Goal-conditioned Reinforcement Learning Approach for Autonomous Parking in Complex Environments

1st Taeyoung Kim
CCS Graduate School of Mobility
Korea Advanced Institute of
Science and Technology(KAIST)
Daejeon, Rep. of Korea, 34141
ngng9957@kaist.ac.kr

2nd Taemin Kang
The Robotics Program
Korea Advanced Institute of
Science and Technology(KAIST)
Daejeon, Rep. of Korea, 34141
tmkang9826@kaist.ac.kr

3rd Seungah Son
CCS Graduate School of Mobility
Korea Advanced Institute of
Science and Technology(KAIST)
Daejeon, Rep. of Korea, 34141
seungahson@kaist.ac.kr

4th Kuk Won Ko
Department of Future Mobility Engineering
Halla University
Wonju, Rep. of Korea, 26404
kukwon.ko@halla.ac.kr

5th Dongsoo Har
CCS Graduate School of Mobility
Korea Advanced Institute of
Science and Technology(KAIST)
Daejeon, Rep. of Korea, 34141
dshar@kaist.ac.kr

***Abstract*—In complex urban environments, autonomous vehicle parking is challenged by irregularly parked cars, static obstacles, and narrow spaces. Traditional parking methods are often limited to well-structured parking lots and fail to adapt to unstructured scenarios like alleyways with obstacles and randomly positioned vehicles. This paper addresses this limitation by applying goal-conditioned reinforcement learning to enable autonomous vehicles to park in congested environments based on specified target positions and orientations. A custom simulation environment is developed using Pygame to simulate parking scenarios across four progressive levels of difficulty, with obstacles to test adaptability. Three RL algorithms, SAC, HER, and an improved HER variant, are implemented to compare the performance in the simulation environment. Experimental results demonstrate that the proposed approach significantly improves parking success rates and trajectory efficiency in complex scenarios, contributing to robust, adaptable parking solutions for autonomous vehicles in real-world applications.**

***Index Terms*—Autonomous Parking, Reinforcement Learning, Goal-conditioned Reinforcement Learning**

I. Introduction

With the rapid progression of urbanization, the shortage of parking spaces and irregular parking environments have become major challenges in modern society. Addressing these issues is becoming increasingly crucial, particularly with the commercialization of autonomous vehicles. Efficient and safe parking is one of the main tasks in vehicle management.

In recent years, reinforcement learning (RL) has emerged as a powerful tool to address complex decision making problems, such as video games [1], [2], sensor networks [3], [4], and robotics control [5]–[7]. Among these diverse applications, RL has proven particularly effective in autonomous driving and parking systems, where it enables vehicles to learn how to achieve given objectives in various scenarios [8], [9]. The flexibility and adaptability of RL are particularly advantageous in learning optimal parking positions and paths. However, most existing research has been conducted in relatively simplified environments, typically based on parking lots with well-defined vertical and horizontal lanes [10]–[12]. In such environments, the presence of clearly marked parking lines significantly reduces the degrees of freedom in vehicle positioning, simplifying the problem and lowering learning complexity. While such assumptions are valid for many structured parking environments, they fail to address the challenges posed by real-world unstructured parking scenarios.

Parking scenarios in real-world environments can be highly complex. In narrow and irregular alleys, vehicles are often parked in disordered manners, and fixed obstacles such as utility poles, trash bins, or buildings frequently obstruct parking paths. In such environments, autonomous vehicles may find it difficult to park efficiently using parking systems with conventional structured RL approaches. To address these challenges, a goal-oriented learning strategy through Goal-conditioned RL (GCRL) [13] offers a promising solution. GCRL enables the vehicle to learn optimal parking trajectories through RL, allowing it to park accurately at specific target positions and orientations, even in complex environments.

This paper addresses the problem of autonomous parking in complex and irregular environments based on a given target position and orientation. A simulation environment is developed, which differs from conventional parking lot setups categorized into vertical and horizontal spaces. The environment includes irregularly parked vehicles and various obstacles, providing a more challenging scenario for autonomous parking systems. The main contributions of this paper are as follows.

