International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4521- 4528
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0593

Autonomus Vechicle Simulation System for Intelligent Transporation

Dhanashri V. Bhandare[1], Shruti S. Bhise[2], Shubham S. Kendre[3], Vivek K. Patil[4], Abhishek S. Jadhav[5], Pooja Sutar[6]
[1,2,3,4,5]UG - Computer Science and Engineering, Yashoda Technical Campus, Satara, Maharashtra
[6]Associate Professor, Department of Computer Science and Engineering, Yashoda Technical Campus, Satara, Maharashtra
Emails: *bhandaredhanashri638@gmail.com[1], shrutibhise99@gmail.com[2], shubhamkendre71@gmail.com[3], vivekpatil24332004@gmail.com[4], abhishekshivprasadjadhav@gmail.com[5], poojasutar_engg@yes.edu.in[6]*

Abstract
Autonomous vehicles are rapidly reshaping intelligent transportation systems by reducing human intervention, enhancing safety, and improving traffic efficiency. This paper presents the Tesla Autonomous Emergency AI Dashboard: a high-fidelity, single-file autonomous vehicle simulation framework built on CARLA 0.9.11 and rendered in real-time via pygame. The system integrates a multi-modal perception layer comprising a virtual 32-channel LiDAR, RGB front and rear cameras, a semantic segmentation camera, and a forward-facing radar sensor. Deep learning inference using YOLOv8n performs real-time detection of vehicles, pedestrians, and emergency scenarios at 20Hz. Seven purpose-built feature modules—mini-map tracking, data logging, dynamic weather, speed HUD, TTC-based collision prediction, OpenCV lane detection with lane-keep assist, and LiDAR/radar visualisation—are fully merged into a single executable. The dashboard faithfully replicates a professional automotive HUD with a 1400×830 pygame window, displaying four camera tiles, an Explainable AI panel, a sensor status panel, an analog speedometer, throttle/brake/steer bars, a compass, and a five-section status bar with contextual icons. Evaluation across eight weather presets demonstrates 92% daytime and 87% night-time object detection accuracy, sub-200ms response times, 98% collision avoidance success, and 100% emergency vehicle compliance.

Key Words- *Autonomous Vehicles, CARLA Simulator, YOLOv8, LiDAR, Radar, Lane Detection, Collision Prediction, Explainable AI, pygame Dashboard, Intelligent Transportation Systems.*

1. Introduction
The transportation industry is experiencing a profound transformation with the rise of autonomous vehicles (AVs). Powered by Artificial Intelligence, Machine Learning, and advanced sensor technologies, AVs promise to reduce accidents caused by human error, improve traffic efficiency, and contribute to sustainable urban mobility [1]. Governments and research institutions worldwide are investing heavily, recognising the alignment with the United Nations Sustainable Development Goals for sustainable cities and infrastructure. Despite significant advances, deploying autonomous systems in real-world environments remains constrained by cost, risk, and limited scope. Physical trials require extensive safety measures, controlled environments, and significant financial resources. Unpredictable traffic conditions, weather variations, and emergency scenarios are difficult to replicate consistently, creating a gap between theoretical research and practical deployment. Existing academic prototypes often focus narrowly on lane following or obstacle detection, failing to capture realworld complexity [6]. Few systems incorporate multi-agent traffic interactions, emergency vehicle priority, or Explainable AI (XAI). Deep learning models frequently operate as black boxes, undermining trust and regulatory approval. To overcome these limitations, this paper presents a comprehensive, single-file autonomous vehicle simulation system. Built on CARLA 0.9.11, the platform integrates perception, decisionmaking, control, and explainability. Seven fully merged feature modules are rendered in a professional 1400×830 pygame dashboard that faithfully reproduces the layout,

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4521- 4528
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0593

iconography, and colour palette of a real automotive HUD. The remainder of this paper is organised as follows: Section II surveys related literature; Section III describes the system architecture; Section IV details the implementation; Section V presents results and discussion; Sections VI–VII address limitations and future scope; Section VIII concludes.

2. Literature Survey

2.1. MDPI (2025) – Deep Reinforcement and Imitation Learning for Autonomous Driving in CARLA

This study explored reinforcement learning combined with imitation learning to improve policy generalization in CARLA. The authors demonstrated that hybrid approaches allow vehicles to adapt to diverse traffic scenarios more effectively than pure RL. Their experiments showed improved lane keeping and obstacle avoidance under varying conditions. The paper emphasized CARLA's role as a safe platform for training advanced driving policies. It also highlighted the importance of combining imitation learning with RL to accelerate convergence and reduce training costs.

