filter.

3.3. Multi-Sensor Fusion Algorithm Based on Neural Network

In recent years, researchers have introduced machine learning technology to mobile robots, which has led to the rapid development of mobile robotic systems [148,149]. Neural networks have also been widely studied and applied in multi-sensor fusion for mobile robots. Neural network-based methods can automatically extract features from data, avoiding the difficulty and uncertainty of manually designing features while having an improved generalization ability and robustness, as shown in Figure 9. Compared to Kalman filter-based methods, neural network-based methods are more capable of handling nonlinear problems and achieving higher accuracy.

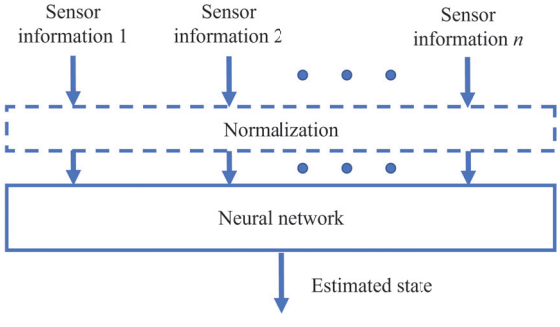


Figure 9. Multi-sensor fusion algorithm based on neural network.

Carlos Eduardo Magrin et al. [150] proposed a hierarchical sensor fusion (HSF) method with an artificial neural network (ANN) for mobile robot self-localization. This method uses a two-level multilayer perceptron (MLP) to fuse the normalized sensor data of the sonar octagon, digital compass, and wireless network signal strength measure. Li et al. [151] applied neural network fusion of LiDAR and an encoder for mobile robot localization. In this method, LiDAR data and odometer data obtained through an encoder are fed into a three-layer neural network for fusion to improve the mobile robot's positioning accuracy. Chi Li et al. [152] presented a deep learning-based approach to localizing a mobile robot using a 2D LiDAR and an IMU. The method utilizes a recursive convolutional neural network (RCNN) to fuse laser and inertial data and estimate the scan-to-scan attitude. The approach demonstrates greater robustness than traditional geometric-based methods in challenging situations, such as high angular speeds. Jieun Lee et al. [153] provided a deep neural network architecture (FusionLoc) for mobile robot relocalization with a camera and 2D LiDAR. This method contains two feature extraction modules, i.e., a multi-head self-attention (MHSA) module and a regression one. To be more specific, the feature extraction module extracts features from RGB images and LiDAR point clouds, respectively; the MHSA module performs feature fusion; and the regression module realizes position estimation. This approach is capable of achieving improved performance as compared with traditional end-to-end relocalization approaches that only acquire data from one sensor. A fingerprint-assisted localization scheme for mobile robots is explored, which fuses RSSI and magnetometer measurements [154]. However, the accuracy and stability of RSSI-based fingerprint localization methods can be affected by various environmental disturbances. The magnetic field strength can provide information about the level of disturbances. The method uses a fusion of magnetometer measurements and fingerprint data from RSSI for mobile robot localization through multilayer perceptron (MLP) feedforward neural networks. Andres J. Barreto-Cubero et al. [155] fused an ultrasonic sensor and a stereo camera with 2D LiDAR through the artificial neural network to improve obstacle detection in mobile robots by using the capability of ultrasonic sensors to detect glass and the ability to accurately detect objects in the 3D environment of the stereo camera.

3.4. Other Multi-Sensor Fusion Algorithms

The weighted average method is a direct and effective approach that simplifies the data fusion process during algorithm design and ensures real-time calculations. The core idea of the method is how to weight multiple sensors and how to determine the correction method. Wen et al. [156] combine an incremental encoder with a camera to calculate the encoder-based local position and globally update the position through the similarity of the camera to the environment. Jiang et al. [157] achieved the combination of feature point cloud maps and grid maps by establishing joint error co-visual between LiDAR and the monocular camera. Zheng et al. [81] divided the SLAM system into two subsystems, i.e., VIO and LIO. The coupled relationship between the two subsystems is achieved by sharing the point map data between the two to achieve accurate position estimation. However, because the weighted average method does not take into account the relationship between the front and rear states of the robot, its anti-disturbance ability is not excellent in the face of highly dynamic environments. Kosisochukwu Pal Nnoli et al. [43] fused infrared and ultrasonic sensors to detect obstacles. Their approach uses an error-filtering covariance and averaging algorithm to logically fuse distance measurements from a pair of infrared and ultrasonic sensors to track the proximity of environmental obstacles within a 180-degree range in front of a mobile robot.

Graph optimization algorithms in SLAM are also used as a framework for multi-sensor fusion. Ran Liu et al. [158] presented a map-building scheme for UWB, LiDAR, and odometry. In this method, a graph optimization framework with two optimization processes is designed to handle the fusion problem of UWB ranging information, odometry, and LiDAR information. The first optimization process takes the poses of the UWB nodes as vertices in the graph, and the edges of the UWB ranges and odometers are used as constraints, and roughly optimized robot trajectories are obtained. The second one integrates the edges

determined based on LiDAR ICP scan matching to obtain the accurate robot trajectory. Along this line, then, the resulted trajectory can be incorporated into the LiDAR data, yielding the occupancy map of the industrial scenes. The graph optimization process can better deal with the influence of nonlinear systems on the robot. However, the algorithm's high calculation complexity, caused by the need to traverse previous state variables, can make real-time updating difficult during robot navigation. The robot's state is generally optimized when loop closure and relocalization are triggered.

4. Discussions and Future Trends

4.1. Discussions

The perception system of an AMR offers information about the robot's association with the environment, which is essential for successful navigation. To ensure the indoor mobile robot completes the navigation task successfully, it is essential to reasonably select and optimize the sensors and algorithms used in the perception system based on the specific needs of the scenario.

IMU-based trajectory inference methods can only be used for short periods of time and can be used to assist other localization systems. For mobile robot localization, beacon-based positioning systems (including RF sensors, ultrasonic sensors, and infrared sensors) can provide stable and accurate absolute position information. However, they require regular installation and maintenance, and widespread deployment can lead to increased costs. LiDAR, or vision cameras, can be used for accurate and stable positioning. However, their localization relies on environmental information extracted by the sensors, making them susceptible to changes in the environment. For scenarios with frequent changes, beacon-based localization methods are more appropriate.

