SYNTIA – Systems and Technologies in Artificial Intelligence Applications.
E-ISSN: XXXX-XXXX. Volume 1 Issue 1 (December 2025). Page: 19-29.
DOI: 10.1234/syntia.v1i1.2025.xxx.

Implementation of Neural Networks for Car Navigation System Development in a 2D Car Game Simulation Using Genetic Algorithms

Erick Wicaksono[1*], Yuri Yudhaswana Joefri[2]

[1*]*Information Technology Department, Engineering Faculty, Universitas Tadulako, Indonesia. E-mail: feininformatika17@gmail.com*
[2] *Information Technology Department, Engineering Faculty, Universitas Tadulako, Indonesia. E-mail: yudhaswana@kde.cs.tut.ac.jp*

Abstract: This study proposes the development of an autonomous car navigation system in a two-dimensional (2D) car game simulation using Artificial Neural Networks (ANNs) optimized through Genetic Algorithms (GAs). The objective of this research is to create an adaptive and intelligent navigation system capable of learning various track conditions and avoiding obstacles effectively. In the proposed approach, neural networks are employed as decision-making models based on sensor inputs, while genetic algorithms are used to optimize network weights through evolutionary processes. Each generation consists of 20 autonomous cars, and their performance is evaluated using a fitness function based on navigation accuracy, obstacle avoidance, and distance traveled along the track. The best-performing individuals are selected and evolved to produce improved generations. Experimental results demonstrate a significant improvement in navigation accuracy and adaptability across successive generations, particularly on tracks with higher difficulty levels. The visualization of neural networks, sensors, and user interface components facilitates performance monitoring and enhances the interpretability of decision-making processes. The Unity game engine is utilized as the simulation platform, integrating GameObjects, a physics engine, and visualization tools to provide a realistic and stable experimental environment. Overall, the results confirm that the integration of neural networks and genetic algorithms is effective for developing adaptive and efficient autonomous vehicle navigation systems in 2D game simulations.

Keywords : Neural Networks; Genetic Algorithms; Autonomous Navigation; 2D Car Simulation; Unity.

Received: November 03, 2025 | Revised: November 27, 2025 | Accepted: December 17, 2025 | Published: December 31, 2025

1. Introduction

In recent decades, video games have evolved into a global phenomenon, reaching unprecedented levels of popularity. The rapid growth of the video game industry is driven by continuous technological advancements, including increased computing capabilities, more realistic graphics, and immersive gameplay mechanics [1]. The success of a video game is determined not only by its visual presentation and narrative elements but also by the effective implementation of artificial intelligence (AI), which plays a crucial role in shaping player experience and interaction [2].

Artificial intelligence enables non-player characters and autonomous agents in games to exhibit adaptive and intelligent behavior. According to H. A. Simon, artificial intelligence is defined as the study of machines capable of performing tasks that typically require human intelligence [3]. In the context of car simulation games, AI contributes significantly to the development of intelligent navigation systems. These systems are no longer merely supplementary features but have become essential components in creating dynamic, realistic, and challenging gameplay experiences, particularly in two-dimensional (2D) car simulation games [4].

In 2D car games, players expect autonomous vehicles to recognize track layouts, navigate efficiently, and avoid obstacles with fast and adaptive responses. The navigation system must be capable of handling various track configurations and making real-time decisions to reach target destinations. Neural networks have emerged as a promising approach to addressing these challenges due to their ability to learn from data and adapt behavior based on experience [5]. As computational models inspired by the human brain, neural

networks have been widely applied in control systems and autonomous navigation tasks [6]. In this study, neural networks are employed as the primary control mechanism for vehicle navigation within a 2D game environment.

Several previous studies have addressed similar research problems. Mahajan and Kaur demonstrated that the integration of genetic algorithms with artificial neural networks can improve training effectiveness and optimize network structures [7]. Meanwhile, Manangka, Suprijono, and Nurcipto developed a driverless car prototype utilizing camera-based sensors and artificial neural networks for lane pattern recognition, highlighting the applicability of learning-based approaches for autonomous navigation [8]. These studies provide a strong foundation and motivate further research on intelligent navigation systems using neural networks in game-based environments.