1) A custom 2D simulation environments are developed using Pygame [14], designed to address complex parking

This work was supported by the Korea Institute for Advancement of Technology (KIAT) grant funded by Ministry of Trade, Industry and Energy (MOTIE) (No. P0028821, Development of 22kW Variable Type EV Charging System for Building Up Global Chain of Intelligent/Universal Mechanical Parking System).

scenarios with irregularly parked vehicles and various obstacles. The simulation environment is structured into four progressively challenging levels to enable incremental learning for the agent.
2) The study applies the GCRL framework to address parking problems by defining goals as the target position and orientation. This approach enables efficient learning and generalization across varying parking scenarios.
3) The performance of three RL algorithms, such as Soft Actor-Critic (SAC) [15], Hindsight Experience Replay (HER) [16], and an enhanced version of HER [17], are analyzed to assess the effectiveness of GCRL in solving parking problems.
4) The simulation environment and proposed framework are compatible with OpenAI Gym [18], allowing seamless integration with existing reinforcement learning libraries and methods. This ensures broader applicability and usability of the proposed approach.

The remainder of this paper is structured as follows. Section 2 describes the concepts of RL, GCRL, SAC, HER, and a variant of HER. Section 3 details our proposed method, including the GCRL framework and simulation environment. Section 4 presents the experimental results and performance comparisons. Section 5 concludes this paper.

II. Background

In this section, the concepts of RL, GCRL, SAC, HER, and a variant of HER are presented.

A. Reinforcement Learning

Reinforcement Learning (RL) is a framework in which an agent learns to make decisions by interacting with an environment to maximize cumulative rewards. At each timestep t, the agent observes a state $s_t \in S$ and selects an action $a_t \in A$ according to a policy $\pi : S \to A$. The environment responds to this action by transitioning to a new state s_{t+1} based on a state transition probability $p(s_{t+1}|s_t, a_t)$ and provides a reward $r_t = r(s_t, a_t)$. The object of the agent is to learn a policy π that maximizes the expected sum of future rewards, known as the return. The learning process is guided by experiences $e_t = (s_t, a_t, r_t, s_{t+1})$, stored in a replay buffer, enabling the agent to leverage past experiences to improve sample efficiency during training.

B. Goal-conditioned Reinforcement Learning

Goal-conditioned Reinforcement Learning (GCRL) extends the RL framework by incorporating goals that the agent seeks to achieve within a task, using a goal-conditioned policy that takes both the state and goal as inputs [13]. At the start of each episode, the environment provides an initial state $s_0 \in S$ and a fixed goal $g \in G$. The state includes an observation o and an achieved goal (AG) ag, with AG often representing the state of an object in object-centered environments. At each timestep t, the agent selects an action $a_t \in A$ based on the policy $\pi : S \times G \to A$, using both the current state s_t and goal g. The environment responds to this action by providing a reward $r_t = r(s_t, g, a_t)$ and transitioning to a new state s_{t+1}, determined by the transition probability $p(s_{t+1}|s_t, a_t)$. This interaction continues until reaching a terminal state, with experiences e_t represented as a 5-tuple $(s_t, g, a_t, r_t, s_{t+1})$, representing the goal-oriented exploration and learning.

C. Soft Actor-Critic

Soft Actor-Critic (SAC) is an off-policy RL algorithm designed to improve both stability and exploration in continuous action spaces [15]. SAC optimizes a stochastic policy by maximizing a combination of expected reward and entropy, where the entropy term encourages exploration by favoring policies with higher randomness. This balance between exploration and exploitation is particularly beneficial in complex environments where deterministic strategies may lead to suboptimal local solutions. SAC uses two critic networks to estimate the Q-value for stability and a policy network that outputs a probability distribution over actions. The actor aims to maximize the soft Q-value, which combines the expected reward and an entropy bonus, enhancing both sample efficiency and learning robustness.