For the SLAM of a mobile robot, both LIDAR and vision cameras have their own advantages. Laser point cloud data require less computation, are easy to compute, are accurate in ranging, and are less susceptible to large changes in lighting. However, the environmental information collected by LiDAR is not sufficient, and the map constructed using 2D LiDAR only contains distance information about the environment. The rotating mechanism of the device is simple, which negatively impacts the stability of its internal structure over time. Vision SLAM provides a more detailed description of the environment, including texture information, which enhances accuracy and enables the recognition of differences between environments. While laser SLAM has limitations, vision SLAM, equipped with a camera, offers greater stability in terms of internal structure. Mobile robots with only a single camera as a sensor in indoor environments are limited by environmental conditions and lack robustness.

For obstacle detection, vision cameras are effective for recognizing obstacles as they provide rich environmental information. Ultrasonic sensors and LiDAR are precise in measuring the distance and position of obstacles.

4.2. Future Trends

Benefiting from computer science, artificial intelligence, and visual approaches, perception technology will develop rapidly, and the following aspects are worth emphasizing:

1: **Multi-source Information Fusion.** The study and development of mobile robots have extended their possible applications, along with increased demands for their usage. However, a single sensor may struggle to satisfy the navigation requirements of complex scenes due to its inherent limitations. Along this line, then, multi-sensor fusion becomes a crucial area of research and development for indoor AMR. Multi-sensor fusion can improve the accuracy, efficiency, and stability of localization, map building, and obstacle detection by providing more comprehensive environmental information to the perception system. Multi-sensor fusion provides accurate and comprehensive information to the decision-making system, enabling mobile robots to adapt better to complex environments and complete navigation tasks more efficiently and robustly.

2: **Flexible and efficient optimization strategies.** When mobile robots work in various complex and dynamic environments, there is a large amount of unknown model noise and roughness in the observation information obtained by sensors. To ensure the efficiency, stability, and accuracy of the AMR system, appropriate optimization strategies should be explored based on the specific environment and dynamic changes. In addition, there are many complex nonlinear operations in the optimization process. To improve computational efficiency when dealing with large-scale or multi-modal data, methods such as parallel computing and distributed computing can be adopted.

3: **Integration with neural networks.** Since neural networks are able to enhance the stability of mobile robot systems, the combination of relevant neural networks and mobile robots has attracted significant interest. Additionally, the self-learning capabilities of neural networks can address sensor interference and external environmental factors, thereby improving localization anti-interference ability. However, current neural network-based algorithms still encounter limitations for localizing in various indoor scenarios. When employing the same neural network algorithm, the mobile robot experiences significant errors in varying environments. Further research should explore the combination of additional sensors and the application of more complex neural network structures to meet the navigation requirements in diverse indoor environments. With the emergence of large-scale language models and advancements in chip technology, neural network-based approaches are expected to rapidly expand in the future.

Author Contributions: Y.L. completed all the writing work and main analysis work for the manuscript; Y.X. proposed the framework and ideas of the whole work and put forward some important guiding opinions; M.W. completed part of the literature review and summary work; S.W. provided some guidance for writing and analyses; T.X. put forward some constructive suggestions for the final work. Y.L. and Y.X. have the same equal contribution to the work. All authors have read and agreed to the published version of the manuscript.

Funding: The work was supported by National Natural Science Foundation of China under Grant 52275488 and 52105019, Key Research and Development Program of Hubei Province, China under Grant 2022BAA064 and PostdoctoralScience Foundation of China under Grant 2023M731192.

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

References

1. Qu, Y.; Yang, M.; Zhang, J.; Xie, W.; Qiang, B.; Chen, J. An Outline of Multi-Sensor Fusion Methods for Mobile Agents Indoor Navigation. *Sensors* **2021**, *21*, 1605. [CrossRef]
2. Bose, D.; Mohan, K.; Cs, M.; Yadav, M.; Saini, D.K. Review of Autonomous Campus and Tour Guiding Robots with Navigation Techniques. *Aust. J. Mech. Eng.* **2022**, *21*, 1580–1590. [CrossRef]
3. Alatise, M.B.; Hancke, G.P. A Review on Challenges of Autonomous Mobile Robot and Sensor Fusion Methods. *IEEE Access* **2020**, *8*, 39830–39846. [CrossRef]
4. Rubio, F.; Valero, F.; Llopis-Albert, C. A review of mobile robots: Concepts, methods, theoretical framework, and applications. *Int. J. Adv. Robot. Syst.* **2019**, *16*, 172988141983959. [CrossRef]
5. Kortenkamp, D. Perception for mobile robot navigation: A survey of the state of the art. In Proceedings of the InDual-Use Space Technology Transfer Conference and Exhibition, Houston, TX, USA, 1 May 1994; Volume 2.
6. Obeidat, H.; Shuaieb, W.; Obeidat, O.; Abd-Alhameed, R. A Review of Indoor Localization Techniques and Wireless Technologies. *Wirel. Pers. Commun.* **2021**, *119*, 289–327. [CrossRef]
7. Panigrahi, P.K.; Bisoy, S.K. Localization strategies for autonomous mobile robots: A review. *J. King Saud Univ. Comput. Inf. Sci.* **2022**, *34*, 6019–6039. [CrossRef]
8. Huang, J.; Junginger, S.; Liu, H.; Thurow, K. Indoor Positioning Systems of Mobile Robots: A Review. *Robotics* **2023**, *12*, 47. [CrossRef]
9. Kim Geok, T.; Zar Aung, K.; Sandar Aung, M.; Thu Soe, M.; Abdaziz, A.; Pao Liew, C.; Hossain, F.; Tso, C.P.; Yong, W.H. Review of Indoor Positioning: Radio Wave Technology. *Appl. Sci.* **2020**, *11*, 279. [CrossRef]
10. Han, X.; Li, S.; Wang, X.; Zhou, W. Semantic Mapping for Mobile Robots in Indoor Scenes: A Survey. *Information* **2021**, *12*, 92. [CrossRef]
11. Chen, W.; Zhou, C.; Shang, G.; Wang, X.; Li, Z.; Xu, C.; Hu, K. SLAM Overview: From Single Sensor to Heterogeneous Fusion. *Remote Sens.* **2022**, *14*, 6033. [CrossRef]