2. Methodology

The type of research used is research and development (R&D), which aims to produce a product, namely a car navigation system in a 2D car simulation game using Neural Networks and Genetic Algorithms.

2.1. Research Type

This study employs a Research and Development (R&D) approach aimed at designing, implementing, and evaluating an intelligent car navigation system for a two-dimensional (2D) car simulation game. The proposed system integrates Artificial Neural Networks (ANNs) as the decision-making model and Genetic Algorithms (GAs) as the optimization and training mechanism. The R&D approach is suitable for developing and validating computational systems that require iterative design, experimentation, and performance evaluation [9], [10].

2.2. Research Materials

The primary materials used in this research are derived from an open-source project developed by Samuel Arzt, a software developer and educational content creator. The project provides a simulation framework and baseline implementation for car navigation using Neural Networks and Genetic Algorithms. Such open-source simulation environments are widely used in AI research due to their flexibility, reproducibility, and suitability for experimental evaluation [11], [12]. The project was adapted and extended to meet the objectives of this study.

2.3. Research Design

This research adopts an experimental R&D design focused on developing an automated navigation system for a 2D car simulation game. The research process consists of several sequential stages: problem identification, system modeling and design, implementation of neural network and genetic algorithm components, system testing, and performance evaluation. Experimental evaluation is essential to measure the effectiveness of AI-based navigation systems under different environmental conditions and track configurations [13], [14].

2.4. Data Types and Sources

This study utilizes both primary and secondary data. Primary data are obtained through direct experimentation within a 2D car simulation environment developed using the Unity game engine. These data include vehicle movement behavior, collision events, and navigation performance metrics. Secondary data are collected from scientific literature, including peer-reviewed journals, conference proceedings, and books related to artificial intelligence, neural networks, genetic algorithms, and autonomous navigation systems [15], [16].

2.5. Data Collection Techniques

Data was collected through several techniques, as follows :

1. Experimentation: The navigation system is tested in various simulated track scenarios to evaluate its responsiveness, adaptability, and robustness under different conditions.
2. Observation: The behavior of autonomous vehicles is observed to identify navigation errors, inefficiencies, and learning limitations of the neural network model.
3. Documentation: Experimental results, performance logs, and visual outputs are systematically recorded to support analysis and reproducibility.

2.5. Data Analysis Methods

The collected data are analyzed using both quantitative and qualitative methods. Quantitative analysis is applied to evaluate system performance using metrics such as average speed, navigation accuracy, learning convergence, and number of collisions. Qualitative analysis is conducted to examine behavioral patterns and decision-making processes of the autonomous vehicle in different simulation scenarios. The analysis results are used to refine the neural network architecture and optimize genetic algorithm parameters to enhance overall navigation performance.

2.6. System Development Method

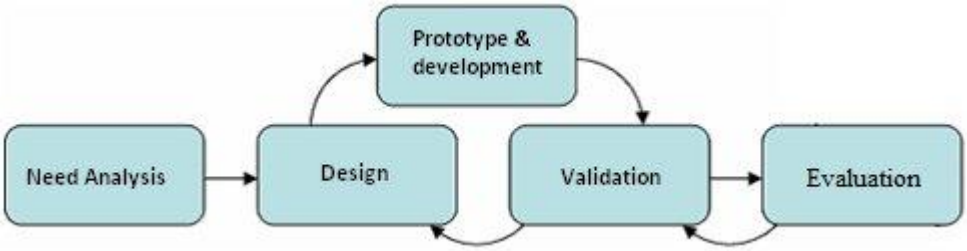

Figure 1. Rapid Game Prototyping Process

The system development method used in this research is the iterative method development, which involves iterative design, implementation, testing, and refinement. Rapid Game Prototyping, an approach derived from a modification of the extreme programming software development method, is also applied.