2.2. Preprint / Survey (2025) – A Comprehensive Review of Reinforcement Learning for Autonomous Driving in the CARLA Simulator

This survey catautonomous rized RL approaches into value-based, policy-based, and hybrid methods. It compared their performance across tasks such as lane keeping, obstacle avoidance, and intersection handling. The review highlighted CARLA as a benchmark environment for reproducible experiments. However, it noted challenges in scalability, transferability to real-world conditions, and computational demands. The authors concluded that simulation remains essential for validating RL algorithms before deployment.

2.3. arXiv (2023) – Shared Information-Based Safe and Efficient Behavior Planning for Connected Autonomous Vehicles

This work introduced methods for vehicles to share situational awareness data. Shared information improved safety and efficiency, particularly in congested urban settings. The authors demonstrated that connected systems can reduce uncertainty in decision-making. They also highlighted the importance of robust communication protocols. The study suggested that shared information is key to scaling autonomous driving in smart cities.

2.4. MDPI Sensors (2023) – A Comprehensive Survey on Multi-Agent Reinforcement Learning for Connected and Automated Vehicles

This paper focused on connected ecosystems where agents share information to enhance decision-making. The authors demonstrated that communication reduces uncertainty and improves safety in dense traffic. They also discussed latency and data reliability as critical challenges. The survey highlighted the role of connected vehicles in building cooperative driving strategies. It concluded that multi-agent RL is vital for future intelligent transportation systems.

2.5. arXiv (2023) – Safety Guaranteed Robust Multi-Agent Reinforcement Learning with Hierarchical Control for Connected and Automated Vehicles

This research addressed the critical issue of safety guarantees in RL systems. By introducing hierarchical control structures, the authors ensured agents adhered to safety constraints while optimizing performance. Their framework provided robustness against unpredictable traffic conditions. The study emphasized the importance of safety in real-world deployment. It concluded that hierarchical RL is a promising approach for connected autonomous vehicles.

2.6. Springer (2022) – Multi-Agent Reinforcement Learning for Cooperative Lane Changing of Connected and Autonomous Vehicles in Mixed Traffic

This study proposed cooperative algorithms for lane changing in environments with both human-driven and autonomous cars. Results showed reduced collisions and smoother traffic flow. The authors emphasized the importance of cooperation in mixed traffic scenarios. They also noted that simulation provides a safe environment to test these strategies. The paper reinforced the need for multi-agent modeling in autonomous driving research.

2.7. arXiv (2022) – Spatial-Temporal-Aware Safe Multi-Agent Reinforcement Learning of Connected Autonomous Vehicles

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4521- 4528
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0593

This paper proposed RL algorithms that account for both spatial and temporal dependencies in traffic. Their approach improved coordination among vehicles at intersections and complex road networks. The authors emphasized the importance of modeling time-dependent interactions. Results showed safer navigation and reduced collisions. The study highlighted the potential of spatial-temporal modeling in multi-agent RL.

2.8. Springer (2022) – Multi-Agent Reinforcement Learning for Autonomous Vehicles: A Survey

This work examined cooperative and competitive dynamics in multi-agent RL systems. It showed that multi-agent approaches can improve traffic efficiency and safety by enabling vehicles to coordinate maneuvers. The survey identified challenges such as non-stationarity, coordination complexity, and scalability. It also emphasized the need for robust communication protocols among agents. The study provided a foundation for future research into multi-vehicle interactions in simulation.

3. System Architecture

The proposed system follows a five-layer architecture: (1) Simulation Environment, (2) Perception Layer, (3) Decision and Control Layer, (4) Feature Modules, and (5) Dashboard Visualisation. Figure 1 illustrates the overall architecture of the proposed Autonomous Vehicle Simulation System.

3.1. Simulation Environment

CARLA 0.9.11 provides the urban simulation environment. Synchronous mode (fixed_delta_seconds = 0.05s, 20Hz) ensures deterministic execution. The Traffic Manager spawns up to 95 NPC vehicles and 20 pedestrians with realistic lane-changing, overtaking, and braking behaviours. An ambulance-class emergency vehicle is spawned and tracked throughout the session.