12. Chen, W.; Shang, G.; Ji, A.; Zhou, C.; Wang, X.; Xu, C.; Li, Z.; Hu, K. An Overview on Visual SLAM: From Tradition to Semantic. *Remote Sens.* **2022**, *14*, 3010. [CrossRef]
13. Chou, C.; Li, H.; Song, D. Encoder-Camera-Ground Penetrating Radar Sensor Fusion: Bimodal Calibration and Subsurface Mapping. *IEEE Trans. Robot.* **2021**, *37*, 67–81. [CrossRef]
14. Shi, H.; Shi, L.; Xu, M.; Hwang, K.S. End-to-End Navigation Strategy With Deep Reinforcement Learning for Mobile Robots. *IEEE Trans. Ind. Inform.* **2020**, *16*, 2393–2402. [CrossRef]
15. Zeng, H.; Song, X.; Jiang, S. Multi-Object Navigation Using Potential Target Position Policy Function. *IEEE Trans. Image Process.* **2023**, *32*, 2608–2619. [CrossRef] [PubMed]
16. Wang, Y.; Li, X.; Zhang, J.; Li, S.; Xu, Z.; Zhou, X. Review of wheeled mobile robot collision avoidance under unknown environment. *Sci. Prog.* **2021**, *104*, 003685042110377. [CrossRef]
17. Qin, H.; Shao, S.; Wang, T.; Yu, X.; Jiang, Y.; Cao, Z. Review of Autonomous Path Planning Algorithms for Mobile Robots. *Drones* **2023**, *7*, 211. [CrossRef]
18. Choi, B.S.; Lee, J.J. Mobile robot localization in indoor environment using RFID and sonar fusion system. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 October 2009; pp. 2039–2044. [CrossRef]
19. Ouyang, M.; Cao, Z.; Guan, P.; Li, Z.; Zhou, C.; Yu, J. Visual-Gyroscope-Wheel Odometry with Ground Plane Constraint for Indoor Robots in Dynamic Environment. *IEEE Sens. Lett.* **2021**, *5*, 1–4. [CrossRef]
20. Morar, A.; Moldoveanu, A.; Mocanu, I.; Moldoveanu, F.; Radoi, I.E.; Asavei, V.; Gradinaru, A.; Butean, A. A Comprehensive Survey of Indoor Localization Methods Based on Computer Vision. *Sensors* **2020**, *20*, 2641. [CrossRef]
21. Li, R.; Du, Z.; Zhao, Y.; Liu, S. Design and implementation of mobile robot ultrasonic localization system. In Proceedings of the 2016 Chinese Control and Decision Conference (CCDC), Yinchuan, China, 28–30 May 2016; pp. 5347–5352. [CrossRef]
22. Chan, T.H.; Hesse, H.; Ho, S.G. LiDAR-Based 3D SLAM for Indoor Mapping. In Proceedings of the 2021 7th International Conference on Control, Automation and Robotics (ICCAR), Singapore, 23–26 April 2021; pp. 285–289. [CrossRef]
23. Borodacz, K.; Szczepański, C.; Popowski, S. Review and selection of commercially available IMU for a short time inertial navigation. *Aircr. Eng. Aerosp. Technol.* **2022**, *94*, 45–59. [CrossRef]
24. Luo, L.; Feng, Z.; Hong-Wei, W.; Long, C. Crawler Robot Indoor Positioning Based on a Combination of Bluetooth and IMU. In Proceedings of the 2022 6th International Conference on Robotics, Control and Automation (ICRCA), Xiamen, China, 26–28 February 2022; pp. 34–39. [CrossRef]
25. Alatise, M.; Hancke, G. Pose Estimation of a Mobile Robot Based on Fusion of IMU Data and Vision Data Using an Extended Kalman Filter. *Sensors* **2017**, *17*, 2164. [CrossRef]
26. Zhang, S.; Tan, X.; Wu, Q. Self-Positioning for Mobile Robot Indoor Navigation Based on Wheel Odometry, Inertia Measurement Unit and Ultra Wideband. In Proceedings of the 2021 5th International Conference on Vision, Image and Signal Processing (ICVISP), Kuala Lumpur, Malaysia, 18–20 December 2021; pp. 105–110. [CrossRef]
27. Lee, S.J.; Lee, K.; Song, J.B. Development of advanced grid map building model based on sonar geometric reliability for indoor mobile robot localization. In Proceedings of the 2014 11th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), Kuala Lumpur, Malaysia, 12–15 November 2014; pp. 292–297. [CrossRef]
28. Liu, H.; Sun, F.; Fang, B.; Zhang, X. Robotic Room-Level Localization Using Multiple Sets of Sonar Measurements. *IEEE Trans. Instrum. Meas.* **2017**, *66*, 2–13. [CrossRef]
29. Liu, Y.; Fan, R.; Yu, B.; Bocus, M.J.; Liu, M.; Ni, H.; Fan, J.; Mao, S. Mobile Robot Localisation and Navigation Using LEGO NXT and Ultrasonic Sensor. In Proceedings of the 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), Kuala Lumpur, Malaysia, 12–15 December 2018; pp. 1088–1093. [CrossRef]
30. Shen, M.; Wang, Y.; Jiang, Y.; Ji, H.; Wang, B.; Huang, Z. A New Positioning Method Based on Multiple Ultrasonic Sensors for Autonomous Mobile Robot. *Sensors* **2019**, *20*, 17. [CrossRef]
31. Hsu, C.C.; Chen, H.C.; Wong, C.C.; Lai, C.Y. Omnidirectional Ultrasonic Localization for Mobile Robots. *Sens. Mater.* **2022**, *34*, 453. [CrossRef]
32. Yuan, S.; Guo, P.; Han, X.; Luan, F.; Zhang, F.; Liu, T.; Mao, H. DSmT-Based Ultrasonic Detection Model for Estimating Indoor Environment Contour. *IEEE Trans. Instrum. Meas.* **2020**, *69*, 4002–4014. [CrossRef]
33. Takai, H.; Miyake, M.; Okuda, K.; Tachibana, K. A simple obstacle arrangement detection algorithm for indoor mobile robots. In Proceedings of the 2010 2nd International Asia Conference on Informatics in Control, Automation and Robotics (CAR 2010), Wuhan, China, 6–7 March 2010; pp. 110–113. [CrossRef]
34. Jean, J.H.; Wang, J.L. Development of an indoor patrol robot based on ultrasonic and vision data fusion. In Proceedings of the 2013 IEEE International Conference on Mechatronics and Automation, Takamatsu, Japan, 4–7 August 2013; pp. 1234–1238. [CrossRef]
35. Grami, T.; Sghaier Tlili, A. Indoor Mobile Robot Localization based on a Particle Filter Approach. In Proceedings of the 2019 19th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA), Sousse, Tunisia, 24–26 March 2019; pp. 47–52. [CrossRef]
36. Derkach, M.; Matiuk, D.; Skarga-Bandurova, I. Obstacle Avoidance Algorithm for Small Autonomous Mobile Robot Equipped with Ultrasonic Sensors. In Proceedings of the 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), Kyiv, Ukraine, 14–18 May 2020; pp. 236–241. [CrossRef]
37. Brassart, E.; Pegard, C.; Mouaddib, M. Localization using infrared beacons. *Robotica* **2000**, *18*, 153–161. [CrossRef]