2.7. Research Stages and Flow

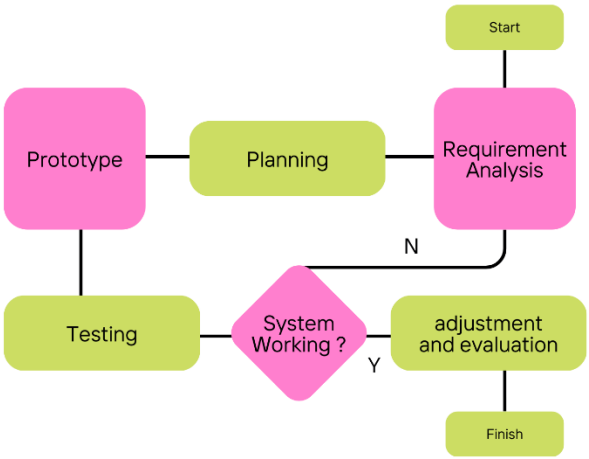

Figure 2. Research Stages

The diagram illustrates an iterative system development process that starts with identifying needs (requirements analysis), followed by design, prototype creation, and testing. After testing, the system is evaluated to determine whether it works properly; if it does not, the process returns to the needs analysis stage for improvement. If the system works well, it proceeds to adjustment and evaluation, and then the process is completed.

3. Results and Discussion

In this simulation, each car is controlled by a neural network consisting of five input nodes (front sensors), two output nodes (turn and engine), and two hidden layers (with four and three nodes). The sensors measure the distance to obstacles and send the data to the neural network. This data is processed through the hidden layers using activation functions, producing an output that controls the direction and speed of the car. The weights and biases in the neural network regulate the signal strength and are optimized using a genetic algorithm This algorithm optimizes the neural network's weights through evolution. Each generation begins with 20 random cars. After all cars die (colliding with obstacles), the two cars with the highest fitness values are selected as "parents." Their weights are combined to produce a new generation of cars, with minimal mutations for genetic variation. This process repeats, improving the cars' performance from generation to generation.

3.1. Implementation of Neural Networks and Genetic Algorithms

This research focuses on the implementation of artificial neural networks (NA) and genetic algorithms (GAR) in developing a car navigation system in a 2D car simulation game. The NAR implementation involves a network configuration with 5 input nodes (front sensors), 2 output nodes (turn and engine), and 2 hidden layers with 4 and 3 nodes, respectively. These sensors measure the distance to obstacles in front of the car and provide input data to the NAR.

Genetic algorithms are used to optimize artificial neural network parameters through selection, crossover, and mutation processes. This algorithm simulates the evolutionary process to produce better neural network generations with each iteration, with the goal of improving the car's navigation capabilities in various game scenarios.

3.2. Discussion

3.2.1. Software and Platform

This research uses Unity and Visual Studio as the primary tools. Unity, as a game engine, provides a comprehensive development environment for 2D game simulations, including a physics engine and scripting API. Visual Studio is used as an IDE to write, edit, and manage code in Unity. This combination enables the development of an efficient navigation system and supports the iteration and optimization of artificial intelligence algorithms.

