J. Electrical Systems 20-10s (2024): 44 -52

[1] **Prashant Ahire***
[2] **S M M Naidu**
[3] **Sandeep Varpe**
[4] **Sakshi Nadarge**
[5] **Anushree Patil**

Simulating Vehicle Driving Using CARLA

JES
Journal of Electrical Systems

Abstract: - The demand for autonomous vehicles is driven by a combination of factors that are shaping the future of transportation. One significant aspect of the demand for autonomous vehicles comes from consumers who are increasingly interested in the potential benefits of self-driving technology. The convenience, safety, and potential cost savings associated with autonomous vehicles are appealing to many individuals, especially in urban areas where traffic congestion and parking challenges are prevalent. Consumers see autonomous vehicles as a way to improve their daily commute, reduce the stress of driving, and enhance overall mobility. Autonomous vehicles are incorporating a variety of sensors, including cameras, LiDAR, radar, and ultrasonic sensors, to improve perception capabilities. Sensor fusion techniques are being used to combine data from multiple sensors for more accurate and reliable object detection and tracking. Among these, the CARLA (Car Learning to Act) simulator has emerged as a leading open-source solution, offering a realistic and customized virtual environment for autonomous driving research. The increasing autonomy of vehicles necessitates a paradigm shift in testing methodologies. Traditional real-world testing is resource-intensive, time-consuming, and often constrained by safety concerns. Simulation offers a viable alternative, allowing researchers and developers to iterate and experiment rapidly in a controlled virtual environment. CARLA, as an open-source and extensible simulator, provides a valuable platform to address these challenges.

Keywords: Object Detection, Collision Avoidance, Machine Learning, Deep Learning, Autonomous Vehicles

I. INTRODUCTION

The pursuit of autonomous vehicle technology has rev- olutionized the landscape of transportation, promising safer and more efficient roads through vehicles capable of nav- igating and making decisions without human intervention. The development of autonomous systems demands meticulous testing and validation, a process that is inherently complex and often impractical to conduct solely in the physical world. As a result, simulation platforms have become instrumental in advancing autonomous vehicle research, providing a controlled and adaptable environment for testing algorithms, training machine learning models, and validating control strategies.

There are plenty of simulators to test the autonomous vehi- cle solutions having their pros and cons, such as CarCraft, Udacity, TORCS, and RRADS, etc. CarCraft offers a realistic driving experience with a focus on vehicle dynamics and control. It provides a platform for testing advanced driving algorithms. The disadvantage of CarCraft is that it has limited environment and scenario diversity compared to other simula- tors. It may not have as many features or sensors as some other simulators. Though Udacity provides a user-friendly interface and is often used for educational purposes in autonomous driving courses, Udacity has limited customization options and may not offer the same level of realism or complexity as more advanced simulators. TORCS is a well-established open-source racing simulator with a large community of users. It offers a wide range of tracks and vehicles for testing. While TORCS is suitable for racing simulations, it may lack specific features required for autonomous driving research and development. RRADS is designed specifically for robotics research and development, including autonomous vehicles. It offers flexibility and customization options. But it has limited documentation and support compared to more widely used simulators like CARLA. It may require more technical exper- tise to set up and use effectively.

CARLA simulator stands out among other autonomous vehicle driving simulators due to several key advantages. CARLA simulator offers distinct advantages that set it apart from other autonomous vehicle driving simulators.

[1] Assistant Professor, Electronics and Telecommunication, International Institute of Information Technology (I^2IT), Pune, MH, India. Email: prashanta@isquareit.edu.in

[2] Electronics and Telecommunication, International Institute of Information Technology (I^2IT), Pune, MH, India

[3] Engineering Sciences, International Institute of Information Technology (I^2IT), Pune, MH, India

[4] Electronics and Telecommunication, International Institute of Information Technology (I^2IT), Pune, MH, India

[5] Electronics and Telecommunication, International Institute of Information Technology (I^2IT), Pune, MH, India