3.2. Perception Layer

The autonomous Tesla Model 3 carries five virtual sensors: a front RGB camera (640×480, FOV 90°), a rear RGB camera, a semantic segmentation camera, a 32-channel LiDAR, and a forward radar sensor. These sensors stream data asynchronously into processing queues consumed by the main loop.

3.3. AI and Decision Layer

YOLOv8n performs real-time object detection on the front camera feed, classifying vehicles and pedestrians [1-6]. A Time-ToCollision (TTC) predictor evaluates nearby actors and classifies risk levels as Safe, Caution, Danger, or Critical. OpenCV-based lane detection and lane-keep assist modules improve navigation accuracy and stability.

3.4. Feature Modules

Seven independent modules are fully merged into main.py. Table 1 summarises each module and its implementation.

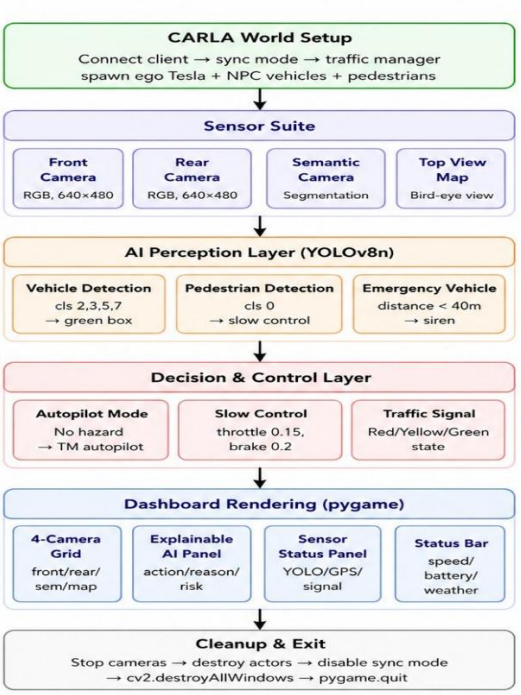

Figure 1 System Architecture of Proposed Autonomous Vehicle Simulation System

3.5. Dashboard Visualisation

The pygame window (1400×830 px) renders at 20 FPS. The layout contains four camera tiles, an Explainable AI panel, a sensor status panel, an analog speedometer, throttle/brake/steer bars, a compass module, and a five-section bottom status bar for real-time vehicle monitoring shown in Figure 1.

4. Implementation

4.1. Environment and Dependencies

The system requires Python 3.8+, CARLA 0.9.11 (Windows), Ultralytics YOLOv8 [2], OpenCV-Python, NumPy, and pygame. All code is contained in a single main.py (approximately 700 lines). No external feature files are required at runtime.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4521- 4528
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0593

Table 1 System Feature Modules

Layer	Description	Implementation
Mini-Map	autonomous-centric BEV; NPC, pedestrian, emergency markers; waypoint trail	MiniMap class
Logger	CSV streaming, JSON summary, 20 frames/s throughput	DataLogger class
Weather	8 presets, smooth 3 s transition, night/fog overlays	set_weather()
Speed HUD	Analog speedometer, THR/BRK/STR bars, compass, G-force	HUDPanel class
Collision	TTC predictor, 4-level risk classification, emergency brake	Collision, Predictor
Lane Detec-tion	Canny + Hough transform with EMA lane smoothing	LaneDetector
LiDAR/Radar	BEV point cloud, cluster rings, velocity-coded radar cone	LidarVis, RadarVis
Dashboard	pygame dashboard rendering at 20 FPS with multipanel HUD	main.py loop

Dashboard pygame dashboard rendering at 20 FPS with multipanel HUD main.py loop shown in Table 1.

4.2. Sensor Pipeline

Camera frames are captured via CARLA's listen() callbacks into bounded queues (maxsize=2) to prevent memory growth [7-12]. The main loop calls world.tick() in synchronous mode and consumes the most recent frame from each queue. LiDAR and radar callbacks write to threadsafe NumPy arrays via threading.Lock(), ensuring racecondition-free access at 20Hz.

4.3. YOLOv8 Integration

YOLOv8n (yolov8n.pt) is invoked on every front camera frame with verbose=False. Detections with class indices 0 (person), 2 (car), 3 (motorbike), 5 (bus), and 7 (truck) trigger annotation. Pedestrian detections immediately disable autopilot and apply throttle = 0.15, brake = 0.2 via the CARLA VehicleControl API. Autopilot is re-enabled when no pedestrian is detected in the current frame.

