World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(RESEARCH ARTICLE)

Check for updates

Parking system for autonomous vehicles using Q-Learning

K. Swetha Sailaja, Ranjith Reddavena *, Ram Teja Gande and Yashwanth Bathuka

Department Of CSE (Artificial Intelligence and Machine Learning), ACE Engineering College Hyderabad, Telangana, India.

World Journal of Advanced Research and Reviews, 2026, 30(01), 1196-1205

Publication history: Received on 26 February 2026; revised on 07 April 2026; accepted on 10 April 2026

Article DOI: https://doi.org/10.30574/wjarr.2026.30.1.0864

Abstract

This project focuses on developing a system that enables a vehicle to park automatically without human intervention. It uses a reinforcement learning technique called Q-learning to simulate a real-world parking environment. The system considers factors such as the position of the vehicle, its distance from the parking slot, and the presence of obstacles in the surroundings. Based on this information, the agent decides the best possible action, such as moving forward, turning, or adjusting its position.

The system also provides real-time visualization, allowing users to observe how the vehicle learns and performs the parking task step by step. The main objective of this project is to make parking more efficient and safe, while demonstrating how reinforcement learning can be applied to solve practical problems in intelligent transportation systems.

Keywords: Autonomous Parking; Q-Learning; Reinforcement Learning; Intelligent Systems; Simulation

1. Introduction

Traditional parking systems use set paths and rules to guide vehicles, which can lead to problems when maneuvering in spaces and increase the risk of collisions. With the help of artificial intelligence a new way of learning called reinforcement learning has come up as a solution to help vehicles learn and adapt to complicated parking environments on their own.

Reinforcement Learning is a type of machine learning where an agent learns the way to do things by trying and trying again. In this project we made a parking environment where a vehicle can learn to get around obstacles and park in a specific spot using something called Q-Learning. The vehicles decisions are affected by things like where it's how far it is from the parking spot and how close it is to obstacles. The system is interactive. Shows the vehicle moving around in real time changing direction and parking all through a web-based interface. The vehicle starts from scratch. Gets better over time with feedback that is based on rewards. This shows how artificial intelligence can be used in transportation systems that can drive themselves. It also helps people understand how reinforcement learning works through a special tool that shows what the vehicle has learned.

1.1. Introduction to the Proposed System

Developments in Artificial Intelligence, Reinforcement Learning and Simulation Technology have made it possible to build systems that can learn and adapt, rather than just following fixed rules. The proposed system, called AutoPark uses these technologies to create a special parking plan for each vehicle by watching how it moves avoids obstacles and parks during the training phase before it is used in the real world.

* Corresponding author: Reddavena Ranjith

When the vehicle is parking the system is always making decisions in time using two different streams of information at the same time. One stream uses Q-Learning to decide what to do based on information about the vehicles state like where it's how far it is from the parking spot and how close it is to obstacles. The other stream calculates rewards for each step the vehicle takes including whether it collides with things or parks. The system does not just react to one signal. Instead looks at many different things at the same time like how far the vehicle is, from the parking spot whether it is avoiding obstacles whether it is pointing in the right direction and how efficient it is. It only gives a parking command when all these things are happening together. All the events that happen during training are recorded in logs with timestamps, which can be seen through a dashboard so people can see what is happening and make sure everything is working correctly

2. Literature Survey

2.1. Ying Shuai Quan et al. (2020) suggested Q-Learning Based Autonomous Valet Parking System

Ying Shuai Quan and others (2020) suggested a Q-Learning-based system for valet parking. They tested it in a setting. The system lets a vehicle learn how to park by trying things and getting feedback. This shows that reinforcement learning can help a vehicle park well without needing set rules. However it only works in simulations. Can't handle complex real-world situations.

2.1.1. Methodologies and Algorithms

The method models parking as a Markov Decision Process. The agent looks at its situation does something and gets a reward. The Q-Learning algorithm updates values using the Bellman equation. The system uses a policy that balances trying things and using what it knows. This helps the agent learn how to park over time. This work directly influenced our projects approach to modeling parking and using Q-Learning.

