Web-Based Self-Driving Car Simulation using JavaScript and Neural Network

Bharat Maheshwari, Deepak Kumar, Dr. P Sudhakar
School of Computer Science Engineering, Galgotias University Greater Noida, India

***Abstract*— This paper describes a web-based autonomous vehicle simulator constructed through JavaScript, HTML5, and the Canvas API, with a built-in feedforward neural network. The simulator simulates some of the key aspects of autonomous driving, including lane detection, obstacle avoidance, and dynamic path planning, all rendered in real time in the web domain. Ray-casting sensors simulate LiDAR-like sensing, and decisions are made by a neural network whose weights are updated through a mutation-based learning method based on genetic algorithms. Compared to resource-intensive high-fidelity simulators such as CARLA or LGSVL, or low-fidelity pedagogic simulators such as SIM.JS without adaptive intelligence, this project strikes a balance between computational expense and behavioral richness. The simulation showed progressively more effective learning in a 1,000-agent simulated population, from 0% success in early runs up to over 75% successful navigation in the 15th generation. The entire platform is browser-native and makes no use of external libraries or installations, and is thus well-suited in particular to educational institutions, prototyping, and outreach events. This project offers an open and scalable alternative to existing simulation platforms, which allow autonomous vehicle learning and testing for a greater population with minimal technical demand.**

Keywords*—*Autonomous Vehicles, Sensor Fusion, LiDAR Data Processing, Vehicle Dynamics, Neural Networks, Trajectory Prediction, Real Time Control Systems

I. INTRODUCTION

Autonomous cars are arguably the most revolutionary application of artificial intelligence and will probably be the greatest technological advance in the near future.[1]

Cars equipped with machine learning technologies are able to rapidly adapt to differing road conditions while continuously learning new experiences. Their onboard computers are able to make instantaneous decisions—oftentimes more quickly than experienced human drivers—by constantly analyzing data from cameras and sensors. The fundamental concepts behind autonomous and AI-powered vehicles mirror those deployed in other sectors employing artificial intelligence.[2]

Computer simulations of cars permit experimentation with an enormous variety of situations by running simulations in parallel on many machines at the same time. Nevertheless, this practice is often beset with difficulties in balancing between simulation accuracy and computational expense, as well as incompleteness and inaccuracies in the computer models themselves.[3]

With the improvement in computational power, it has become possible to train sophisticated neural networks that can understand a vehicle's environment and make decisions in real time.

For example, the Tesla Model S used an eye-specific processor (MobileEye EyeQ) dedicated to deep learning-based visual obstacle detection and avoidance.[1] Although such high-performance s Systems show the promise of deep neural networks in autonomous vehicles; they usually depend on specialized hardware and considerable computational resources. In contrast, our project is optimized to operate in a web browser, avoiding the necessity for dedicated installations or high-performance devices. Light in weight, browser-based, this technology becomes more accessible and easier to deploy in situations where hardware capabilities are limited, such as educational software or low-cost simulations.

Current systems such as CARLA and LGSVL offer sensor-rich, high-fidelity simulation environments but are technically and computationally intensive to install. While ideal for high-level testing, the technical nature of such systems may limit access, especially for education and early-test applications. Simple tools such as SIM.JS offer web-based simulation but are typically lacking in dynamic flexibility or intelligent decision-making capability. This is a

middle-of-the-road solution: A browser-native, standalone car simulator coded completely with JavaScript, HTML5, and the Canvas API. It uses a feedforward neural network trained with a mutation-based method and real-time sensor and decision rendering. The simulator can correctly simulate some of the most significant autonomous vehicle behaviors, such as lane following and obstacle avoidance, without any installations outside of the browser. The solution is meant to be an educational platform and a rapid prototyping platform for the testing of AI in autonomous systems.

II. LITERATURE REVIEW

Recent developments in autonomous vehicle research have increasingly stressed the significance of simulation frameworks for the development and testing of self-driving algorithms in controlled and reproducible environments. Research in [4] and [6] discuss architectures based on machine learning in which simulation is used as a methodology for modeling and validating self-driving behavior, such as lane following and obstacle avoidance. They generally work well for prototyping but are very often not sufficiently realistic for fine-grained behavioral testing. To tackle this, [10] proposes a working CNN model with simulation-based training of the network on image-based inputs before real-world deployment, thus requiring fewer expensive physical trials.