3.2.2. Unity Assets and Components

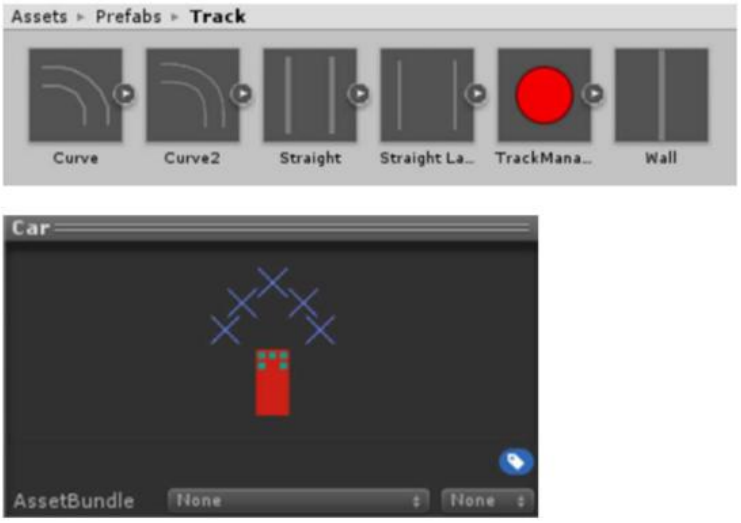

Figure 3. Tracks and 2D Car Models

Figure 3 shows the main assets used in the Unity-based 2D Car Game simulation, namely the track prefab and the 2D car model as an autonomous agent. The track prefab consists of modular elements such as straight lines, curves, walls, and a track manager that allows the creation of various route variations to test the adaptability of the navigation system. Meanwhile, the 2D car model is equipped with virtual sensors that function to detect track boundaries and obstacles, where the sensor data becomes input for the neural network in making navigation decisions. This combination of assets supports the implementation and evaluation of neural networks optimized using genetic algorithms, and utilizes Unity as a reliable simulation environment through the integration of GameObjects, Physics Engine, and visualization components.

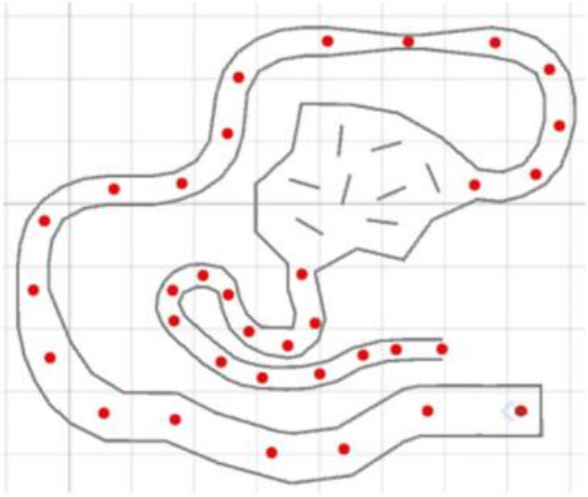

Figure 4. Track Scenes

The figure illustrates a 2D track representation used as the simulation environment for autonomous car navigation, where the gray lines indicate track boundaries, the central area acts as an obstacle or restricted zone, and the red dots represent checkpoints or waypoints. These checkpoints are used to evaluate vehicle performance in following the track, avoiding collisions, and determining fitness values within the genetic algorithm process. Featuring sharp turns, narrow paths, and winding routes, the track is designed to test the adaptability of the neural network in making sensor-based navigation decisions under complex and dynamic conditions.

Figure 5. Sprites Asset

The figure shows a collection of 2D sprites in Unity used as visual elements in the autonomous car game simulation. The background sprite serves as the track's backdrop, the car represents the vehicle controlled by the neural network–based navigation system, and the wall sprite defines track boundaries or obstacles. The sensor sprite visualizes the direction and range of the vehicle's sensors for environmental detection. In addition, the best and secondBest sprites are used as visual indicators to highlight the top-performing vehicles in each generation, while the cross sprite represents failure or collision states. Together, these sprites support the visualization of evaluation, learning, and selection processes in the genetic algorithm–based 2D simulation.