One key advantage is its open-source nature, which allows for extensive customiza- tion and adaptation to specific research or development needs. This flexibility enables users to modify the software, integrate new features, and collaborate with the community to enhance the simulator's capabilities continuously. This open approach fosters innovation and accelerates progress in autonomous driving research. Another significant advantage of CARLA is its high level of realism and fidelity in simulation. The simulator provides detailed graphics, accurate physics, and dynamic environments that closely replicate real-world driving conditions. This realism is essential for testing and validating autonomous driving algorithms, sensor systems, and decision-making processes in a safe and controlled virtual environment. Researchers and developers can conduct extensive experiments and scenario testing in CARLA, leading to more robust and reliable autonomous vehicle systems.

In addition to its technical capabilities, CARLA's active devel- opment and maintenance by a dedicated team ensure that the simulator remains up-to-date with the latest advancements in autonomous driving technology. Regular updates, bug fixes, and new features enhance the usability and effectiveness of CARLA for researchers, developers, and educators in the autonomous vehicle domain. This commitment to continuous improvement and innovation solidifies CARLA's position as a leading autonomous vehicle driving simulator, driving progress and excellence in the field of autonomous driving research and development.

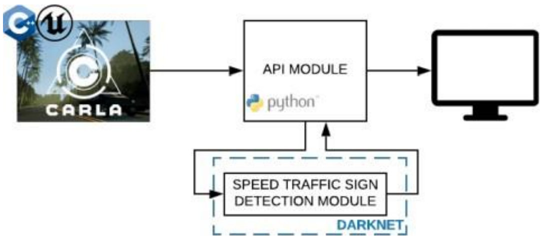

Fig. 1. Project's Pipeline

II. ARCHITECTURE OF CARLA SIMULATOR

The architecture of the autonomous vehicle consists of four major layers, as illustrated in Figure 3, that are sensor layer, perception layer, planning layer and control layer.

A. Sensor Layer

The sensor layer in the architecture of CARLA is responsible for simulating the various sensors that an autonomous vehicle typically uses to perceive its environment. These sensors provide crucial data for the perception module, allowing the vehicle to understand the surrounding world.

CARLA simulates camera sensors that capture visual information. These cameras can be configured with different parameters, such as field of view, resolution, and position on the vehicle. Visual data is essential for tasks like object detection, lane keeping, and scene understanding. LiDAR sensors simulate laser-based devices that measure distances to objects in the environment. CARLA allows users to configure LiDAR sensors with different settings, such as the number of beams, rotation speed, and range. LiDAR data is valuable for creating detailed 3D maps of the surroundings and detecting obstacles. RADAR sensors are simulated to capture radio waves reflected off objects in the environment. CARLA provides radar sensors with adjustable parameters like range and field of view. RADAR data is useful for detecting objects and estimating their velocities, particularly in adverse weather conditions. The GPS sensor simulates the global positioning system, providing information about the vehicle's geographic coordinates. GPS data is essential for localization, helping the autonomous system understand its position within the simulated environment.

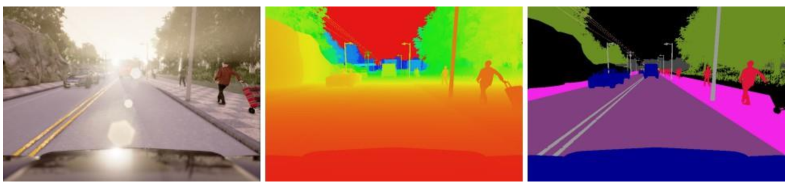

Fig. 2: Sensing modalities provided by CARLA. From left to right: normal image camera, ground depth, and ground semantic segmentation. Depth and semantic segmentation are pseudo-sensors that executes the role of perception. Additional sensor models can be plugged in via the API.

B. Perception Layer

The perception layer in the architecture of CARLA is responsible for interpreting and understanding the data collected from simulated sensors. It processes the information captured by cameras, LiDAR, RADAR, and other sensors to extract relevant details about the environment. The goal is to generate a comprehensive representation of the surroundings, which can then be used by higher-level modules, such as planning and control, for decision-making and action execution.