38. Krejsa, J.; Vechet, S. Infrared Beacons based Localization of Mobile Robot. *Electron. Electr. Eng.* **2012**, *117*, 17–22. [CrossRef]
39. Li, T.H.; Chang, S.J.; Tong, W. Fuzzy Target Tracking Control of Autonomous Mobile Robots by Using Infrared Sensors. *IEEE Trans. Fuzzy Syst.* **2004**, *12*, 491–501. [CrossRef]
40. Juang, J.; Yang, Y. Fuzzy Sensor Fusion and Curve Approximation for indoor Map Building. In Proceedings of the 2015 International Conference on Artificial Intelligence and Industrial Engineering, Phuket, Thailand, 26–27 July 2015. [CrossRef]
41. Oultiligh, A.; Ayad, H.; Pozna, C.; Mogan, G.; ELbouzekraoui, M.; Elkari, B. Obstacle Avoidance using Fuzzy Controller for Unicycle Robot. In Proceedings of the 2020 International Conference on Control, Automation and Diagnosis (ICCAD), Paris, France, 7–9 October 2020; pp. 1–6. [CrossRef]
42. Solano, D.M.; Grande, R.E.; Bonilla, M.N.I. PID Control and Fuzzy Logic System to the Obstacle Avoidance in an Autonomous Robot. In Proceedings of the 2021 18th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), Mexico City, Mexico, 10–12 November 2021; pp. 1–6. [CrossRef]
43. Nnoli, K.P.; Benyeogor, M.S.; Bolu, J.I.; Olakanmi, O.O. Edge-Based Infrared-Ultrasonic Anti-Collision Radar System for Robotic Navigation: *Applications of Cost-effective Bisensory System for Obstacle Detection, Tracking, and Avoidance. In Proceedings of the 2022 IEEE International Symposium on Technologies for Homeland Security (HST), Boston, MA, USA, 14–15 November 2022; pp. 1–7. [CrossRef]
44. Zhao, J.; Fang, J.; Wang, S.; Wang, K.; Liu, C.; Han, T. Obstacle Avoidance of Multi-Sensor Intelligent Robot Based on Road Sign Detection. *Sensors* **2021**, *21*, 6777. [CrossRef]
45. Habich, T.L.; Stuede, M.; Labbe, M.; Spindeldreier, S. Have I been here before? Learning to Close the Loop with LiDAR Data in Graph-Based SLAM. In Proceedings of the 2021 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Delft, The Netherlands, 12–16 July 2021; pp. 504–510. [CrossRef]
46. Meng, J.; Wan, L.; Wang, S.; Jiang, L.; Li, G.; Wu, L.; Xie, Y. Efficient and Reliable LiDAR-Based Global Localization of Mobile Robots Using Multiscale/Resolution Maps. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–15. [CrossRef]
47. Hähnel, D.; Burgard, W.; Thrun, S. Learning compact 3D models of indoor and outdoor environments with a mobile robot. *Robot. Auton. Syst.* **2003**, *44*, 15–27. [CrossRef]
48. Wang, X.; Marcotte, R.J.; Olson, E. GLFP: Global Localization from a Floor Plan. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 1627–1632. [CrossRef]
49. Kuang, H.; Chen, X.; Guadagnino, T.; Zimmerman, N.; Behley, J.; Stachniss, C. IR-MCL: Implicit Representation-Based Online Global Localization. *IEEE Robot. Autom. Lett.* **2023**, *8*, 1627–1634. [CrossRef]
50. Kim, J.; Chung, W. Localization of a Mobile Robot Using a Laser Range Finder in a Glass-Walled Environment. *IEEE Trans. Ind. Electron.* **2016**, *63*, 3616–3627. [CrossRef]
51. Grisetti, G.; Stachniss, C.; Burgard, W. Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters. *IEEE Trans. Robot.* **2007**, *23*, 34–46. [CrossRef]
52. Kohlbrecher, S.; Von Stryk, O.; Meyer, J.; Klingauf, U. A flexible and scalable SLAM system with full 3D motion estimation. In Proceedings of the 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics, Kyoto, Japan, 1–5 November 2011; pp. 155–160. [CrossRef]
53. Konolige, K.; Grisetti, G.; Kümmerle, R.; Burgard, W.; Limketkai, B.; Vincent, R. Efficient Sparse Pose Adjustment for 2D mapping. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 18–22 October 2010; pp. 22–29. [CrossRef]
54. Hess, W.; Kohler, D.; Rapp, H.; Andor, D. Real-time loop closure in 2D LIDAR SLAM. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 1271–1278. [CrossRef]
55. Xiang, Y.; Yang, X.Q.; Yang, W.W.; Miao, W.H. Localization and Mapping Algorithm for the Indoor Mobile Robot Based on LIDAR. *IOP Conf. Ser. Mater. Sci. Eng.* **2020**, *831*, 012021. [CrossRef]
56. Wang, H.; Wang, C.; Xie, L. Intensity-SLAM: Intensity Assisted Localization and Mapping for Large Scale Environment. *IEEE Robot. Autom. Lett.* **2021**, *6*, 1715–1721. [CrossRef]
57. Tian, Y.; Liu, X.; Li, L.; Wang, W. Intensity-Assisted ICP for Fast Registration of 2D-LIDAR. *Sensors* **2019**, *19*, 2124. [CrossRef]
58. Hu, W.; Zhang, K.; Shao, L.; Lin, Q.; Hua, Y.; Qin, J. Clustering Denoising of 2D LiDAR Scanning in Indoor Environment Based on Keyframe Extraction. *Sensors* **2022**, *23*, 18. [CrossRef]
59. Jiang, S.; Wang, S.; Yi, Z.; Zhang, M.; Lv, X. Autonomous Navigation System of Greenhouse Mobile Robot Based on 3D Lidar and 2D Lidar SLAM. *Front. Plant Sci.* **2022**, *13*, 815218. [CrossRef]
60. Guan, W.; Huang, L.; Wen, S.; Yan, Z.; Liang, W.; Yang, C.; Liu, Z. Robot Localization and Navigation Using Visible Light Positioning and SLAM Fusion. *J. Light. Technol.* **2021**, *39*, 7040–7051. [CrossRef]
61. Lee, J.; Tsubouchi, T.; Yamamoto, K.; Egawa, S. People Tracking Using a Robot in Motion with Laser Range Finder. In Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–15 October 2006; pp. 2936–2942. [CrossRef]
62. Mozos, O.M.; Kurazume, R.; Hasegawa, T. Multi-Part People Detection Using 2D Range Data. *Int. J. Soc. Robot.* **2010**, *2*, 31–40. [CrossRef]
63. Guerrero-Higueras, Á.M.; Álvarez Aparicio, C.; Calvo Olivera, M.C.; Rodríguez-Lera, F.J.; Fernández-Llamas, C.; Rico, F.M.; Matellán, V. Tracking People in a Mobile Robot From 2D LIDAR Scans Using Full Convolutional Neural Networks for Security in Cluttered Environments. *Front. Neurorobot.* **2019**, *12*, 85. [CrossRef]