In the meantime, other researchers have tackled simulator environments in various ways-with one important application being simulation-based evaluation of vision systems without risking vehicle and human lives-with image processing applied to deep learning to design an end-to-end driving system. On the other hand, the work reported by [14] brings the refinement of injecting real human driving behavior into deep neural networks so that simulated data becomes more realistic and variable for training.

Structural considerations of ray casting architectures are outlined by [12], relevant in generating 3D environmental models in the simulation engines, so that spatial reasoning can be accurately supported for virtual driving agents. These works mark out simulation as being crucial, not just for development, but for creating a respectable, adaptable, and data-efficient self-driving system.

A. Summary of Findings

Several critical insights regarding the development of self-driving car simulation models have emerged from the reviewed literature:

- Limitations of High-Fidelity Simulation: [3] emphasized the need for realistic simulation in validating the safety of self-driving vehicles. Though as lean as possible, such high-fidelity simulators are computationally expensive and are therefore not best suited for students or less-equipped institutions.
- Deep Learning Models and Accessibility Problems: Such advanced models as DAVE-2 and [1] rely on realistic driving behavior with CNNs and large data. These are GPU, ROS, or high framework-dependent, and therefore they are not appropriate for large-scale educational application due to hardware and software complexity.

TABLE I. DETAILS REVIEW OF THE RELATED PAPERS.

S. No	Research Paper Name	Author(s)	Published Year	Summary	Pros	Cons
1	A Mental Simulation Approach for Learning Neural-Network Predictive Control (in Self-Driving Cars)	M. Da Lio, R. Donà, G. P. R. Papini, F. Biral and H. Svensson	2020	The paper presents a mental simulation approach to train neural network predictive control for self-driving cars in safe, realistic scenarios.	Enhances learning safety and efficiency by using simulations instead of risky real-world trials.	May not fully capture unpredictable real-world driving complexities and edge cases.
2	Toward a Framework for Highly	Koopman, P. and Wagner, M.	2018	This paper proposes a structured framework to validate the safety of	Improves validation efficiency by combining layered	Complex validation architecture

	Automated Vehicle Safety Validation			highly automated vehicles beyond traditional mileage-based testing.	simulations, test fidelity, and runtime monitoring for safety assurance.	demands significant design transparency, which is difficult with machine learning's opaque nature.
3	Review on self-driving cars using neural network architectures	Gudla, Rohan	2022	This paper reviews neural network-based methods for self-driving cars, focusing on lane detection, path planning, and traffic sign recognition.	Covers a broad range of deep learning approaches and real-world applications, offering a strong foundation for AV research.	Lacks original experimentation or implementation results beyond reviewing existing techniques
4	Real-time self-driving car navigation using deep neural network	Do, Truong-Dong	2018	The paper develops a Raspberry Pi-based self-driving RC car using a CNN model to predict steering angles from camera input.	Low-cost, real-time self-driving solution demonstrating practical end-to-end deep learning on embedded systems.	High camera latency and limited processing.
5	Improving the learning of self-driving vehicles based on real driving behavior using deep neural network techniques	Zaghari, Nayereh	2021	The paper proposes a deep learning framework (LSV-DNN) that mimics human driving behavior using real-world data and YOLOv3 detection.	Achieves high accuracy in steering, braking, and obstacle detection by learning directly from real driving behavior and road scenarios.	Requires substantial real-world data collection and preprocessing.
6	Convolutional Neural Network based Working Model of Self Driving Car - a Study	P. G. Chaitra, V. Deepthi, S. Gautami, H. M. Suraj and N. Kumar	2020	This paper explores a CNN-based prototype self-driving car using Raspberry Pi, Arduino, and deep learning for object and lane detection.	Cost-effective design using accessible hardware with real-time image processing for core driving tasks.	Limited processing power restricts performance.
7	Self-driving car to drive autonomously using image processing and deep learning	Garg, Nitika	2022	This paper proposes a deep learning-based self-driving car model using image processing, CNNs, and real-time simulation with Flask.	Achieves accurate autonomous driving tasks like lane changes and U-turns with low-cost simulation and real-time web communication.	Requires large training datasets and high computing power, which limits scalability and real-world deployment without further optimization.