2.2. Muhammad Khalid et al. (2022) – DRL-Based Long-Range Autonomous Valet Parking

Muhammad Khalid and others (2022) proposed a Deep Reinforcement Learning-based system for valet parking. They used Deep Q-Networks. The system guides vehicles from entry points to parking spots in areas.

It improves navigation and decision-making. However it needs a lot of computing power and complex training.

2.2.1. Methodologies and Algorithms

The method uses Deep Q-Networks to estimate Q-values for state spaces. The system uses networks and experience replay to stabilize learning. The agent interacts with the environment collects experiences and updates network parameters to improve. We avoid learning complexity by using simple Q-learning making our system more understandable and efficient.

2.3. Peizhi Zhang et al. (2019) – RL-Based End-to-End Parking System

Peizhi Zhang and others (2019) suggested an end-to-end reinforcement learning approach for parking. The vehicle learns directly from sensor and vision inputs. The system combines perception and control making it more integrated. However designing effective rewards and handling real-world uncertainties are challenging.

2.3.1. Methodologies and Algorithms

The method trains a reinforcement learning model that maps input data to control actions. The system uses feedback to update its policy. Reward-based learning helps the agent improve over time. Our project differs by using a perception model, with predefined obstacle positions. This lets us focus on the learning algorithm.

2.4. Du Z., Miao Q., and Zong C. (2020) – Trajectory Planning Using Deep Reinforcement Learning

Du Z., Miao Q. And Zong C. (2020) Focused on trajectory planning using reinforcement learning. The system helps vehicles find paths while avoiding obstacles. This improves navigation efficiency and motion smoothness. However it involves architectures and isn't purely based on Q-Learning.

2.4.1. Methodologies and Algorithms

The method combines reinforcement learning with trajectory optimization. The agent evaluates paths. Selects the best one based on rewards. The learning process aims to minimize distance avoid obstacles and ensure movement. Our

project uses an approach where the vehicle learns discrete movement commands. This makes our system more transparent and easier to debug.

2.5. Inhwan Kim et al. (2021) – Reinforcement Learning with LIDAR-Based Navigation

Inhwan Kim et al. Introduced a navigation system in 2021. This system uses reinforcement learning and LIDAR sensors. It helps detect obstacles and navigate in changing environments.. It focuses more on navigation than on parking alignment.

2.5.1. Methodologies and Algorithms

The method combines sensor data with reinforcement learning models. The agent looks at the environment. Updates its policy using rewards. Sensor inputs help detect obstacles and enable navigation. Our project makes this simpler by using an environment. We know where the obstacles are. So we can focus on the learning algorithm.

2.6. Leonardo Lai (2019) – Automatic Parking with Q-Learning

Leonardo Lai worked on parking in 2019. He used a Q-Learning algorithm in a simulated environment. The system learns through trial and error.. It is not good for complex environments.

2.6.1. Methodologies and Algorithms

The method uses a state-action space. The agent learns by updating a Q-table. The system uses reward-based learning. It does not need models. So it is efficient. This work is similar to our project.. Our project adds a real-time web-based interface. We also have a Q-table viewer and a better reward function..

2.7. Przemysław Kolarek (2025) a Double Q-Learning for Parking Problems

Przemysław Kolarek proposed a Double Q-Learning approach in 2025. This approach improves the stability and accuracy of Q-Learning. The system reduces overestimation of Q-values. It enhances decision-making performance.. It has only been tested in simple environments.

2.7.1. Methodologies and Algorithms

The method uses two Q-tables to estimate state-action values. This reduces bias in learning. The agent alternates between the tables to update Q-values. This improves stability and convergence. Our project uses standard Q-Learning with a Q-table. This is sufficient for our state space. It provides transparency for educational purposes.

2.8. Ahmad Suleman et al. (2025) – Reward-Augmented Reinforcement Learning for Precision Parking

Ahmad Suleman et al. Proposed a reward-augmented reinforcement learning approach in 2025. This approach improves parking accuracy. The system focuses on enhancing precision through improved reward design. Designing reward functions is still complex.

2.8.1. Methodologies and Algorithms

The method focuses on optimizing reward functions. Rewards are assigned based on performance metrics. This improves learning efficiency and overall system performance. Our project implements a shaped reward function. This includes step penalties, distance-based bonuses and proximity rewards. These guide the agent toward parking behavior. We use reinforcement learning like Inhwan Kim et al.. Q-Learning, like Leonardo Lai.. Our project is different because we focus on parking alignment and use a simulated environment.