Object detection algorithms in the perception layer analyze sensor data to identify and locate objects in the environment. This includes recognizing vehicles, pedestrians, cyclists, and other relevant entities. Object detection is fundamental for understanding the dynamics of the scene and predicting potential interactions. Lane detection algorithms process visual sensor data to identify lane markings on the road. This information is crucial for determining the vehicle's position within the lane and aiding in tasks such as lane-keeping and lane-changing. Perception algorithms are designed to recognize and interpret traffic signs and signals. This includes identifying stop signs, traffic lights, speed limit signs, and other regulatory signs. Recognition of these elements is vital for obeying traffic rules and ensuring safe navigation.

C. Planning Layer

The planning layer in the architecture of CARLA is responsible for generating high-level plans and trajectories for the autonomous vehicle based on the information provided by the perception layer. This layer focuses on decision-making, determining the optimal course of action for the vehicle to navigate its environment safely and efficiently. The planning layer often begins with route planning, where the system decides the overall path the vehicle should take to reach its destination. This involves considering factors such as map information, user-defined way-points, and dynamic changes in the environment. Behavior planning involves making high-level decisions about the vehicle's actions. This includes determining whether the vehicle should change lanes, overtake another vehicle, make a turn at an intersection, or stop at a traffic light. The behavior planner considers the current situation and the desired destination. Path planning focuses on finding a collision-free path for the vehicle within the planned trajectory. This involves considering the vehicle's physical constraints, such as its dimensions and turning radius, while avoiding obstacles and maintaining a safe distance from other objects in the environment. Decision-making algorithms in the planning layer take into account various factors, including the current state of the vehicle, traffic conditions, legal requirements, and user-defined preferences. These algorithms aim to generate plans that balance safety, efficiency, and compliance with traffic rules.

D. Control Layer

The control layer in the architecture of CARLA is respon- sible for executing the plans and trajectories generated by the planning layer. It translates the high-level commands into low-level control signals that directly manipulate the vehicle's actuators, such as throttle, brake, and steering. The control layer ensures that the vehicle physically follows the planned trajectory while considering real-time feedback from sensors. The control layer models the dynamics of the vehicle, in- cluding its acceleration, braking, and steering behavior. The output of the planning layer is a high-level plan or trajectory, specifying the desired path and behavior of the vehicle. The control layer translates this high-level plan into low-level control commands that directly influence the vehicle's motion. These commands typically include throttle, brake, and steering inputs. Throttle control involves adjusting the engine power to control the vehicle's speed. Brake control regulates the braking force applied to the vehicle. Steering control determines the steering angle necessary to follow the planned path.

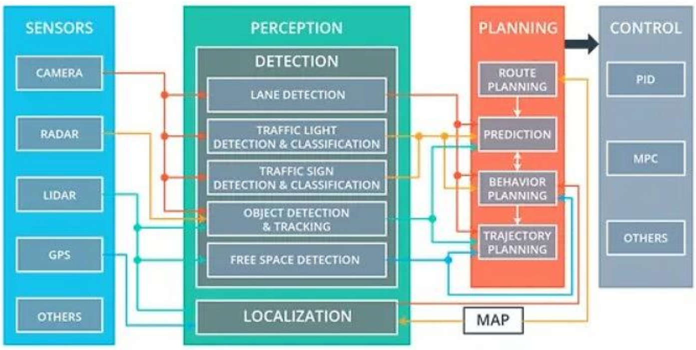

Fig. 3. Architecture of CARLA.

III. Object Detection

Object detection is a critical component in autonomous vehicle driving systems as it allows the vehicle to identify and locate various objects in its environment, such as pedestrians, vehicles, and obstacles. In CARLA, object detection is typically part of the perception layer, which processes data from simulated sensors to understand the surroundings.

The YOLO (You Only Look Once) architecture revolutionized object detection by introducing a single-stage approach that significantly improved speed without compromising accuracy. Unlike traditional two-stage detectors that first propose regions of interest and then classify those regions, YOLO performs both tasks simultaneously. This unique design enables YOLO to achieve real-time performance, making it ideal for applications requiring rapid processing, such as autonomous vehicles, surveillance systems, and interactive systems.