8	Ray casting architectures for volume visualization	H. Ray, H. Pfister, D. Silver and T. A. Cook	1999	The paper surveys ray casting architectures for volume visualization, comparing performance, memory systems, and rendering techniques for large datasets	Provides detailed evaluation of specialized hardware solutions that enable high-performance volume rendering for scientific and medical visualization.	Hardware complexity and lack of flexibility limit widespread adoption compared to general-purpose GPU-based solutions.
9	Radar/Lidar sensor fusion for car-following on highways	D. Göhring, M. Wang, M. Schnürmacher and T. Ganjineh	2011	The paper presents a real-time radar/Lidar sensor fusion algorithm enabling autonomous highway car-following with improved velocity and distance estimation.	Combines strengths of radar and Lidar to enhance precision in velocity and distance estimation for safer autonomous driving.	Fusion system complexity increases due to unsynchronized sensors and additional computational overhead for real-time data processing.

- Simplified Browser-Based Models: Minimalistic models based on JavaScript and HTML5 in [11] provide basic simulations to learners. They are simple to execute in browsers with no extra configurations but provide little realism, flexibility, or neural learning capabilities provided on more advanced platforms.
- Shortage of Educational Guidance in New Paradigms: Research-grade experiments are the setting of simulation platforms like CARLA and TORCS and not intended for beginners. In all the literature reviewed, we find little emphasis on browser-compliant, user-friendly simulations with pedagogical simplicity and technical sophistication.

B. Gaps Identified

The work that we have surveyed reflects an enormous gap in creating light-weight, browser-compatible autonomous vehicle simulations that are computationally efficient and educationally intuitive. Everything that already exists centers on fidelity, realism, and performance, which need high-end GPUs, domain-specific libraries, and domain expertise—barriers to students and developers with limited access to high-end systems. Deep learning architectures employed in previous studies (e.g., LSTMs, CNNs) tend to be black boxes and complicated, and new-comers find it difficult to comprehend what is happening within. Even though training aids are available in a condensed form, they are not dynamic adaptive, do not simulate actual traffic movements, or expose underlying algorithmic principles such as path planning or neural control logic. Further, none of the models touched upon in the survey cross the explainability-interactivity chasm in browser environments. Educationally meaningful priorities such as visualizing neural network judgments, tunable training hyperparameters, and modularity with transparency are barely mentioned. There is also minimal investment in canvas-rendered simulations or JavaScript versions of neural networks, which may draw on deployment in browser environments with minimal installations. This provides a good potential to build a low-resource, web-based autonomous vehicle simulator for learning that is capable of simulating real-time AI decision-making, vehicle control, and neural network learning but is accessible, comprehensible, and modifiable.

III. PROPOSED SOLUTION

In order to fill the gap in the research that we have identified, we propose the development of a web-based, light-weight autonomous vehicle simulator with a foundation in education accessibility, modularity, and computational efficiency without compromising key AI and vehicle control

concepts. The platform will utilize web-based technologies that do not have to be installed or be unique in terms of hardware, enabling students and instructors to directly engage with core autonomous vehicle, machine learning, and control system concepts.

A. Lidar-Based Environment Perception

A simulated Lidar sensor that emits several rays within a predetermined angular range is installed in the vehicle. The normalized distance to the closest obstacle is returned by each ray. The neural network uses this collection of values as its sensory input.

B. Using a Feed Forward Neural Network (FFNN) to Make Decisions

The Lidar input is processed by an FFNN, which then decides on the steering and acceleration control commands for the car. An output layer with binary neurons for action decisions, one or more hidden layers for abstraction, and an input layer (Lidar rays) make up the network.

C. Evolutionary Learning through Mutation

The system trains the network using evolutionary algorithms rather than conventional supervised learning. Only the best-performing car, as determined by its travel distance and collision avoidance, is chosen after several agents are initialized with randomized networks. The following generation is created by mutations in its network, which encourages steady progress.

D. Collision Detection and Avoidance Algorithm

The simulation detects whether and where Lidar beams collide with obstacles by using ray-obstacle intersection algorithms based on 2D line geometry. Real-time obstacle avoidance is made possible by the direct influence of this data on neural responses

E. Real-Time Simulation on HTML5 Canvas

JavaScript and HTML5 Canvas are used to run the entire simulation in-browser and provide visual feedback. Real-time adjustments and visual observation of behaviour, learning, and training are made possible by the modular design.