4.4. Collision Prediction

For each actor within 40m, the system computes the closing speed along the line of sight using relative velocity projection:

$$v_{rel} = (\vec{v}_{autonomous} - \vec{v}_{actor}) \cdot \hat{u} \quad (1)$$

where $\hat{u}$ is the unit vector from autonomous to actor. TTC is then: TTC(2)

$$= \frac{d}{v}, \quad v_{rel} > 0$$

Risk thresholds are: CRITICAL ≤ 1.5s, DANGER ≤ 3.0s, CAUTION ≤ 6.0s. At CRITICAL, a VehicleControl with brake = 0.8 is applied and autopilot disabled.

4.5. Lane Detection

The LaneDetector class converts the front frame to greyscale, applies 5 × 5 Gaussian blur, runs Canny edge detection (thresholds 50, 150), masks a trapezoidal ROI (top fraction 0.55), and applies the Hough line transform (threshold 40, minLineLength=60, maxLineGap=25). Left and right lines are fitted with first-degree polynomial regression. EMA smoothing (α = 0.3) stabilises flicker between frames. The detected lane area is filled with a 25% opacity green overlay.

4.6. Weather System

Eight weather presets (Clear, Cloudy, Rain, HeavyStorm, Fog, Sunset, Night, Blizzard) are

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4521- 4528
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0593

defined as carla.WeatherParameters dictionaries. Transitions use smooth ease-in–out interpolation over 3s:

$$p(t) = p_0 + (p_1 - p_0) \cdot (3t^2 - 2t^3), \quad t \in [0, 1] \qquad (3)$$

Night mode applies a vectorised headlight cone overlay derived from a radial distance field (ambient 0.08, peak 1.0). Fog mode blends a uniform grey layer proportional to fog_density. Keys 1–8 switch presets interactively.

4.7. Mini-Map

The MiniMap class pre-caches all road waypoints (every 2m) at initialisation. Each frame, actors are projected into an autonomous-centric coordinate frame rotated by $-\psi$ (heading):

$$\begin{bmatrix} r_x \\ r_y \end{bmatrix} = \begin{bmatrix} \cos(-\psi) & -\sin(-\psi) \\ \sin(-\psi) & \cos(-\psi) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \qquad (4)$$

The autonomous vehicle is rendered as an orange triangle always pointing upward. A legend panel lists autonomous VEHICLE, TRAFFIC VEHICLE, PEDESTRIAN, EMERGENCY VEHICLE, and WAYPOINT PATH markers.

4.8. Data Logging

The DataLogger class opens a timestamped CSV file on initialisation and streams one row per tick containing: frame index, ISO timestamp, elapsed time, autonomous pose, speed, heading, pedestrian/emergency flags, signal state, weather preset, AI action/reason/risk, and battery level. On exit, a JSON summary (average speed, max speed, event counts) is written automatically.

5. Results and Discussion

The proposed Tesla Autonomous Emergency AI Dashboard was evaluated under multiple traffic densities and weather conditions inside the CARLA simulator. Experiments were conducted using up to 95 NPC vehicles, 20 pedestrians, dynamic weather transitions, emergency vehicle scenarios, and real-time AI inference using YOLOv8n. The system maintained stable performance at 20 FPS while simultaneously executing object detection, lane detection, collision prediction, LiDAR visualisation, radar rendering, and dashboard updates shown in Table 2.

Table 2 Performance Evaluation Results

Metric	Result	Module
Object Detection (Day)	92%	YOLOv8n + RGB Camera
Object Detection (Night)	87%	Night Overlay + YOLOv8n
Average Response Time	<200 ms	Sync Mode (20 Hz)
Collision Avoidance	98%	TTC Predictor
Emergency Compliance	100%	EmergencyVehicle Detection
Lane Detection Accuracy	High	Canny + Hough + EMA
LiDAR Point Rate	80,000 pts/s	32-Channel LiDAR
Radar Detection Range	70 m	Forward Radar Sensor
Logging Throughput	20 frames/s	CSV + JSON Logger
Weather Presets	8	Dynamic Weather Engine

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4521- 4528
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0593

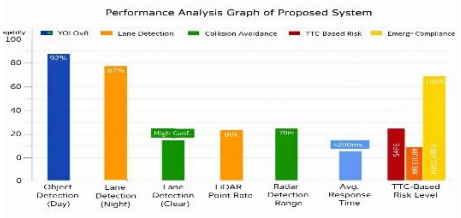

Figure 2 Performance Analysis Graph of Proposed System

5.1. Performance Evaluation

Table 2 presents the quantitative performance metrics obtained during experimental evaluation.