At its core, YOLO divides the input image into a grid and predicts bounding boxes and class probabilities for each grid cell. This grid-based approach allows YOLO to efficiently cover the entire image and detect objects at different locations and scales. By leveraging convolutional neural networks (CNNs) for feature extraction, YOLO captures rich representations of the input image, enabling robust detection across various environments and object types.

Furthermore, YOLO incorporates anchor boxes and multiple scales to improve detection accuracy. Anchor boxes are predefined bounding boxes with different aspect ratios, which help YOLO handle objects of various shapes and sizes. By predicting offsets and scales relative to these anchor boxes, YOLO achieves precise localization of objects, even in cluttered scenes or when objects are partially occluded.

The architecture of YOLO typically consists of convolutional layers for feature extraction followed by fully connected layers for predicting bounding boxes and class probabilities. Through training on large datasets and iterative refinement, YOLO has evolved through multiple versions, each enhancing performance and addressing specific challenges. These improvements include better handling of small objects, improved localization accuracy, and increased robustness to environmental variations.

During training, YOLO requires a labeled dataset with annotated bounding boxes and class labels. The network is trained using a combination of classification and regression loss functions. The classification loss measures the accuracy of the predicted class probabilities, while the regression loss measures the accuracy of the predicted bounding box coordinates.

Non-maximum suppression (NMS) is a fundamental post-processing technique widely used in object detection to refine the output of detection models. Its primary purpose is to address the issue of duplicate or redundant bounding box detections that may occur when an object is localized by multiple overlapping bounding boxes. NMS operates by iteratively selecting the bounding box with the highest confidence score among a group of overlapping boxes and suppressing (i.e., removing) all other boxes that have significant overlap with the selected box.

A. Training

Fig. 4. Object Detection

The training phase involves feeding the annotated and pre-processed dataset into the configured YOLO model. During training, the model learns to detect objects by adjusting its weights based on the disparity between predicted bounding boxes and ground truth annotations. This iterative process continues for multiple epochs, with performance monitored on the validation set to prevent over fitting.

Following training, the model’s performance is evaluated on the test set using metrics like mean Average Precision (mAP) to assess its accuracy, precision, and recall. If necessary, the trained model can undergo fine-tuning to address any deficiencies observed during evaluation. Finally, the trained YOLO model is deployed to the Carla simulator, enabling vehicles to detect and respond to objects and obstacles in real-time during simulated driving scenarios.

Training a YOLO model for object detection in the context of simulating vehicle driving using the Carla simulator involves several sequential steps. Firstly, data collection is paramount. A diverse dataset of images representing different driving scenarios within the Carla simulator needs to be collected. These scenarios should encompass various environmental factors such as different lighting conditions, weather variations, road types, and traffic densities. Additionally, the dataset should cover a wide range of objects pertinent to driving simulations, including vehicles, pedestrians, traffic signs, and obstacles commonly encountered on roads.

Once the dataset is collected, the next step is annotation. Each image in the dataset needs to be annotated with bounding boxes around objects of interest. These bounding boxes should precisely delineate the objects and be accompanied by corresponding class labels. Annotation tools like LabelImg or VIA are typically employed for this task, facilitating the marking of bounding boxes and assignment of class labels.

Following annotation, the dataset undergoes preprocessing to ensure compatibility with the YOLO architecture. This includes tasks such as resizing images to a consistent resolution, normalizing pixel values, and converting annotations into a format suitable for YOLO training, such as the Darknet annotation format. Furthermore, the dataset is divided into training, validation, and test sets to facilitate model training, hyper-parameter tuning, and performance evaluation, respectively.

With the dataset prepared, the YOLO architecture is configured for training. The appropriate YOLO version, such as YOLOv3 or YOLOv4, is selected, and its architecture is configured with parameters like the number of convolutional layers, anchor box sizes, and optimization parameters. Additionally, the loss function, typically a combination of classification loss and localization loss, is defined to guide the model's training process.