3.2.3. Simulation Visualtion and Control

3.2.3.1 Generation initialization

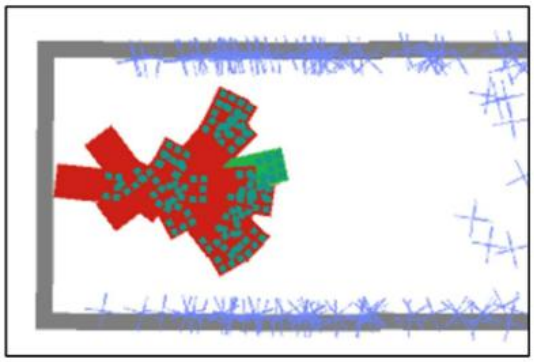

Figure 6. Generation Initialization

The figure presents a visualization of the autonomous car navigation process in a 2D simulation, where the red object represents the vehicle body, the green area indicates the selected direction or action produced by the neural network's decision, and the small dots illustrate active neurons or network weights. The blue lines surrounding the track visualize the vehicle's sensors used to measure distances to walls and obstacles. This condition demonstrates how the vehicle processes sensor information in real time to determine its movement direction, while also illustrating the learning and decision-making mechanism optimized through a genetic algorithm within a constrained track environment.

3.2.3.2 Visual Neural Network

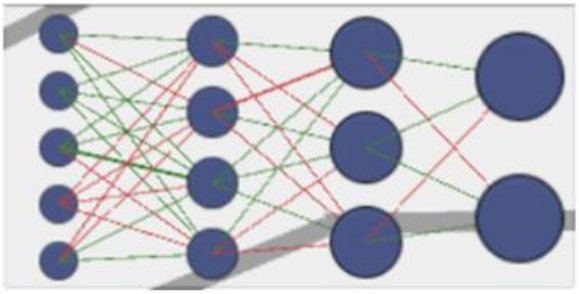

Figure 7. Visual Neural Network

This figure shows a visualization of a 2D vehicle and its sensor system used in the autonomous navigation simulation. The red object represents the car, while the blue lines and markers indicate sensor rays used to detect track boundaries and surrounding obstacles. The dots around the vehicle illustrate sensor readings that serve as inputs for the neural network to determine movement and steering decisions. This visualization helps illustrate how the vehicle interacts with its environment and how sensor information is processed in real time within the learning-based navigation system.

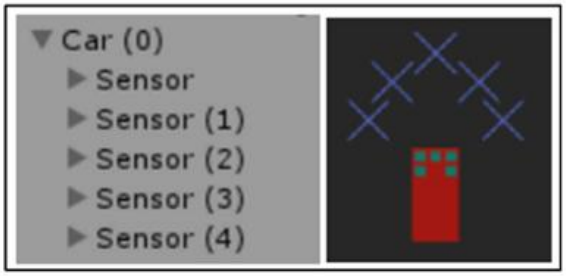

Figure 8. Sensoring Mechanism

Figure 8 shows the use of five sensors positioned at the front of the vehicle to facilitate the detection of environmental conditions.

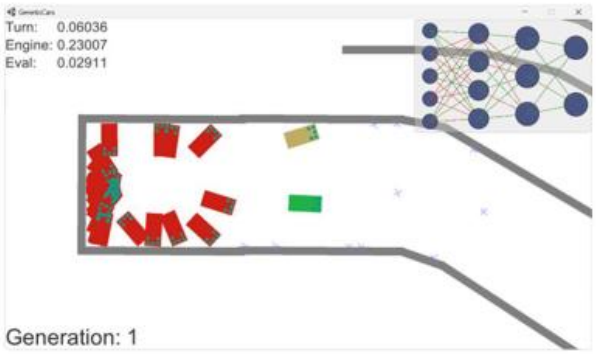

Figure 8. Simulation Testing

The figure illustrates the autonomous car navigation simulation in the first generation (Generation: 1), where multiple 2D vehicles are tested on the same track environment. Red vehicles indicate agents that failed or collided, while the green vehicle represents the best-performing agent in that generation. The upper-right section shows a visualization of the neural network architecture used for sensor-based decision making. Parameters such as turn, engine, and evaluation reflect the neural network outputs and the vehicle's performance assessment. This visualization supports monitoring the learning process, selection mechanism, and evolution of neural network weights optimized through the genetic algorithm.