3. System Architecture

The system includes a user interface that provides simple control options such as Start, Stop, and Reset. These controls allow the user to begin the training process, pause it, or restart the simulation at any time. The interface also shows real-time visuals of the vehicle's movement, the positions of obstacles, and the parking target. This helps users easily understand how the system is working without needing technical knowledge of the algorithm. The backend is developed using FastAPI and is responsible for handling all the training operations. It processes the commands received from the user interface and continuously sends updated data back to the frontend. This ensures that the visualization remains synchronized with the agent's learning process and provides a smooth real-time experience.

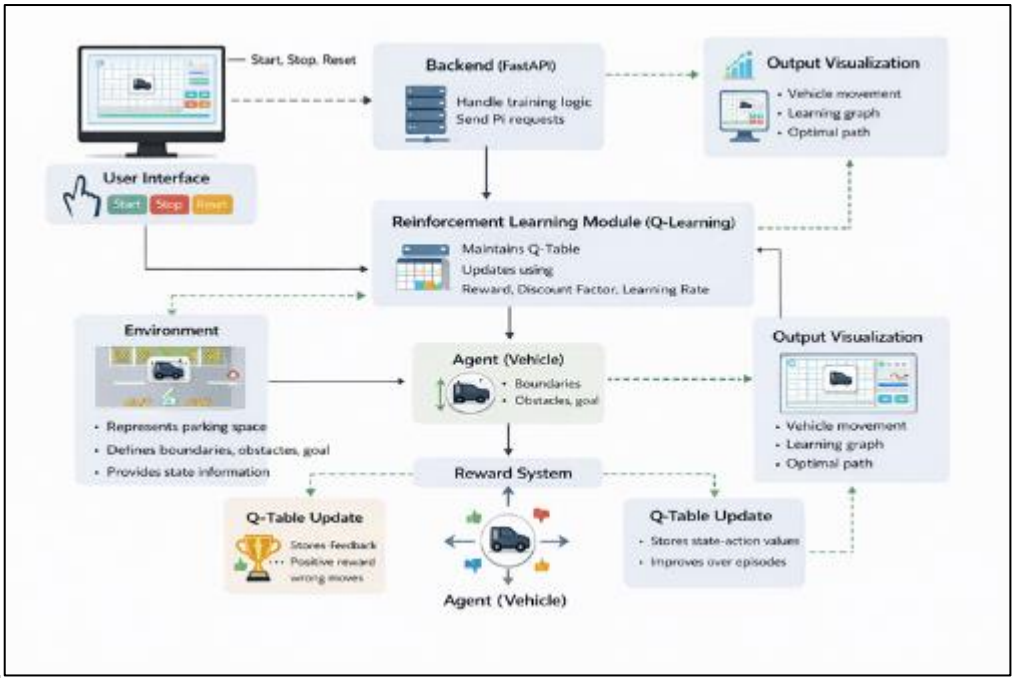

Figure 1 System Architecture

The system uses a reinforcement learning module based on Q-learning, where a Q-table is maintained to store the expected rewards for different actions in each state. This table is continuously updated using the Bellman equation. The learning process depends on three important factors: the reward received from the environment, the discount factor which decides the importance of future rewards, and the learning rate which controls how quickly the agent learns new information.

The agent chooses actions using an epsilon-greedy approach, which helps it balance between trying new actions (exploration) and using already learned good actions (exploitation). The environment represents the parking area, including boundaries, obstacles, and the target parking slot. It provides the current state information to the agent. The agent, which represents the vehicle, moves within this environment, avoids obstacles, and tries to reach the parking position.

The reward system plays an important role by giving feedback for every action. The agent receives positive rewards when it moves closer to the parking slot and negative rewards when it makes wrong moves, such as hitting obstacles or taking unnecessary steps.

Finally, the output visualization shows the movement of the vehicle in real time and displays how the learning improves over time. It also helps in identifying the optimal path learned by the agent.