IV. METHODOLOGY

A. System Design and architecture

- Algorithms for Avoiding Collisions: In order to identify nearby obstacles and dynamically modify the vehicle's trajectory, collision avoidance algorithms analyze sensor data. These algorithms enable real-time responses to environmental changes by preventing crashes through the use of geometric analysis and safety margins. By anticipating possible collisions and adjusting steering or braking accordingly, this guarantees safe and easy navigation.
- The Lidar Sensor: The Lidar sensor emits virtual rays in various directions to mimic scanning the environment. It calculates the separations between adjacent objects and converts them into numerical inputs for the neural network. This gives the vehicle spatial awareness, which is essential for safe navigation and decision-making because it enables precise perception of road edges, other cars, and obstacles.
- Neural Network Feed Forward: The car's decision-making engine is a feed forward neural network. After processing inputs from sensors (such as Lidar) through several hidden layers, it outputs control signals for the throttle and steering. By learning the best driving techniques through training, the network imitates cognitive behaviour and allows autonomous control without the need for hard-coded rules.
- AI-Powered Control Automobile: AI control converts predictions into actions that resemble those in the real world by integrating neural network outputs with the mechanical simulation of the vehicle. It uses neural outputs to translate movement into acceleration, braking, and steering. In order to maintain responsive control and a realistic driving experience, this layer makes sure that the AI's decisions are mirrored in the behaviour of the vehicle.
- Training and Choosing the Best Automobile: During training, randomized neural networks are used to simulate several cars in identical settings. Performance indicators such as survival time or distance travelled are assessed. In subsequent generations, the top-performing networks are chosen, replicated, and altered.

B. Mathematics and physics Formulation

Kinematic motion, sensor raycasting, neural network control, and collision detection are some of the fundamental mathematical and physical models

utilized in the self-driving car simulation that are presented in this section.

- Car Movement (Kinematics): Based on its velocity (v), orientation (θ), and friction (f), the car moves in two dimensions. After every frame, its updated position is determined as follows:

$$x' = x - sin(\theta) \cdot v$$
$$y' = y - cos(\theta) \cdot v$$

Frictional deceleration is modeled as:

$$v = v - f$$

For ease of simulation, these equations define a simplified kinematic model that does not include lateral slip or turning radius.

- Sensor Raycasting (Lidar Simulation): A simulated Lidar sensor that sends out n rays over a field of view s is installed in every vehicle. Linear interpolation (lerp) is used to calculate each ray's direction

$$angle_i = lerp\left(\frac{s}{2}, -\frac{s}{2}, \frac{i}{n-1}\right)$$

By extending from the car's position (x,y) in the ray direction for length L, one can find the endpoint of each ray:

$$x_{end} = x - \sin(angle_i) \cdot L$$
$$y_{end} = y - \cos(angle_i) \cdot L$$

Geometric intersection tests are used to detect obstacles at these endpoints.

- Neural Network Control (Feed Forward): A feed-forward neural network comprising input layers (sensor data), hidden layers, and output layers (steering and throttle decisions) governs the vehicle. Weighted Sum for each neuron:

$$z = \sum_{i=1}^{n} x_i \cdot w_i + b$$

  - x_i are inputs
  - w_i are weights
  - b is the bias

Activation Function (Binary Step):

$$activation(z) = \begin{cases} 1 \; if \; z > 0 \\ 0 \; otherwise \end{cases}$$

- Intersection Detection (Collision Logic): A line intersection test is used to determine whether a Lidar ray crosses an obstruction (represented as a line segment from(x_3, y_3) to (x_4, y_4)). The collision is determined by calculating the point of intersection and comparing it with the ray's length using vector math or parametric equations. Real-time obstacle detection and decision-making are made possible by this method.

C. *System Requirements*

- Frontend Requirements: It is necessary to have a contemporary web browser (such as Chrome or Firefox) that supports HTML5, CSS3, and JavaScript with the Canvas API. This guarantees responsive performance and real-time rendering of roads, cars, and traffic environments right in the browser.
- AI model: The AI model uses a lightweight neural network for decision-making (lane detection, obstacle avoidance, etc.) and is constructed with vanilla JavaScript. It is effective for in-browser execution on low-resource systems because it does not require external libraries or frameworks.