64. Yan, Z.; Duckett, T.; Bellotto, N. Online learning for 3D LiDAR-based human detection: Experimental analysis of point cloud clustering and classification methods. *Auton. Robot.* **2020**, *44*, 147–164. [CrossRef]
65. Li, Z.; Wang, C.; Yan, W.; Ji, Y.; Zou, A.; Lai, J. Research on obstacle detection and location of indoor robot based on LIDAR. In Proceedings of the 9th International Symposium on Advanced Optical Manufacturing and Testing Technologies: Optical Test, Measurement Technology, and Equipment, Chengdu, China, 26–29 June 2019; p. 31. [CrossRef]
66. Henry, P.; Krainin, M.; Herbst, E.; Ren, X.; Fox, D. RGB-D Mapping: Using Depth Cameras for Dense 3D Modeling of Indoor Environments. In *Experimental Robotics*; Khatib, O., Kumar, V., Sukhatme, G., Eds.; Springer: Berlin/Heidelberg, Germany, 2014; Volume 79, pp. 477–491. [CrossRef]
67. Wimbauer, F.; Yang, N.; Von Stumberg, L.; Zeller, N.; Cremers, D. MonoRec: Semi-Supervised Dense Reconstruction in Dynamic Environments from a Single Moving Camera. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 6108–6118. [CrossRef]
68. Campos, C.; Elvira, R.; Rodriguez, J.J.G.; Montiel, J.M.M.; Tardos, J.D. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [CrossRef]
69. Li, Y.; Liu, X.; Dong, W.; Zhou, H.; Bao, H.; Zhang, G.; Zhang, Y.; Cui, Z. DELTAR: Depth Estimation from a Light-weight ToF Sensor and RGB Image. *arXiv* **2022**, arXiv:2209.13362.
70. Davison, A.J.; Reid, I.D.; Molton, N.D.; Stasse, O. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1052–1067. [CrossRef] [PubMed]
71. Konolige, K.; Agrawal, M. FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping. *IEEE Trans. Robot.* **2008**, *24*, 1066–1077. [CrossRef]
72. Newcombe, R.A.; Fitzgibbon, A.; Izadi, S.; Hilliges, O.; Molyneaux, D.; Kim, D.; Davison, A.J.; Kohi, P.; Shotton, J.; Hodges, S. KinectFusion: Real-time dense surface mapping and tracking. In Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality, Basel, Switzerland, 26–29 October 2011; pp. 127–136. [CrossRef]
73. Rebecq, H.; Horstschaefer, T.; Gallego, G.; Scaramuzza, D. EVO: A Geometric Approach to Event-Based 6-DOF Parallel Tracking and Mapping in Real Time. *IEEE Robot. Autom. Lett.* **2017**, *2*, 593–600. [CrossRef]
74. Bajpayee, A.; Techet, A.H.; Singh, H. Real-Time Light Field Processing for Autonomous Robotics. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 4218–4225. [CrossRef]
75. Liu, L.; Song, X.; Wang, M.; Liu, Y.; Zhang, L. Self-supervised Monocular Depth Estimation for All Day Images using Domain Separation. *arXiv* **2021**, arXiv:2108.07628.
76. Bescos, B.; Cadena, C.; Neira, J. Empty Cities: A Dynamic-Object-Invariant Space for Visual SLAM. *IEEE Trans. Robot.* **2021**, *37*, 433–451. [CrossRef]
77. Tourani, A.; Bavle, H.; Sanchez-Lopez, J.L.; Voos, H. Visual SLAM: What Are the Current Trends and What to Expect? *Sensors* **2022**, *22*, 9297. [CrossRef]
78. Zhan, H.; Weerasekera, C.S.; Bian, J.W.; Reid, I. Visual Odometry Revisited: What Should Be Learnt? In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 4203–4210. [CrossRef]
79. Zhou, Y.; Gallego, G.; Shen, S. Event-Based Stereo Visual Odometry. *IEEE Trans. Robot.* **2021**, *37*, 1433–1450. [CrossRef]
80. Zhu, Z.; Peng, S.; Larsson, V.; Xu, W.; Bao, H.; Cui, Z.; Oswald, M.R.; Pollefeys, M. NICE-SLAM: Neural Implicit Scalable Encoding for SLAM. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 12776–12786. [CrossRef]
81. Zheng, C.; Zhu, Q.; Xu, W.; Liu, X.; Guo, Q.; Zhang, F. FAST-LIVO: Fast and Tightly-coupled Sparse-Direct LiDAR-Inertial-Visual Odometry. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2022; pp. 4003–4009. [CrossRef]
82. Wu, M.; Wang, S.; Meng, J.; Xie, Y. AprilTag-Aided Pose Estimation Optimization of Landmark VSLAM. In Proceedings of the 2023 42nd Chinese Control Conference (CCC), Tianjin, China, 24–26 July 2023; pp. 4556–4561. [CrossRef]
83. Schreiber, M.; Belagiannis, V.; Glaser, C.; Dietmayer, K. Dynamic Occupancy Grid Mapping with Recurrent Neural Networks. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 6717–6724. [CrossRef]
84. Eisoldt, M.; Flottmann, M.; Gaal, J.; Buschermohle, P.; Hinderink, S.; Hillmann, M.; Nitschmann, A.; Hoffmann, P.; Wiemann, T.; Porrmann, M. HATSDF SLAM—Hardware-accelerated TSDF SLAM for Reconfigurable SoCs. In Proceedings of the 2021 European Conference on Mobile Robots (ECMR), Bonn, Germany, 31 August–3 September 2021; pp. 1–7. [CrossRef]
85. Rosinol, A.; Abate, M.; Chang, Y.; Carlone, L. Kimera: An Open-Source Library for Real-Time Metric-Semantic Localization and Mapping. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 1689–1696. [CrossRef]
86. Vijayanarasimhan, S.; Ricco, S.; Schmid, C.; Sukthankar, R.; Fragkiadaki, K. SfM-Net: Learning of Structure and Motion from Video. *arXiv* **2017**, arXiv:1704.07804.
87. Cao, A.Q.; De Charette, R. MonoScene: Monocular 3D Semantic Scene Completion. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 3981–3991. [CrossRef]