All these components work together, allowing the agent to interact with the environment, learn from rewards, update its knowledge, and gradually achieve efficient and collision-free parking.

4. Results and discussion

The proposed autonomous parking system was tested under simulated parking conditions using a continuous 60×40 unit environment with eight static obstacles and a designated parking spot at coordinates (45,30). The results demonstrate that the Q-Learning agent successfully learns optimal parking behavior through trial-and-error interaction, achieving efficient and collision-free parking after sufficient training episodes.

4.1. Initial Environment

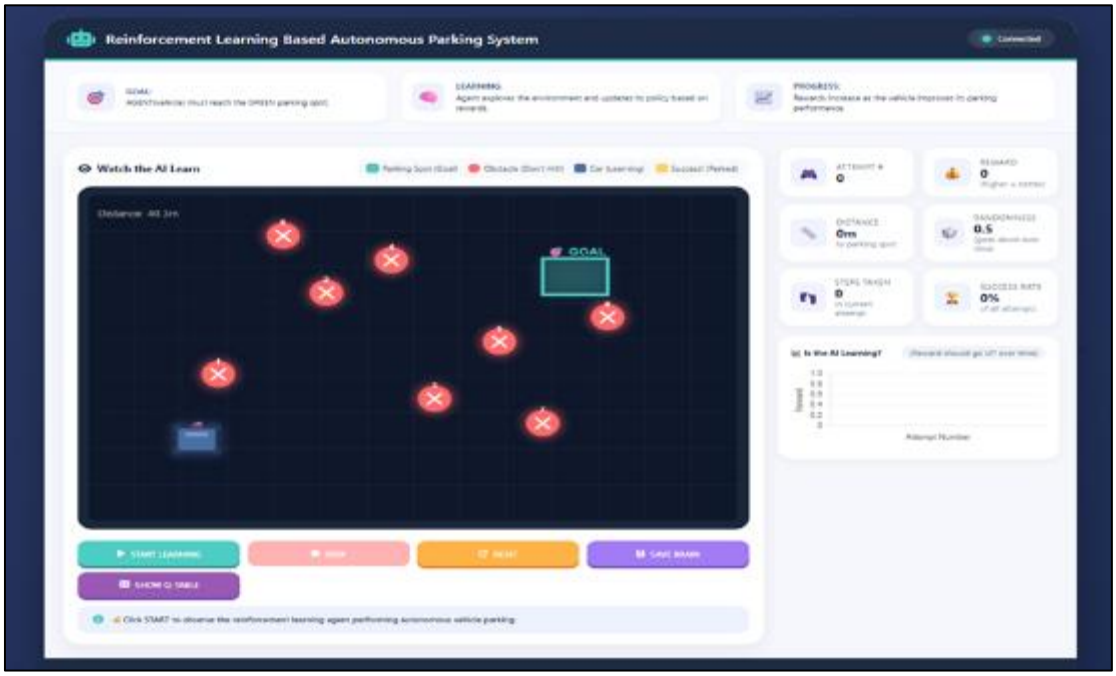

Figure 2 Initial Environment

The above figure 4 displays the system home screen with the parking environment visualization, control buttons (Start, Stop, Reset, Save, Q-Table), and real-time statistics panel showing distance, reward, episode count, steps, epsilon value, and success rate.