3.2. Simulation Implementation Testing

3.2.1. Functionality Testing

Functional testing aims to ensure that all key components and features of the autonomous vehicle navigation system operate according to established specifications.

Table 1. Functionality Testing

Test Cases and Results			
Test Type	**Description**	**Parameters**	**Result**
Environment Setup Testing	Configuring tracks and vehicles in the simulation environment.	Tracks: 4 (Track 1, Track 2, Track 3, Track 4); Vehicles: 20	✓ Tracks and vehicles were successfully initialized.
Initial Generation Testing	Ensuring vehicles are generated with correct neural networks and sensors.	Neural Networks and Sensors	✓ All vehicles were successfully initialized with functional neural networks.
Sensor Operation	Verifying sensor accuracy in measuring distance to obstacles.	Virtual Sensors	✓ Sensors accurately measured distances to obstacles.
Neural Network Decision-Making	Verifying that the neural network makes decisions based on sensor input.	Neural Networks	✓ Neural networks produced relevant decision outputs.
Vehicle Movement	Observing vehicle movement based on neural network decisions.	Neural Networks, Physics Engine	✓ Vehicles moved according to neural network decisions.
Collision Detection	Ensuring collisions between vehicles and obstacles are correctly detected.	Virtual Sensors, Physics Engine	✓ All collisions were accurately detected and colliding vehicles were stopped.
Fitness Evaluation	Measuring the fitness value of each vehicle on the track.	Fitness Calculation	✓ Fitness values for all vehicles were calculated.

3.2.2. Perform Testing

Performance testing was conducted to ensure that the neural network and genetic algorithm-based navigation system operated efficiently and effectively.

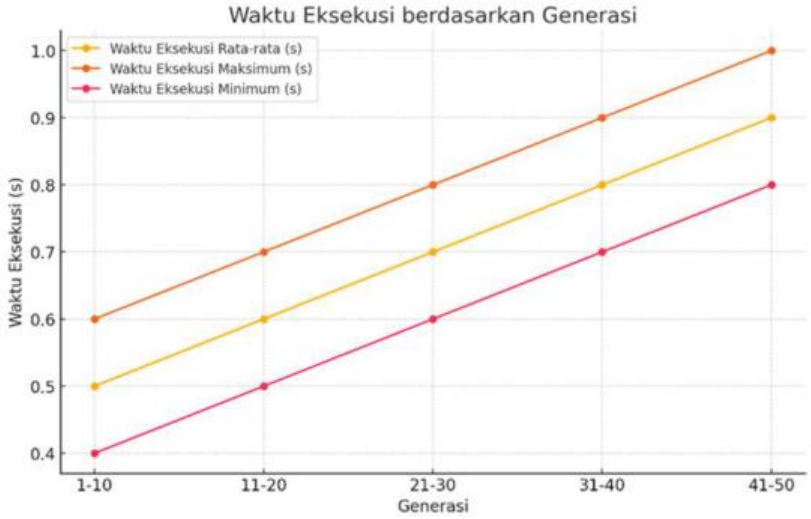

Figure 9. Execution Time by Generation

The figure presents a graph of execution time based on generations in the training process of the autonomous car navigation system. The graph shows three parameters: minimum, average, and maximum execution time, all of which tend to increase as the number of generations grows. This trend indicates that the computational complexity and individual evaluations in the genetic algorithm become higher in later generations, due to the processes of selection, reproduction, and optimization of neural network weights to achieve improved navigation performance.

3.2.3. Accuration Testing

Accuracy testing aims to assess the extent to which an automated navigation system can reach its destination without encountering obstacles. This testing involves several simulated scenarios with varying levels of difficulty to evaluate the system's ability to detect and avoid obstacles and make appropriate navigation decisions.