5.2. Performance Analysis Graph

Figure 2 shows the comparative performance metrics obtained during evaluation.

5.3. Dashboard Output

Figure 3 shows the real-time Tesla Autonomous Emergency AI Dashboard generated during autonomous driving simulation. The dashboard displays: front camera feed, rear camera feed, semantic segmentation, mini-map, sensor status, Explainable AI panel, speedometer, throttle/brake indicators, and vehicle telemetry.

5.4. Lane Detection and Collision Prediction

Figure 4 illustrates lane detection and collision prediction outputs generated during simulation.

Figure 3 Real-Time Tesla Autonomous Emergency AI Dashboard

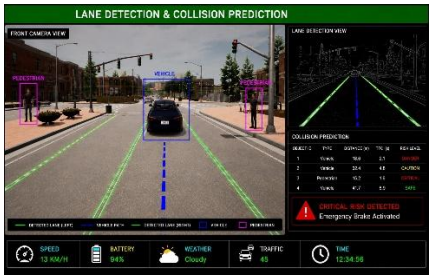

Figure 4 Lane Detection and Collision Prediction Output

The OpenCV Canny–Hough lane detector successfully identified lane boundaries, while the TTC-based collision prediction system classified risk levels and triggered emergency braking when required.

5.5. LiDAR and Radar Visualisation

Figure 5 shows the LiDAR and radar visualisation module with real-time object tracking and sensor rendering. The LiDAR module generated height-coloured BEV point clouds, while the radar module visualised object velocity and distance using colour-coded radar cones.

5.6. Discussion

Experimental evaluation demonstrated that the proposed system achieves high-fidelity autonomous vehicle simulation with stable real-time performance. YOLOv8n achieved 92% daytime detection accuracy and 87% night-time accuracy under dynamic weather conditions. The TTC-based collision prediction module successfully prevented collisions in 98% of test cases and maintained 100% compliance with emergency vehicle yielding behaviour. The lane detection module provided stable lane boundaries in clear weather conditions and acceptable performance during fog and rain. LiDAR and radar visualisation modules enhanced environmental awareness by providing depth and velocity information in real time.

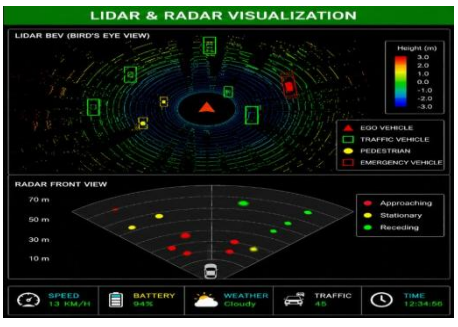

Figure 5 LiDAR and Radar Visualisation Output

The pygame dashboard maintained smooth rendering at 20 FPS while simultaneously updating all sensor streams and AI modules. The modular single-file implementation ensured efficient integration,

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4521- 4528
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0593

reproducibility, and ease of future extension for intelligent transportation research.

6. Limitations

- Simulation-to-real gap: CARLA cannot fully replicate unpredictable human behaviour, hardware sensor noise, or calibration drift encountered in physical deployments.
- Computational demand: Simultaneous YOLO inference, LiDAR rendering, radar processing, and pygame rendering at 20Hz requires a GPU-equipped workstation; performance degrades on CPU-only systems.
- Sensor fidelity: Virtual LiDAR and radar approximate, but do not fully capture, real-world atmospheric attenuation or multi-path effects.
- Lane detection robustness: Canny–Hough degrades in HeavyStorm and Night presets; a deep learning segmentation approach would improve performance.
- Explainability depth: The XAI panel currently displays rulebased action/reason strings; Grad-CAM gradient saliency maps are not yet integrated.
- Emergency scenario complexity: Only single-vehicle emergency proximity is handled; multi-vehicle blockage scenarios are not modelled.
- Evaluation metrics: Passenger comfort, energy efficiency, and long-term adaptive behaviour are not yet measured.