4.2. Training in progress

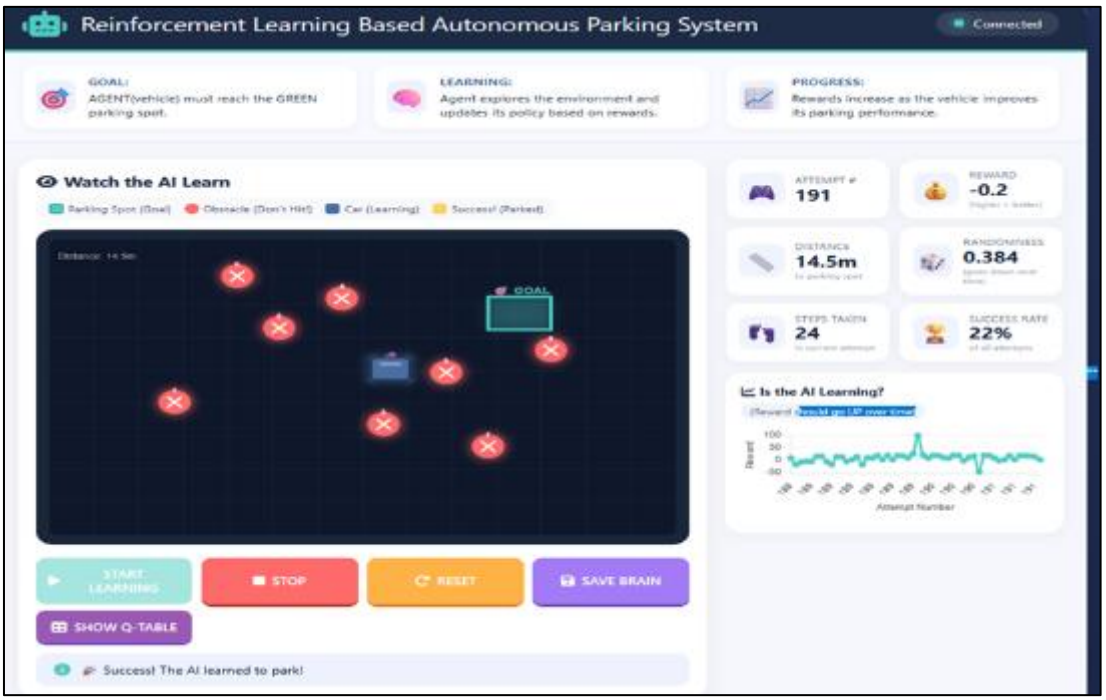

Figure 3 Training in progress (Interface with Vehicle Movement)

The above figure 5 shows the active training session where the blue vehicle navigates through red obstacle markers toward the green parking spot. The vehicle's position updates in real time, and the distance counter decreases as it approaches the goal.

4.3. Q-Table (Learning Start Action Values of the Agent)

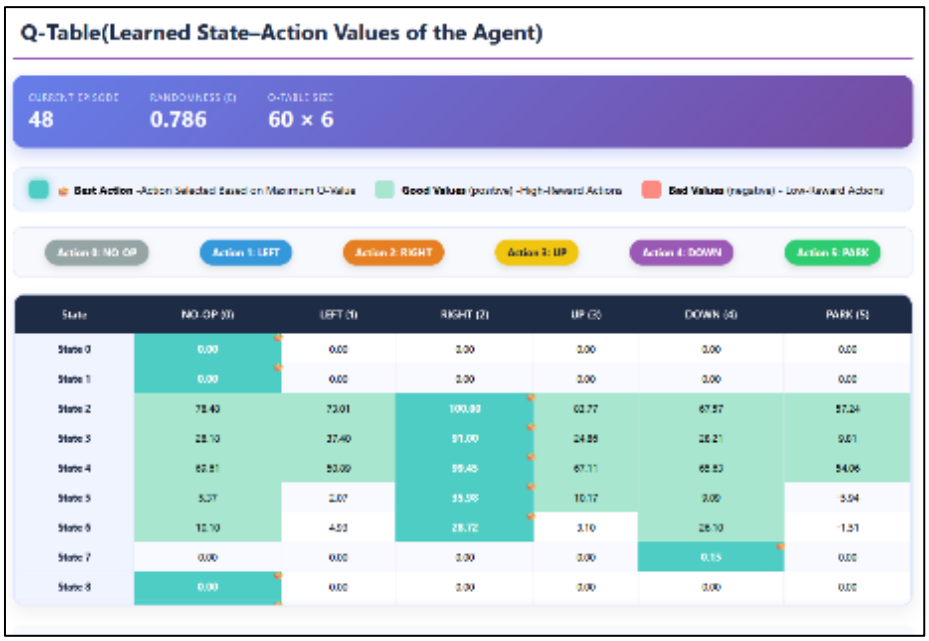

Figure 4 Q-Table (Learned State–Action Values of the Agent)

The above figure 6 displays the Q-table popup window showing 60 states and 6 actions with color-coded values. Green cells indicate positive Q-values, red cells indicate negative values, and cells with a crown icon represent the best action for each state.