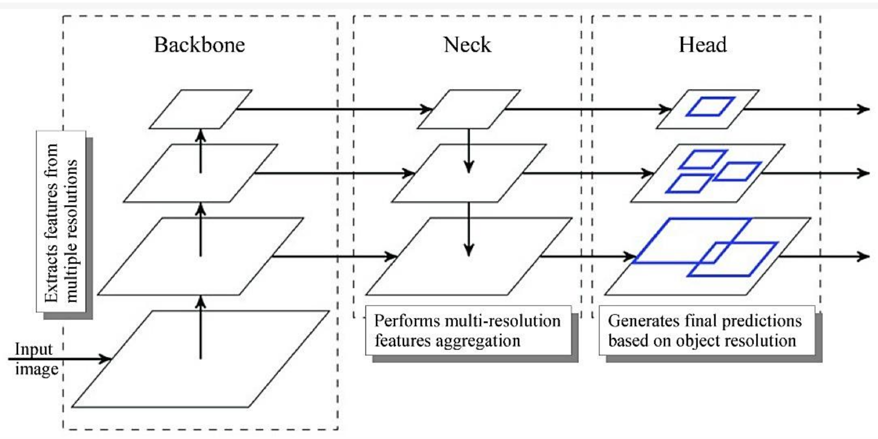

Fig. 5. Training Dataset

IV. COLLISION AVOIDANCE

Achieving collision avoidance in autonomous vehicle driving using CARLA (Car Learning to Act) involves a series of steps that integrate perception, planning, and control algorithms. The first step is perception, where sensor data from cameras, LiDAR, and RADAR is utilized to perceive the surrounding environment. This data is processed to detect and track objects such as vehicles, pedestrians, and obstacles.

The next step is planning, where a high-level plan is generated to determine the desired trajectory and behavior of the autonomous vehicle. This plan takes into account the current state of the vehicle, the detected objects, and the road conditions. Path planning algorithms are used to generate a safe and collision-free trajectory for the vehicle to follow. Traffic rules, speed limits, and other constraints are incorporated into the planning process to ensure compliance and safety.

Control algorithms come into play to execute the planned trajectory and ensure the vehicle follows the desired path. Feedback control techniques are employed to adjust the vehicle's steering, acceleration, and braking commands. The vehicle's state is continuously monitored, and adjustments are made to maintain stability and safety. The dynamics of the vehicle and its limitations are considered to ensure smooth and safe control.

To achieve collision avoidance, collision avoidance algorithms are integrated into the planning and control modules. The system continuously assesses the risk of collision with detected objects. If a potential collision is detected, appropriate actions are triggered to avoid it. These actions may involve adjusting the trajectory, decelerating, or even stopping the vehicle if necessary. Predicted motion of the detected objects and their potential future trajectories are taken into account to make informed decisions.

Testing and validation are crucial to ensure the effectiveness and safety of the collision avoidance system. CARLA's simulation environment provides a platform for extensive testing. Various scenarios, including different traffic situations, pedestrian interactions, and complex road conditions, are simulated. The performance of the collision avoidance system is evaluated in terms of safety, efficiency, and compliance with traffic rules. Based on the results of testing and validation, the algorithms are iterated and refined to enhance their performance and robustness.

V. Integration of Object Detection and

Collision Avoidance

The integration of object detection and collision avoidance techniques within the CARLA simulator is a crucial step in simulating realistic driving scenarios. Object detection techniques are responsible for identifying and localizing objects of interest in the simulated environment. These techniques utilize deep learning algorithms to analyze sensor data and classify objects into predefined categories, such as pedestrians, vehicles, and obstacles. The output of the object detection module provides valuable information about the surrounding environment, which is essential for collision avoidance.