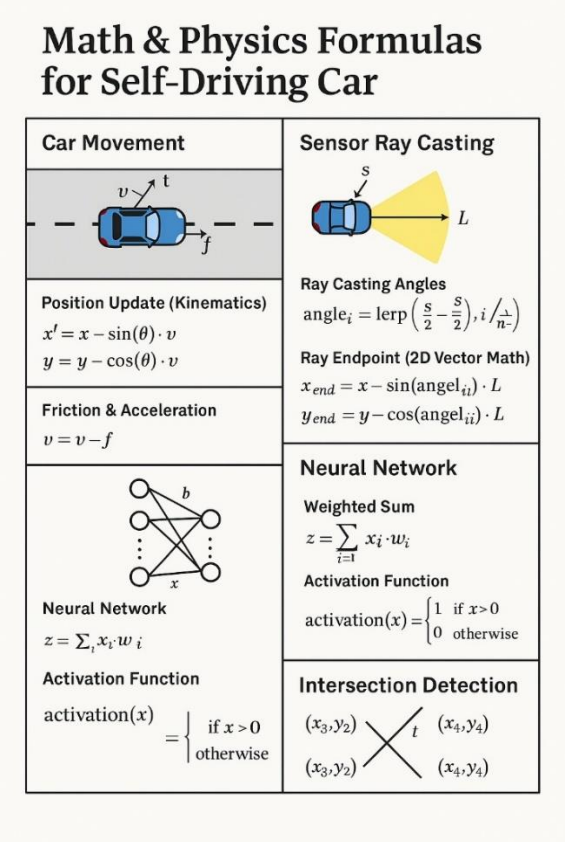

Fig 1. Math and Physics Formulas for Self-Driving Cars.

V. RESULTS AND DISCUSSION

We tested the model in three distinct environments—no obstacles, static obstacles, and dynamic obstacles—to assess how well the AI-driven self-driving car simulation performed. To guarantee evaluation consistency, every test was run with the same simulation parameters.

A. *No Obstacle Environment*

The road was clear and unobstructed in this

baseline situation. The vehicle continuously stayed inside lane lines and kept a smooth trajectory. The neural network reached the goal with 100% success when it used optimal path planning. This scenario illustrated the AI's basic lane discipline and route tracking capabilities.

B. Static Obstacle Environment:

There were fixed roadblocks at predetermined locations as part of this setup. By applying its learnt decision-making logic, the AI was able to recognize and avoid obstacles. With sporadic small deviations when obstacles were positioned close to turns, the success rate was roughly 92%. Based on sensory input, the model demonstrated obvious awareness and avoidance behaviour.

C. Dynamic Obstacle Environment:

In this intricate scenario, moving obstacles were used to mimic pedestrians or traffic. Real-time prediction and rerouting presented difficulties for the AI. In spite of this, it had a 78% success rate in staying on course and avoiding collisions. Rapid intersections between several dynamic agents were found to result in performance drops. Results may be improved by augmenting the model with temporal data or recurrent learning.

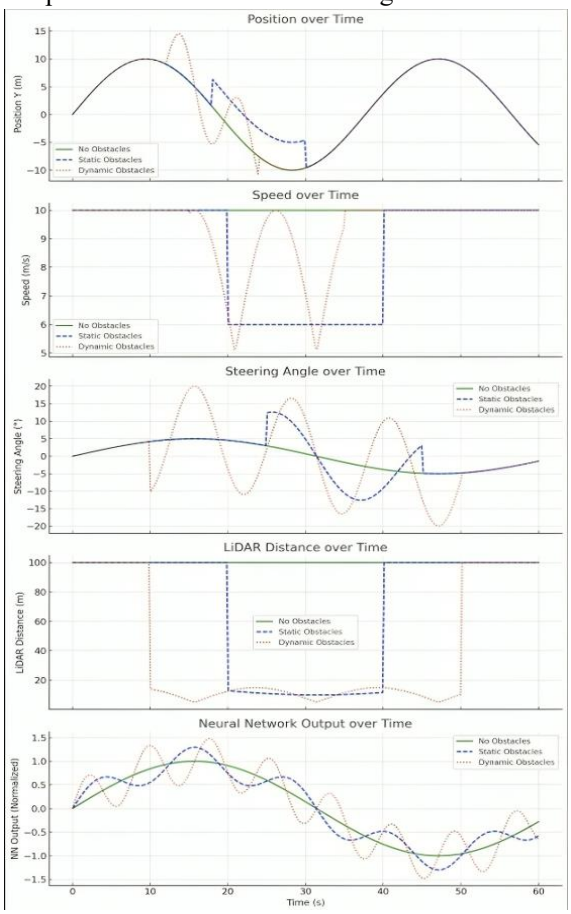

Fig 2. Data collection analysis and graph of various factors

VI. CONCLUSION

This project's self-driving car simulation effectively illustrates the basic ideas underlying artificial intelligence-based autonomous vehicle navigation. The simulation allows a virtual car to sense its surroundings, make decisions in real time, and adjust through mutation-based learning by fusing neural networks with sensor-based input. Understanding how AI can be trained to drive safely and effectively, avoiding obstacles and adapting to changing traffic conditions, is made easier with the help of this interactive model.