Tabel 2. Accuration Testing

Generation	Track 1 (Easy)	Track 2 (Medium)	Track 3 (Hard)
1-10	30%	20%	10%
11-20	40%	25%	15%
21-30	55%	35%	20%
31-40	60%	40%	25%
41-50	70%	50%	30%
51+	100%	60%	40%

3.2.3. Scalability Testing

Scalability testing aims to ensure that the car simulation using neural networks and genetic algorithms in Unity can run smoothly and efficiently on various hardware configurations and scales.

3.2.3.1 Unity Profiler

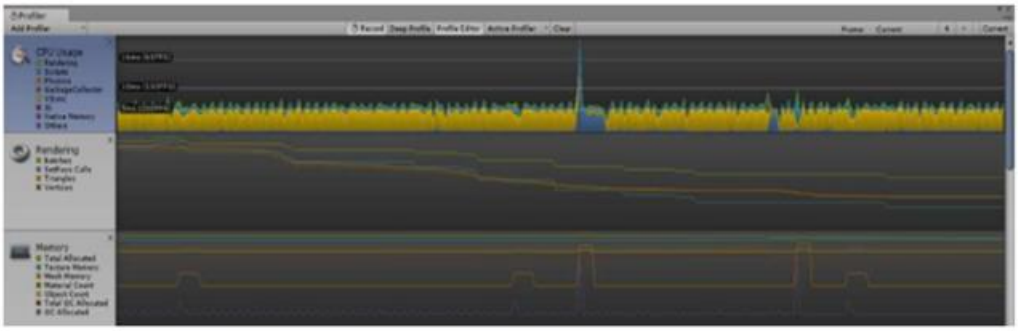

Figure 10. Unity Profiler

The figure shows the Unity Profiler output used to monitor system performance during the autonomous car navigation simulation. The graph presents resource usage such as CPU, rendering, physics, and memory over time, allowing developers to identify computational load spikes occurring during neural network evaluation and genetic algorithm evolution. This information is essential to ensure that the simulation runs stably and efficiently while handling increased computational complexity across generations without significant performance degradation.

3.2.3.2 Scalability Testing Graph

The Figure 11 presents four performance graphs showing how increasing the number of cars affects FPS, CPU usage, GPU usage, and memory usage on high-end and low-end PCs. As the number of cars increases, FPS decreases on both systems, with a sharper drop on the low-end PC. At the same time, CPU, GPU, and memory usage increase steadily, where the low-end PC consistently shows higher resource usage than the high-end PC, indicating lower efficiency and greater performance strain as the workload grows.

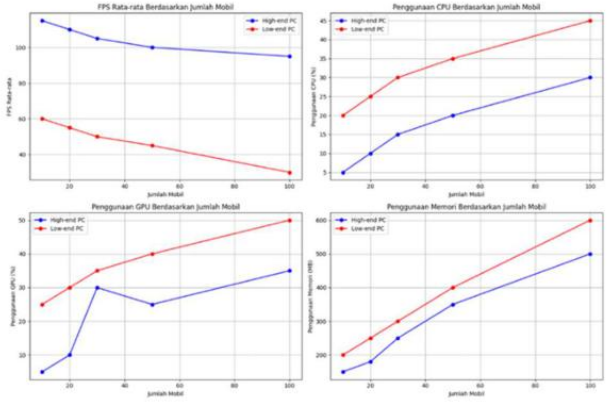

Figure 11. Scalability Testing Graph

4. Conclusion

This study successfully implemented an autonomous car navigation system in a 2D game simulation using Artificial Neural Networks optimized by Genetic Algorithms. The proposed approach enables vehicles to make sensor-based navigation decisions and progressively improve their performance through evolutionary learning across generations. Experimental results show that the system demonstrates increasing navigation accuracy, improved obstacle avoidance, and better adaptability to varying track difficulty levels as the number of generations increases.

The integration of neural networks as decision-making controllers and genetic algorithms as optimization mechanisms proved effective in generating robust navigation behavior without explicit rule-based programming. Functional, performance, accuracy, and scalability testing results indicate that the system operates reliably within the Unity simulation environment and maintains acceptable performance under increasing computational loads.