Collision avoidance algorithms, on the other hand, utilize the object detection results to make decisions and control the vehicle's behavior. These algorithms aim to predict potential collisions and take appropriate actions to avoid them. They can be rule-based systems that follow predefined rules and heuristics, or machine learning-based approaches that learn from data to make decisions. Sensor fusion techniques can also be employed to combine information from multiple sensors, such as cameras, LiDAR, and RADAR, to improve the accuracy and reliability of collision avoidance.

The integration of object detection and collision avoidance involves feeding the output of the object detection module into the collision avoidance algorithm. The object detection results provide the necessary inputs, such as the positions, velocities, and types of detected objects, to the collision avoidance algorithm. Based on this information, the collision avoidance algorithm determines the appropriate actions to be taken by the simulated vehicle, such as braking, steering, or accelerating, to avoid potential collisions.

One of the key challenges in this integration process is ensuring the synchronization and real-time performance of both components. The object detection module should provide timely and accurate results to the collision avoidance algorithm to enable quick decision-making. Additionally, the collision avoidance algorithm should respond promptly to the detected objects to ensure the safety of the simulated driving scenario.

Another consideration is the scalability and adaptability of the integrated system. The object detection and collision avoidance techniques should be capable of handling various driving scenarios, including different road conditions, traffic densities, and object types. The system should be able to generalize well and adapt to new and unseen situations.

Overall, the integration of object detection and collision avoidance techniques within the CARLA simulator is crucial for creating realistic and safe driving simulations. It enables the evaluation and validation of ADAS systems by simulating complex driving scenarios and assessing the performance of the integrated system in avoiding potential collisions.

VI. Experimental Results

This research paper has presented a comprehensive study on simulating vehicle driving using the CARLA simulator with integrated object detection and collision avoidance techniques. The objective of this research was to enhance the realism and accuracy of simulated driving scenarios, ultimately contributing to the development and evaluation of advanced driver assistance systems (ADAS) and autonomous driving technologies.

Through the integration of object detection techniques with a collision avoidance algorithm, the proposed system demonstrated promising results. The object detection module exhibited high accuracy in identifying and localizing various objects, including pedestrians, vehicles, and obstacles. The collision avoidance algorithm effectively utilized the object detection results to make timely and appropriate decisions, successfully avoiding potential collisions in different driving scenarios.

The research findings highlight the significance of integrating object detection and collision avoidance techniques within driving simulators. By accurately representing real-world driving scenarios, the CARLA simulator with the proposed system enables the evaluation and validation of ADAS systems in a safe and controlled environment.

In conclusion, the integration of object detection and collision avoidance within the CARLA simulator offers a valuable tool for simulating and evaluating vehicle driving scenarios. This research contributes to the advancement of ADAS technologies and paves the way for safer and more efficient autonomous driving systems in the future.

Fig. 6. Experimental Result

Fig. 7. Collision Avoidance

VII. RELATED WORK

Prior studies have explored various aspects of CARLA simulator, such as environment modeling, vehicle dynamics, and sensor simulation. These works provide foundational knowledge and techniques for integrating advanced functionalities like object detection and collision avoidance. Research on object detection algorithms, including YOLO (You Only Look Once) variants like YOLOv7, has been extensively conducted. Understanding the strengths and limitations of these algorithms is crucial for implementing efficient and accurate object detection within the simulation environment. Some research endeavors have explored similar integrations within different simulators or environments. These studies provide valuable insights into the challenges and strategies involved in combining object detection with collision avoidance for simulated vehicle driving. Research on transfer learning techniques can be relevant for this project, particularly for bridging the gap between simulation and real-world deployment. By leveraging transfer learning, models trained in simulation environments can be fine-tuned to perform effectively in real-world scenarios.

VIII. CONCLUSION

CARLA simulator marks a significant milestone in advancing the capabilities of autonomous vehicle simulation. This fusion of cutting-edge technologies enables more accurate perception and proactive decision-making, fostering safer and more realistic driving scenarios.

With YOLOv7, the simulator can efficiently detect and classify objects in the environment with remarkable speed and accuracy. This real-time detection capability enhances the vehicle’s awareness of its surroundings, enabling it to respond promptly to dynamic changes in the environment, such as the presence of pedestrians, vehicles, or obstacles.