This simulation allowed us to see how reinforcement learning methods can be used to simulate driving behaviour similar to that of humans. A straightforward yet effective method of model persistence and iteration is demonstrated by the capacity to store and reuse optimized neural networks using local storage. The foundation for more intricate and realistic applications in the field of autonomous driving is laid by this simulation, even though it runs in a controlled 2D environment. More complex obstacle scenarios, dynamic traffic regulations, or integration with real-world datasets to improve accuracy and realism are possible future developments.

All things considered, this project offers a useful and instructive platform for investigating the fundamentals of artificial intelligence in automotive applications and emphasizes the expanding role of simulation in creating intelligent, safe transportation systems.

REFERENCES

[1] T. -D. Do, M. -T. Duong, Q. -V. Dang and M. -H. Le, "Real-Time Self-Driving Car Navigation Using Deep Neural Network," 2018 4th International Conference on Green Technology and Sustainable Development (GTSD), Ho Chi Minh City, Vietnam, 2018, pp. 7-12.

[2] Gudla, Rohan, Vijay Shankar Telidevulapalli, Jayasree Sarada Kota, and Gayathri Mandha. "Review on self-driving cars using neural network architectures." World Journal of Advanced Research and Reviews 16, no. 2 (2022): 736-746.

[3] Koopman, P. and Wagner, M., "Toward a Framework for Highly Automated Vehicle Safety Validation," SAE Technical Paper 2018-01-1071, 2018.
[4] Soni, Abhishek, et al. "Design of a machine learning-based self-driving car." Machine Learning for Robotics Applications (2021): 139-151.
[5] Sellat, Qusay, et al. "Intelligent Semantic Segmentation for Self-Driving Vehicles Using Deep Learning." Computational Intelligence and Neuroscience 2022.1 (2022): 6390260.
[6] Sharma, Chirag, S. Bharathiraja, and G. Anusooya. "Self-driving car using deep learning technique." International Journal of Engineering and Technical Research 9.06 (2020).
[7] S. Ramos, S. Gehrig, P. Pinggera, U. Franke and C. Rother, "Detecting unexpected obstacles for self-driving cars: Fusing deep learning and geometric modeling," 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 2017, pp. 1025-1032.
[8] Simhambhatla, Ramesh, et al. "Self-driving cars: Evaluation of deep learning techniques for object detection in different driving conditions." SMU Data Science Review 2.1 (2019): 23.
[9] Holzhüter, Hanno, et al. "Technical concepts of automotive LiDAR sensors: a review." Optical Engineering 62.3 (2023): 031213-031213.
[10] P. G. Chaitra, V. Deepthi, S. Gautami, H. M. Suraj and N. Kumar, "Convolutional Neural Network based Working Model of Self Driving Car - a Study," 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2020, pp. 645-650
[11] Garg, Nitika, et al. "Self-driving car to drive autonomously using image processing and deep learning." International Journal of Research in Engineering, Science and Management 5.1 (2022): 125-132.
[12] H. Ray, H. Pfister, D. Silver and T. A. Cook, "Ray casting architectures for volume visualization," in IEEE Transactions on Visualization and Computer Graphics, vol. 5, no. 3, pp. 210-223, July-Sept. 1999.
[13] D. Göhring, M. Wang, M. Schnürmacher and T. Ganjineh, "Radar/Lidar sensor fusion for car-following on highways," The 5th International Conference on Automation, Robotics and Applications, Wellington, New Zealand, 2011, pp. 407-412.
[14] Zaghari, Nayereh, et al. "Improving the learning of self-driving vehicles based on real driving behavior using deep neural network techniques." The Journal of Supercomputing 77 (2021): 3752-3794.
[15] Do, Truong-Dong, et al. "Real-time self-driving car navigation using deep neural network." 2018 4th International Conference on Green Technology and Sustainable Development (GTSD). IEEE, 2018.
[16] M. Da Lio, R. Donà, G. P. R. Papini, F. Biral and H. Svensson, "A Mental Simulation Approach for Learning Neural-Network Predictive Control (in Self-Driving Cars)," in IEEE Access, vol. 8, pp. 192041-192064, 2020.