Furthermore, the visualization of neural networks, sensors, and evolutionary processes enhances system interpretability and supports effective evaluation of learning behavior. Overall, this research confirms that the combination of neural networks and genetic algorithms is a viable and efficient approach for developing adaptive autonomous navigation systems in 2D car game simulations. Future work may focus on extending the model to more complex environments, dynamic obstacles, and real-time learning mechanisms to further enhance system realism and performance.

Author Contributions Statement

Erick Wicaksono was responsible for the conceptualization of the research, system design, implementation of neural networks and genetic algorithms, development of the 2D car game simulation, data collection, and initial manuscript drafting. Yuri Yudhaswana Joefrie contributed to the research methodology, experimental design, performance analysis, result interpretation, and critical revision of the manuscript. Both authors have read and approved the final version of the manuscript and agree to be responsible for all aspects of the work.

References

[1] E. Goh, O. Al-Tabbaa, and Z. Khan, "Unravelling the complexity of the Video Game Industry: An integrative framework and future research directions," Telematics and Informatics Reports, vol. 12, Art. no. 100100, Dec. 2023, doi: 10.1016/j.teler.2023.100100.

[2] A. del Bosque, G. Lampropoulos, and D. Vergara, "The Role of Artificial Intelligence in Gaming," Applied Sciences, vol. 15, no. 23, Art. no. 12358, 2025, doi: 10.3390/app152312358.

[3] H. A. Simon, "Artificial intelligence: an empirical science," Artificial Intelligence, vol. 77, no. 1, pp. 95–127, Aug. 1995, doi: 10.1016/0004-3702(95)00039-H.

[4] J. Togelius and S. M. Lucas, "Evolving controllers for simulated car racing," in Proc. 2005 IEEE Congress on Evolutionary Computation (CEC 2005), 2005, vol. 2, pp. 1906–1913, doi: 10.1109/CEC.2005.1554920.

[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[6] Y. Chen, C. Cheng, Y. Zhang, X. Li, and L. Sun, "A Neural Network-Based Navigation Approach for Autonomous Mobile Robot Systems," Applied Sciences, vol. 12, no. 15, Art. no. 7796, 2022, doi: 10.3390/app12157796.

[7] R. Mahajan and G. Kaur, "Neural Networks using Genetic Algorithms," International Journal of Computer Applications, vol. 77, no. 14, pp. 6–11, Sep. 2013, doi: 10.5120/13549-1153.

[8] L. R. Manangka, H. Suprijono, and D. Nurcipto, "Pengenalan Pola Lintasan Berbasis Neural Network Pada Prototype Self-Driving Car," Elektrika, vol. 12, no. 2, pp. 67–72, 2020.

[9] R. Mahajan and G. Kaur, "Neural Networks Using Genetic Algorithms," International Journal of Computer Applications, vol. 77, no. 14, pp. 6–11, 2013.

[10] J. H. Holland, Adaptation in Natural and Artificial Systems, Ann Arbor, MI, USA: University of Michigan Press, 1975.

[11] J. Togelius, S. M. Lucas, H. Richter, and J. H. Chiang, "Evolving Controllers for Simulated Car Racing," in Proc. IEEE Congress on Evolutionary Computation (CEC), 2005, pp. 1906–1913.

[12] S. Arzt, "Neural Network Car Simulation Project," GitHub Repository, 2020. [Online]. Available: https://github.com/samuelarzt

[13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[14] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 4th ed., Pearson, 2021.

[15] C. Bierwirth, "A Generalized Permutation Approach to Job Shop Scheduling with Genetic Algorithms," OR Spectrum, vol. 17, no. 2–3, pp. 87–92, 1995.

[16] Y. Chen, C. Cheng, Y. Zhang, X. Li, and L. Sun, "A Neural Network-Based Navigation Approach for Autonomous Mobile Robot Systems," Applied Sciences, vol. 12, no. 15, Art. no. 7796, 2022.