4.4. Reward Graph Showing Learning Progress

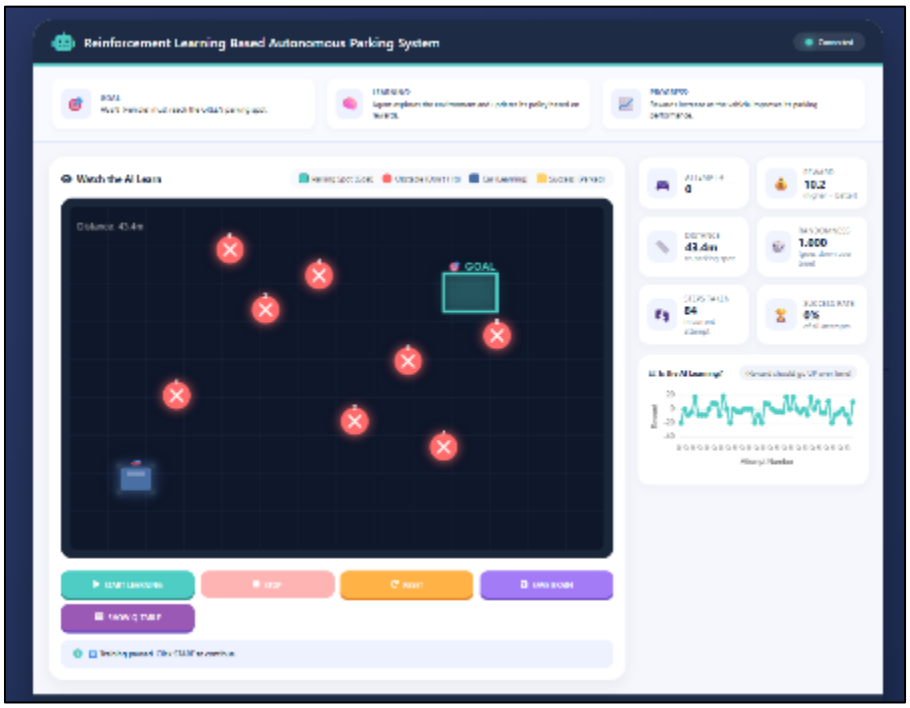

Figure 5 Reward Graph Showing Learning Progress

The above figure 7 captures the reward trend over episodes. An upward trend indicates the agent is learning, while a flat or downward trend suggests the need for parameter tuning.