88. Kim, H.; Moon, J.; Lee, B. RGB-to-TSDF: Direct TSDF Prediction from a Single RGB Image for Dense 3D Reconstruction. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 6714–6720. [CrossRef]
89. Yu, C.; Liu, Z.; Liu, X.J.; Xie, F.; Yang, Y.; Wei, Q.; Fei, Q. DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1168–1174. [CrossRef]
90. Kulhanek, J.; Derner, E.; De Bruin, T.; Babuska, R. Vision-based Navigation Using Deep Reinforcement Learning. In Proceedings of the 2019 European Conference on Mobile Robots (ECMR), Prague, Czech Republic, 4–6 September 2019; pp. 1–8. [CrossRef]
91. Xiao, W.; Yuan, L.; He, L.; Ran, T.; Zhang, J.; Cui, J. Multigoal Visual Navigation With Collision Avoidance via Deep Reinforcement Learning. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 1–9. [CrossRef]
92. Xie, L.; Markham, A.; Trigoni, N. SnapNav: Learning Mapless Visual Navigation with Sparse Directional Guidance and Visual Reference. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 1682–1688. [CrossRef]
93. Tai, L.; Li, S.; Liu, M. A deep-network solution towards model-less obstacle avoidance. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Republic of Korea, 9–14 October 2016; pp. 2759–2764. [CrossRef]
94. De Cristóforis, P.; Nitsche, M.; Krajník, T.; Pire, T.; Mejail, M. Hybrid vision-based navigation for mobile robots in mixed indoor/outdoor environments. *Pattern Recognit. Lett.* **2015**, *53*, 118–128. [CrossRef]
95. Biswas, J.; Veloso, M.M. WiFi Localization and Navigation for Autonomous Indoor Mobile Robots. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010; pp. 4379–4384. [CrossRef]
96. Schwarz, D.; Schwarz, M.; Stückler, J.; Behnke, S. Cosero, Find My Keys! Object Localization and Retrieval Using Bluetooth Low Energy Tags. In *RoboCup 2014: Robot World Cup XVIII*; Bianchi, R.A.C., Akin, H.L., Ramamoorthy, S., Sugiura, K., Eds.; Series Title: Lecture Notes in Computer Science; Springer International Publishing: Cham, Germany, 2015; Volume 8992, pp. 195–206. [CrossRef]
97. Loganathan, A.; Ahmad, N.S. Self-Adaptive Filtering Approach for Improved Indoor Localization of a Mobile Node with Zigbee-Based RSSI and Odometry. *Sensors* **2019**, *19*, 4748. [CrossRef]
98. Xin, J.; Xie, G.; Yan, B.; Shan, M.; Li, P.; Gao, K. Multimobile Robot Cooperative Localization Using Ultrawideband Sensor and GPU Acceleration. *IEEE Trans. Autom. Sci. Eng.* **2022**, *19*, 2699–2710. [CrossRef]
99. Yang, B.; Yang, E. A Survey on Radio Frequency based Precise Localisation Technology for UAV in GPS-denied Environment. *J. Intell. Robot. Syst.* **2021**, *103*, 38. [CrossRef]
100. Shen, J.; Junyang, S.; Molisch, A.F.; Salmi, J. Accurate Passive Location Estimation Using TOA Measurements. *IEEE Trans. Wirel. Commun.* **2012**, *11*, 2182–2192. [CrossRef]
101. Guvenc, I.; Chong, C.C. A Survey on TOA Based Wireless Localization and NLOS Mitigation Techniques. *IEEE Commun. Surv. Tutorials* **2009**, *11*, 107–124. [CrossRef]
102. Ho, K.C. Bias Reduction for an Explicit Solution of Source Localization Using TDOA. *IEEE Trans. Signal Process.* **2012**, *60*, 2101–2114. [CrossRef]
103. Zafari, F.; Gkelias, A.; Leung, K.K. A Survey of Indoor Localization Systems and Technologies. *IEEE Commun. Surv. Tutorials* **2019**, *21*, 2568–2599. [CrossRef]
104. Wang, Y.; Ho, K.C. An Asymptotically Efficient Estimator in Closed-Form for 3-D AOA Localization Using a Sensor Network. *IEEE Trans. Wirel. Commun.* **2015**, *14*, 6524–6535. [CrossRef]
105. Mazuelas, S.; Bahillo, A.; Lorenzo, R.M.; Fernandez, P.; Lago, F.A.; Garcia, E.; Blas, J.; Abril, E.J. Robust Indoor Positioning Provided by Real-Time RSSI Values in Unmodified WLAN Networks. *IEEE J. Sel. Top. Signal Process.* **2009**, *3*, 821–831. [CrossRef]
106. He, S.; Chan, S.H.G. Wi-Fi Fingerprint-Based Indoor Positioning: Recent Advances and Comparisons. *IEEE Commun. Surv. Tutorials* **2016**, *18*, 466–490. [CrossRef]
107. Davidson, P.; Piche, R. A Survey of Selected Indoor Positioning Methods for Smartphones. *IEEE Commun. Surv. Tutorials* **2017**, *19*, 1347–1370. [CrossRef]
108. Ye, H.; Peng, J. Robot Indoor Positioning and Navigation Based on Improved WiFi Location Fingerprint Positioning Algorithm. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 8274455. [CrossRef]
109. Zhang, L.; Chen, Z.; Cui, W.; Li, B.; Chen, C.; Cao, Z.; Gao, K. WiFi-Based Indoor Robot Positioning Using Deep Fuzzy Forests. *IEEE Internet Things J.* **2020**, *7*, 10773–10781. [CrossRef]
110. Zhou, G.; Xu, S.; Zhang, S.; Wang, Y.; Xiang, C. Multi-Floor Indoor Localization Based on Multi-Modal Sensors. *Sensors* **2022**, *22*, 4162. [CrossRef] [PubMed]
111. Zhang, L.; Zhang, S.; Leng, C.T. A Study on the Location System Based on Zigbee for Mobile Robot. *Appl. Mech. Mater.* **2014**, *651–653*, 612–615. [CrossRef]
112. Wang, Z.; Liu, M.; Zhang, Y. Mobile localization in complex indoor environment based on ZigBee wireless network. *J. Phys.* **2019**, *1314*, 012214. [CrossRef]
113. Raghavan, A.N.; Ananthapadmanaban, H.; Sivamurugan, M.S.; Ravindran, B. Accurate mobile robot localization in indoor environments using bluetooth. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010; pp. 4391–4396. [CrossRef]