Furthermore, the incorporation of collision avoidance mechanisms adds an extra layer of safety by enabling the simulated vehicle to anticipate and mitigate potential collisions. By continuously analyzing the environment and predicting potential hazards, the vehicle can navigate complex scenarios with greater confidence and reliability.

IX. FUTURE WORK

In the realm of simulating vehicle driving using CARLA simulator with object detection using YOLOv7 and collision avoidance, there exist several promising avenues for future exploration and enhancement. One direction for future work involves delving deeper into the realm of object detection algorithms. While YOLOv7 provides impressive real-time performance and accuracy, continued research into more advanced algorithms could further improve detection capabilities, particularly in complex scenarios with occlusions or varying lighting conditions. Exploring novel architectures or integrating multiple detection models for different object classes could potentially enhance the overall perception system within the simulator.

Additionally, future efforts could focus on advancing collision avoidance strategies within the simulated environment. While the current implementation employs proactive measures to mitigate potential collisions, further research could explore predictive modeling techniques to anticipate the behavior of other vehicles and pedestrians. By incorporating predictive capabilities into collision avoidance algorithms, simulated vehicles could make more informed decisions in dynamic and unpredictable scenarios, ultimately enhancing safety and adaptability.

REFERENCES

[1] Sreenivas, K.; Kamakshiprasad, V. Improved image tamper localisation using chaotic maps and self-recovery. J. Vis. Commun. Image Represent. 2017, 49, 164–176.

[2] Ali, A.; Hassan, A.; Ali, A.R.; Khan, H.U.; Kazmi, W.; Zaheer, A. Real-time vehicle distance estimation using single view geometry. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Snowmass, CO, USA, 1–5 March 2020; pp. 1111–1120.

[3] Ding, M.; Zhang, Z.; Jiang, X.; Cao, Y. Vision-based distance measurement in advanced driving assistance systems. Appl. Sci. 2020, 10, 7276.

[4] Lim, Y.-C.; Lee, C.-H.; Kwon, S.; Jung, W.-Y. Distance estimation algorithm for both long and short ranges based on stereo vision system. In Proceedings of the 2008 IEEE Intelligent Vehicles Symposium, Eindhoven, The Netherlands, 4–6 June 2008; pp. 841–846.

[5] Liu, L.-C.; Fang, C.-Y.; Chen, S.-W. A novel distance estimation method leading a forward collision avoidance assist system for vehicles on highways. IEEE Trans. Intell. Transp. Syst. 2016, 18, 937–949.

[6] Ha¨ne, C.; Sattler, T.; Pollefeys, M. Obstacle detection for self-driving cars using only monocular cameras and wheel odometry. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015; pp. 5101–5108.

[7] Tram, V.T.B.; Yoo, M. Vehicle-to-vehicle distance estimation using a low-resolution camera based on visible light communications. IEEE Access 2018, 6, 4521–4527.

[8] Kim, G.; Cho, J.-S. Vision-based vehicle detection and inter-vehicle distance estimation. In Proceedings of the 2012 12th International Conference on Control, Automation and Systems, Jeju, Republic of Korea, 17–21 October 2012; pp. 625–629.

[9] Min, K.; Han, S.; Lee, D.; Choi, D.; Sung, K.; Choi, J. SAE Level 3 Autonomous Driving Technology of the ETRI. In Proceedings of the 2019 International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Republic of Korea, 16–18 October 2019; pp. 464–466.

[10] Sanil, N.; Venkat, P.A.N.; Rakesh, V.; Mallapur, R.; Ahmed, M.R. Deep Learning Techniques for Obstacle Detection and Avoidance in Driverless Cars. In Proceedings of the 2020 International Conference on Artificial Intelligence and Signal Processing (AISP), Amaravati, India, 10–12 January 2020.

[11] Barea, R.; Perez, C.; Bergasa, L.M.; Lopez-Guillen, E.; Romera, E.; Molinos, E.; Ocana, M.; Lopez, J. Vehicle Detection and Localization Using 3D LIDAR Point Cloud and Image Semantic Segmentation. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 3481–3486.