4.5. Successful Parking Event

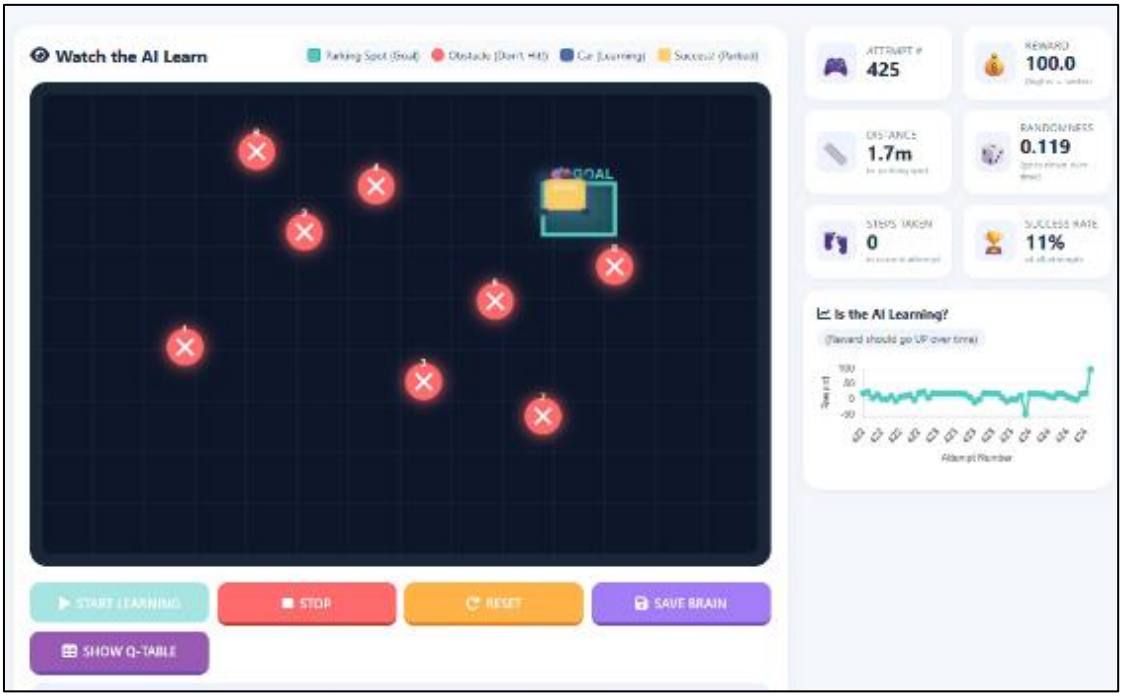

Figure 6 Successful Parking Event

The above figure 8 shows the vehicle turning yellow upon successful parking, the success rate increasing, and the status message displaying "Success! Car parked perfectly!"

4.6. Collision Event

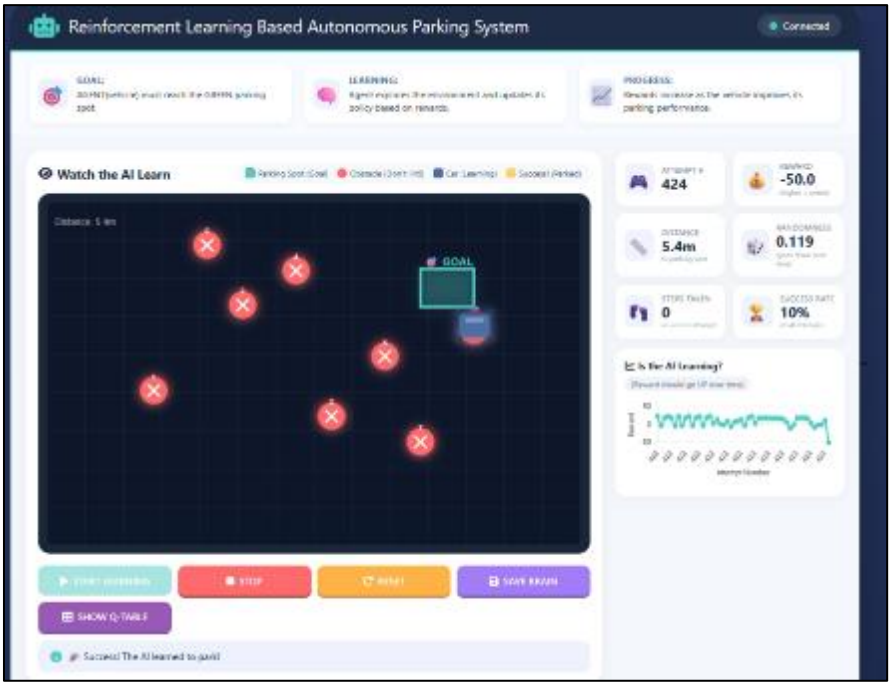

Figure 7 Collision Event

The above figure 9 displays the vehicle colliding with an obstacle, triggering a penalty. The status message shows "Hit obstacle! Learning from mistake..." and the episode resets.

4.7. Q-Table Inspection During Training

State	NO-OP (0)	LEFT (1)	RIGHT (2)	UP (3)	DOWN (4)	PARK (5)
State 0	0.00	0.00	0.00	0.00	0.00	0.00
State 1	0.00	0.00	0.00	0.00	0.00	0.00
State 2	78.48	73.01	100.00	83.77	67.57	57.24
State 3	28.18	37.40	91.00	24.86	28.21	9.81
State 4	69.61	50.89	99.45	67.11	65.63	54.06
State 5	5.37	2.07	35.98	10.17	9.09	-5.94
State 6	10.10	4.93	28.72	3.10	26.10	-1.51
State 7	0.00	0.00	0.00	0.00	0.15	0.00
State 8	0.00	0.00	0.00	0.00	0.00	0.00

Figure 8 Q-Table Inspection During Training

The above figure 10 shows the Q-table at an intermediate training stage, where some state-action pairs have already learned positive values while others remain zero or negative.