114. Yamami, Y.; Tang, S. AoA Estimation for High Accuracy BLE Positioning. In Proceedings of the IEEE Consumer Communications and Networking Conference, Las Vegas, NV, USA, 8–11 January 2023. [CrossRef]
115. Weinmann, K.; Simske, S. Design of Bluetooth 5.1 Angle of Arrival Homing Controller for Autonomous Mobile Robot. *Robotics* **2023**, *12*, 115. [CrossRef]
116. Alexandr, A.; Anton, D.; Mikhail, M.; Ilya, K. Comparative Analysis of Indoor Positioning Methods Based on the Wireless Sensor Network of Bluetooth Low Energy Beacons. In Proceedings of the 2020 International Conference Engineering and Telecommunication (En &T), Dolgoprudny, Russia, 25–26 November 2020. [CrossRef]
117. Shi, D.; Mi, H.; Collins, E.G.; Wu, J. An Indoor Low-Cost and High-Accuracy Localization Approach for AGVs. *IEEE Access* **2020**, *8*, 50085–50090. [CrossRef]
118. Leng, J.; Ma, G.; Zhu, J.; Ma, H. Improved TDOA Two-Stage UWB Localization Algorithm For Indoor Mobile Robot. In Proceedings of the IEEE International Conference on Recent Advances in Systems Science and Engineering (IEEE RASSE 2021), Shanghai, China, 12–14 December 2021. [CrossRef]
119. Li, P.; Xu, Y.; Shen, T.; Bi, S. INS/UWB integrated AGV localization employing Kalman filter for indoor LOS/NLOS mixed environment. In Proceedings of the 2019 International Conference on Advanced Mechatronic Systems (ICAMechS), Kusatsu, Shiga, Japan, 26–28 August 2019; pp. 294–298. [CrossRef]
120. Xu, T.; Zhang, H.; Zhou, X.; Yuan, X.; Tan, X.; Zhang, J.; Zhong, H. A Weight Adaptive Kalman Filter Localization Method Based on UWB and Odometry. In Proceedings of the 2022 International Conference on Advanced Robotics and Mechatronics (ICARM), Guilin, China, 9–11 July 2022; pp. 956–961. [CrossRef]
121. Zhou, H.; Yao, Z.; Zhang, Z.; Liu, P.; Lu, M. An Online Multi-Robot SLAM System Based on Lidar/UWB Fusion. *IEEE Sens. J.* **2022**, *22*, 2530–2542. [CrossRef]
122. Wu, H.; Tao, B.; Gong, Z.; Yin, Z.; Ding, H. A Standalone RFID-Based Mobile Robot Navigation Method Using Single Passive Tag. *IEEE Trans. Autom. Sci. Eng.* **2021**, *18*, 1529–1537. [CrossRef]
123. Wang, J.; Takahashi, Y. Particle Smoother-Based Landmark Mapping for the SLAM Method of an Indoor Mobile Robot with a Non-Gaussian Detection Model. *J. Sens.* **2019**, *2019*, 3717298. [CrossRef]
124. Shamsfakhr, F.; Macii, D.; Fontanelli, D.; Motroni, A.; Nepa, P.; Palopoli, L.; Buffi, A. RFID-based robot localisation: An unconstrained optimisation problem by exploiting RSSI. In Proceedings of the 2022 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Ottawa, ON, Canada, 16–19 May 2022; pp. 1–6. [CrossRef]
125. Wijk, O.; Christensen, H. Localization and navigation of a mobile robot using natural point landmarks extracted from sonar data. *Robot. Auton. Syst.* **2000**, *31*, 31–42. [CrossRef]
126. Lee, J.S.; Nam, S.Y.; Chung, W.K. Robust RBPF-SLAM for Indoor Mobile Robots Using Sonar Sensors in Non-Static Environments. *Adv. Robot.* **2011**, *25*, 1227–1248. [CrossRef]
127. Arras, K.; Tomatis, N. Improving robustness and precision in mobile robot localization by using laser range finding and monocular vision. In Proceedings of the 1999 Third European Workshop on Advanced Mobile Robots (Eurobot'99). Proceedings (Cat. No.99EX355), Zurich, Switzerland, 6–8 September 1999; pp. 177–185. [CrossRef]
128. Lingemann, K.; Nüchter, A.; Hertzberg, J.; Surmann, H. High-speed laser localization for mobile robots. *Robot. Auton. Syst.* **2005**, *51*, 275–296. [CrossRef]
129. Choi, B.S.; Lee, J.W.; Lee, J.J.; Park, K.T. A Hierarchical Algorithm for Indoor Mobile Robot Localization Using RFID Sensor Fusion. *IEEE Trans. Ind. Electron.* **2011**, *58*, 2226–2235. [CrossRef]
130. Cho, B.S.; Moon, W.s.; Seo, W.J.; Baek, K.R. A dead reckoning localization system for mobile robots using inertial sensors and wheel revolution encoding. *J. Mech. Sci. Technol.* **2011**, *25*, 2907–2917. [CrossRef]
131. Myung, H.; Lee, H.K.; Choi, K.; Bang, S. Mobile robot localization with gyroscope and constrained Kalman filter. *Int. J. Control Autom. Syst.* **2010**, *8*, 667–676. [CrossRef]
132. Zhang, X.; Xian, B.; Zhao, B.; Zhang, Y. Autonomous Flight Control of a Nano Quadrotor Helicopter in a GPS-Denied Environment Using On-Board Vision. *IEEE Trans. Ind. Electron.* **2015**, *62*, 6392–6403. [CrossRef]
133. López, E.; García, S.; Barea, R.; Bergasa, L.; Molinos, E.; Arroyo, R.; Romera, E.; Pardo, S. A Multi-Sensorial Simultaneous Localization and Mapping (SLAM) System for Low-Cost Micro Aerial Vehicles in GPS-Denied Environments. *Sensors* **2017**, *17*, 802. [CrossRef]
134. Bachrach, A.; Prentice, S.; He, R.; Henry, P.; Huang, A.S.; Krainin, M.; Maturana, D.; Fox, D.; Roy, N. Estimation, planning, and mapping for autonomous flight using an RGB-D camera in GPS-denied environments. *Int. J. Robot. Res.* **2012**, *31*, 1320–1343. [CrossRef]
135. Chowdhary, G.; Johnson, E.N.; Magree, D.; Wu, A.; Shein, A. GPS-denied Indoor and Outdoor Monocular Vision Aided Navigation and Control of Unmanned Aircraft: J. Field Robotics. *J. Field Robot.* **2013**, *30*, 415–438. [CrossRef]
136. Yang, Z.; Shen, S. Monocular Visual–Inertial State Estimation With Online Initialization and Camera–IMU Extrinsic Calibration. *IEEE Trans. Autom. Sci. Eng.* **2017**, *14*, 39–51. [CrossRef]
137. Huh, S.; Shim, D.H.; Kim, J. Integrated navigation system using camera and gimbaled laser scanner for indoor and outdoor autonomous flight of UAVs. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 3158–3163. [CrossRef]
138. Magree, D.; Johnson, E.N. Combined laser and vision-aided inertial navigation for an indoor unmanned aerial vehicle. In Proceedings of the 2014 American Control Conference, Portland, OR, USA, 4–6 June 2014; pp. 1900–1905. [CrossRef]