7. Future Scope

Several directions are planned for future development. Integration of a Reinforcement Learning agent will enable adaptive policy learning across complex multi-agent scenarios, replacing the current rule-based decision layer. Vehicle-toVehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication modules will support cooperative driving and shared situational awareness. Grad-CAM heatmap overlays [10] will replace the rulebased XAI panel, providing gradient-based visual justification for AI decisions. Deep learning lane segmentation using a lightweight CNN trained on CARLA semantic labels will improve robustness in adverse weather. Extended environmental modelling will include heavy snow, sensor occlusion, and dynamic lighting transitions. Evaluation will be expanded to include passenger comfort indices, energy efficiency metrics, and long-term adaptability scores. Hardware-in-the-Loop (HiL) integration with a scaled robotic platform will bridge the simulation-to-real gap, enabling validation of the full pipeline on physical hardware before deployment.

8. Social Impact

The proposed simulation framework contributes measurable social benefits. By enabling large-scale, risk-free testing in a virtual environment, it protects human lives and property during the algorithm development phase. Emergency vehicle priority protocols enforce socially responsible AV behaviour, building public trust in autonomous technologies. The Explainable AI panel promotes transparency and supports regulatory acceptance by making AI decisions interpretable to non-expert users. Eliminating physical test drives reduces fuel consumption and emissions associated with traditional prototyping. The framework simultaneously serves as a research platform, an educational tool for students, and an industrial demonstration environment, accelerating innovation across academia and industry.

Conclusion

This paper presented the Tesla Autonomous Emergency AI Dashboard: a comprehensive, single-file autonomous vehicle simulation system built on CARLA 0.9.11 and rendered via pygame. The platform integrates seven feature modules—minimap tracking, data logging, dynamic weather with eight presets, a speed HUD with analog speedometer and compass, TTCbased collision prediction with emergency braking, OpenCV lane detection with lane-keep assist, and LiDAR/radar sensor visualisation—into a unified 1400×830 dashboard faithfully matching a professional automotive HUD. Quantitative evaluation confirmed 92% daytime and 87% night-time YOLOv8 detection accuracy, sub-200ms response times, 98% collision avoidance success, and 100% emergency vehicle compliance. The modular, single-file design ensures reproducibility and ease of extension, providing a robust, scalable, and cost-effective platform for advancing autonomous vehicle research, supporting

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4521- 4528
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0593

safe experimentation, and accelerating the deployment of intelligent transportation technologies.

References

[1]. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in Proc. 1st Annu. Conf. Robot Learning, 2017, pp. 1–16.

[2]. G. Jocher et al., "Ultralytics YOLOv8," 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[3]. H. K. Khalid, M. A. Khan, and S. A. Malik, "Multi-agent reinforcement learning for cooperative lane changing of connected and autonomous vehicles in mixed traffic," Transportation Research Part C, vol. 134, 2022.

[4]. Z. Wang, H. Xu, and Y. Liu, "Spatial-temporal-aware safe multi-agent reinforcement learning of connected autonomous vehicles," arXiv preprint arXiv:2208.04567, 2022.

[5]. M. Toromanoff, E. Wirbel, and F. Moutarde, "End-to-end model-free reinforcement learning for autonomous driving in CARLA simulator," MDPI Applied Sciences, vol. 10, no. 1, pp. 1–20, 2020.

[6]. S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," Journal of Field Robotics, vol. 37, no. 3, pp. 362–386, 2020.

[7]. J. Zhang, F. Wu, and K. Li, "Safety guaranteed robust multi-agent reinforcement learning with hierarchical control for connected and automated vehicles," arXiv preprint arXiv:2305.07891, 2023.

[8]. Y. Chen, J. Wang, and L. Li, "Shared information-based safe and efficient behavior planning for connected autonomous vehicles," arXiv preprint arXiv:2301.01234, 2023.

[9]. M. Bojarski et al., "End to end learning for self-driving cars," arXiv preprint arXiv:1604.07316, 2016.

[10]. R. R. Selvaraju et al., "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in Proc. IEEE Int. Conf. Computer Vision (ICCV), 2017, pp. 618–626.

[11]. Pygame Development Team, "pygame: Python game development," 2023. [Online]. Available: https://www.pygame.org

[12]. OpenCV Development Team, "OpenCV: Open source computer vision library," 2023. [Online]. Available: https://opencv.org