[12] Pe´rez-Gil, O´ .; Barea, R.; Lo´pez-Guille´n, E.; Bergasa, L.M.; Revenga, P.A.; Gutie´rrez, R.; D´ıaz, A. DQN-Based Deep Reinforcement Learning for Autonomous Driving. In Proceedings of the Advances in Physical Agents II, Alcala´ de Henares, Spain, 19–20 November 2020; Springer International Publishing: Cham, Switzerland, 2021; pp. 60–67.

[13] Marti, E.; de Miguel, M.A.; Garcia, F.; Perez, J. A Review of Sensor Technologies for Perception in Automated Driving. IEEE Intell. Transp. Syst. Mag. 2019, 11, 94–108.

[14] Serban, A.C.; Poll, E.; Visser, J. A Standard Driven Software Architecture for Fully Autonomous Vehicles. In Proceedings of the 2018 IEEE International Conference on Software Architecture Companion (ICSA-C), Seattle, WA, USA, 30 April–4 May 2018; pp.120–127.

[15] Dworak, D.; Ciepiela, F.; Derbisz, J.; Izzat, I.; Komorkiewicz, M.; Wo´jcik, M. Performance of LiDAR Object Detection Deep Learning Architectures Based on Artificially Generated Point Cloud Data from CARLA Simulator. In Proceedings of the 2019 24th International Conference on Methods and Models in Automation and Robotics (MMAR), Miedzyzdroje, Poland, 26–29 August 2019; pp. 600–605.

[16] Mauri, A.; Khemmar, R.; Decoux, B.; Haddad, M.; Boutteau, R. Real-time 3D multi-object detection and localization based on deep learning for road and railway smart mobility. J. Imaging 2021, 7, 145.

[17] Zhu, J.; Fang, Y. Learning object-specific distance from a monocular image. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3839–3848.

[18] Reddy, N.D.; Vo, M.; Narasimhan, S.G. Carfusion: Combining point tracking and part detection for dynamic 3d reconstruction of vehicles. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1906–1915.

[19] Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; Koltun, V. CARLA: An open urban driving simulator. In Proceedings of the Conference on Robot Learning, Mountain View, CA, USA, 13–15 November 2017;pp. 1–16.

[20] Zhu, X.; Lyu, S.; Wang, X.; Zhao, Q. TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 2778–2788.

[21] Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. Adv. Neural Inf. Process. Syst. 2015, 28.

[22] Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.

[23] Sermanet, P.; Eigen, D.; Zhang, X.; Mathieu, M.; Fergus, R.; LeCun,Y. Overfeat: Integrated recognition, localization and detection using convolutional networks. arXiv 2013, arXiv:1312.6229.

[24] Li, B.; Zhang, T.; Xia, T. Vehicle detection from 3d lidar using fully convolutional network. arXiv 2016, arXiv:1608.07916.

[25] Li, B. 3d fully convolutional network for vehicle detection in point cloud. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 1513–1518.

[26] Zhang, S.; Benenson, R.; Omran, M.; Hosang, J.; Schiele, B. Towards reaching human performance in pedestrian detection. IEEE Trans. Pattern Anal. Mach. Intell. 2017, 40, 973–986.

[27] Luo, H.; Yang, Y.; Tong, B.; Wu, F.; Fan, B. Traffic sign recognition using a multi-task convolutional neural network. IEEE Trans. Intell. Transp. Syst. 2017, 19, 1100–1111.

[28] Behrendt, K.; Novak, L.; Botros, R. A deep learning approach to traffic lights: Detection, tracking, and classification. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 1370–1377.

[29] Weber, M.; Wolf, P.; Zo¨llner, J.M. DeepTLR: A single deep convolutional network for detection and classification of traffic lights. In Proceedings of the 2016 IEEE Intelligent Vehicles Symposium (IV), Gothenburg, Sweden, 19–22 June 2016; pp. 342–348.