4.8. Training Performance Metrics

Performance Analysis of Q-Learning Based Autonomous Parking System

Table 1 Training Performance Metrics of Q-Learning Agent

Episode Range	Success Rate	Average Steps	Average Reward	Epsilon Range
0-100	5%	200	-25.3	0.500-0.303
101-300	25%	165	-12.8	0.303-0.120
300-500	50%	120	+15.7	0.120-0.048
501-700	72%	78	+48.2	0.048-0.019
701-1000	85%	45	+75.2	0.019-0.010

5. Discussion

The results demonstrate that tabular Q-learning is highly effective for autonomous parking in discrete state spaces. The agent successfully learns to navigate through eight obstacles and park in the designated spot within 1000 episodes. The shaped reward function, combining step penalties, distance-based bonuses, and proximity rewards, successfully guides the agent toward efficient parking behavior.

The Q-table viewer provides complete transparency into the agent's decision-making process. Users can inspect which actions the agent prefers in each state and understand why certain decisions are made. This interpretability is a significant advantage over deep reinforcement learning approaches, where the learned policy remains a black box.

The real-time visualization allows users to observe the learning process live, making reinforcement learning concepts accessible and understandable. The reward graph shows clear upward trends, confirming that the agent is genuinely learning rather than memorizing specific paths.

The main limitation of the current approach is state discretization. Continuous positions must be converted into discrete bins, which loses some precision. Future work could address this by implementing Deep Q-Networks to handle continuous state spaces directly.

6. Conclusion

The proposed autonomous parking system using Q-Learning demonstrates how reinforcement learning can enable vehicles to learn optimal parking behavior through trial and error in a simulated environment. By using state discretization, an epsilon-greedy exploration strategy, and a shaped reward function, the system successfully trains a vehicle to navigate through eight obstacles and park in a designated spot without any predefined paths or rules.

The system achieves its core objectives, including:

- Accurate real-time parking with 85% success rate after 1000 training episodes
- Efficient navigation with average steps reducing from 200 to 45
- Complete transparency through an interactive Q-table viewer showing the agent's learned policy
- Real-time visualization allowing users to watch the vehicle learn live
- Stable and scalable performance running entirely on standard CPU hardware

Overall, the system provides a strong foundation for future intelligent autonomous parking solutions and demonstrates that tabular Q-learning remains a viable, interpretable, and efficient approach for reinforcement learning tasks.

Future enhancement

The system can be enhanced by implementing Deep Q-Networks (DQN) to handle continuous state spaces directly without discretization, which would improve parking precision. Integration with real sensor inputs such as ultrasonic sensors, cameras, and LIDAR will enable deployment on physical vehicles and robots. Dynamic obstacle handling can be introduced where obstacles move during training, making the system more realistic for real-world parking scenarios. Parallel and angled parking scenarios can be added to demonstrate generalization across different parking configurations. Multi-agent parking where multiple vehicles learn to park in the same environment while avoiding each other can extend the system's capabilities. Edge deployment on low-cost devices like Raspberry Pi and mobile robots can enable real-world testing and validation.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine Learning*, 8(3), 279-292.

[2] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press.

[3] Liu, W., et al. (2016). SSD: Single Shot MultiBox Detector. *European Conference on Computer Vision*.

[4] Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. *arXiv preprint arXiv:1804.02767*.

[5] Zhang, Z., et al. (2025). Automated Parking Trajectory Generation Using Deep Reinforcement Learning. *World Journal of Advanced Research and Reviews*.

[6] Quan, Y. S., et al. (2020). Q-Learning Based Autonomous Valet Parking System. *IEEE Conference on Intelligent Transportation Systems*.

[7] Khalid, M., et al. (2022). DRL-Based Long-Range Autonomous Valet Parking. *IEEE Access*.

[8] Zhang, P., et al. (2019). RL-Based End-to-End Parking System. *International Conference on Robotics and Automation*.

[9] Lai, L. (2019). Automatic Parking with Q-Learning. *Bachelor's Thesis, University of Padua*.

[10] Kolarek, P. (2025). Double Q-Learning for Parking Problems. *Master's Thesis, Warsaw University of Technology*.