139. Zong, C.; Ji, Z.; Yu, Y.; Shi, H. Research on Obstacle Avoidance Method for Mobile Robot Based on Multisensor Information Fusion. *Sens. Mater.* **2020**, *32*, 1159. [CrossRef]
140. Mu, L.; Yao, P.; Zheng, Y.; Chen, K.; Wang, F.; Qi, N. Research on SLAM Algorithm of Mobile Robot Based on the Fusion of 2D LiDAR and Depth Camera. *IEEE Access* **2020**, *8*, 157628–157642. [CrossRef]
141. Jiang, P.; Chen, L.; Guo, H.; Yu, M.; Xiong, J. Novel indoor positioning algorithm based on Lidar/inertial measurement unit integrated system. *Int. J. Adv. Robot. Syst.* **2021**, *18*, 172988142199992. [CrossRef]
142. Lin, J.; Zheng, C.; Xu, W.; Zhang, F. R2LIVE: A Robust, Real-time, LiDAR-Inertial-Visual tightly-coupled state Estimator and mapping. *arXiv* **2021**, arXiv:2102.12400.
143. Xu, W.; Cai, Y.; He, D.; Lin, J.; Zhang, F. FAST-LIO2: Fast Direct LiDAR-Inertial Odometry. *IEEE Trans. Robot.* **2022**, *38*, 2053–2073. [CrossRef]
144. Alhamdi, E.; Hedjar, R. Comparative Study of Two Localization Approaches for Mobile Robots in an Indoor Environment. *J. Robot.* **2022**, *2022*, 1999082. [CrossRef]
145. Xu, S.; Chou, W.; Dong, H. A Robust Indoor Localization System Integrating Visual Localization Aided by CNN-Based Image Retrieval with Monte Carlo Localization. *Sensors* **2019**, *19*, 249. [CrossRef]
146. Ge, G.; Zhang, Y.; Wang, W.; Jiang, Q.; Hu, L.; Wang, Y. Text-MCL: Autonomous Mobile Robot Localization in Similar Environment Using Text-Level Semantic Information. *Machines* **2022**, *10*, 169. [CrossRef]
147. Huang, Y.H.; Lin, C.T. Indoor Localization Method for a Mobile Robot Using LiDAR and a Dual AprilTag. *Electronics* **2023**, *12*, 1023. [CrossRef]
148. Kendall, A.; Grimes, M.; Cipolla, R. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 2938–2946. [CrossRef]
149. Chen, C.; Wang, B.; Lu, C.X.; Trigoni, N.; Markham, A. A Survey on Deep Learning for Localization and Mapping: Towards the Age of Spatial Machine Intelligence. *arXiv* **2020**, arXiv:2006.12567.
150. Magrin, C.E.; Todt, E. Multi-Sensor Fusion Method Based on Artificial Neural Network for Mobile Robot Self-Localization. In Proceedings of the 2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE), Rio Grande, Brazil, 23–25 October 2019; pp. 138–143. [CrossRef]
151. Li, H.; Mao, Y.; You, W.; Ye, B.; Zhou, X. A neural network approach to indoor mobile robot localization. In Proceedings of the 2020 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES), Xuzhou, China, 16–19 October 2020; pp. 66–69. [CrossRef]
152. Li, C.; Wang, S.; Zhuang, Y.; Yan, F. Deep Sensor Fusion Between 2D Laser Scanner and IMU for Mobile Robot Localization. *IEEE Sens. J.* **2021**, *21*, 8501–8509. [CrossRef]
153. Lee, J.; Lee, H.; Oh, J. FusionLoc: Camera-2D LiDAR Fusion Using Multi-Head Self-Attention for End-to-End Serving Robot Relocalization. *IEEE Access* **2023**, *11*, 75121–75133. [CrossRef]
154. Sarcevic, P.; Csik, D.; Odry, A. Indoor 2D Positioning Method for Mobile Robots Based on the Fusion of RSSI and Magnetometer Fingerprints. *Sensors* **2023**, *23*, 1855. [CrossRef]
155. Barreto-Cubero, A.J.; Gómez-Espinosa, A.; Escobedo Cabello, J.A.; Cuan-Urquizo, E.; Cruz-Ramírez, S.R. Sensor Data Fusion for a Mobile Robot Using Neural Networks. *Sensors* **2021**, *22*, 305. [CrossRef]
156. Lv, W.; Kang, Y.; Qin, J. FVO: Floor vision aided odometry. *Sci. China Inf. Sci.* **2019**, *62*, 12202. [CrossRef]
157. Jiang, G.; Yin, L.; Jin, S.; Tian, C.; Ma, X.; Ou, Y. A Simultaneous Localization and Mapping (SLAM) Framework for 2.5D Map Building Based on Low-Cost LiDAR and Vision Fusion. *Appl. Sci.* **2019**, *9*, 2105. [CrossRef]
158. Liu, R.; He, Y.; Yuen, C.; Lau, B.P.L.; Ali, R.; Fu, W.; Cao, Z. Cost-Effective Mapping of Mobile Robot Based on the Fusion of UWB and Short-Range 2-D LiDAR. *IEEE/ASME Trans. Mechatron.* **2022**, *27*, 1321–1331. [CrossRef]

MDPI AG
Grosspeteranlage 5
4052 Basel
Switzerland
Tel.: +41 61 683 77 34

Sensors Editorial Office
E-mail: sensors@mdpi.com
www.mdpi.com/journal/sensors

MDPI

Academic Open
Access Publishing

mdpi.com

ISBN 978-3-7258-3184-5