D. Hindsight Experience Replay

Hindsight Experience Replay (HER) addresses the challenge of sparse rewards in GCRL by enhancing sample efficiency through reinterpretation of unsuccessful episodes as successful ones, enabling the agent to learn from failures [16]. HER creates a hindsight experience by substituting the original goal g with a hindsight goal g^h, which is an achieved goal (AG) from the same episode. By recalculating the reward based on g^h, i.e., $r_t^h = r(s_t, g^h, a_t)$, the original experience $e_t = (s_t, g, a_t, r_t, s_{t+1})$ is transformed into a new hindsight experience $e_t^h = (s_t, g^h, a_t, r_t^h, s_{t+1})$. This approach allows the agent to learn from outcomes it could achieve, rather than only from the specified goal, significantly improving learning efficiency in sparse reward environments.

E. Failed goal Aware Hindsight Experience Replay

Authors of [17] proposed a novel variant of HER known as Failed Goal Aware Hindsight Experience Replay (FAHER). This approach integrates a clustering-based sampling strategy into HER to enhance the efficiency of experience sampling in robotic tasks. Traditional HER methods typically rely on uniform sampling from a replay buffer, which can result in inefficient training. FAHER addresses this by clustering episodes based on the similarity of achieved goals, with a focus on failed goals, defined as the original goal of an unsuccessful episode. This targeted sampling mechanism improves the likelihood of selecting informative episodes, thus enhancing the learning process.

III. Proposed method

A. Problem Definition and Goal

The parking problem addressed in this study focuses on enabling an autonomous vehicle to park in complex, unstructured environments, such as narrow alleys with vehicles parked

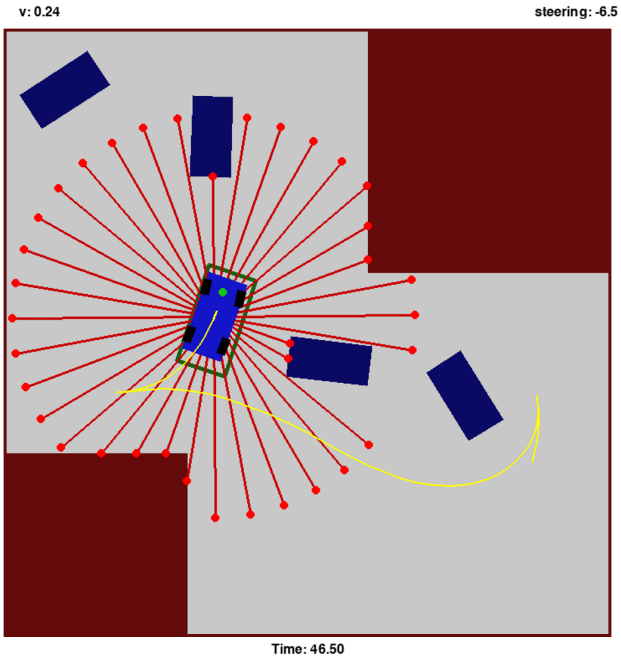

Fig. 1. Illustrations of Autonomous Parking environments.

irregularly due to obstacles like poles and trash bins. Unlike conventional parking setups where parking spaces are well divided, these environments present significant challenges due to unpredictable layouts and obstacles that demand adaptable and precise control.

The goal is defined as reaching a specified parking position with a precise orientation angle. This involves navigating through obstructed surroundings, avoiding collisions, and accurately positioning the vehicle at the target position. To tackle this, the problem is framed using a GCRL approach, where the actions of the agent are conditioned on achieving a dynamically specified goal, i.e., the designated parking spot with specific coordinates and orientation. This approach allows for flexible adaptation to varying goals, enhancing the ability of the agent to generalize across different parking scenarios in irregular environments.

B. Simulation Environment Design

To address the challenges of parking in unstructured and complex environments, a custom simulation environment is developed using Pygame. This environment is designed to be compatible with traditional OpenAI Gym environments to leverage existing codebases and ease the integration of RL algorithms. The environment follows the Gym API structure by defining standard functions such as *reset*, *step*, *seed*, and *compute_reward*, along with an *action_space* class. These components ensure a seamless interface with commonly used RL libraries and algorithms, supporting smooth transitions between simulation environments.

The Pygame environment is illustrated in Fig 1. The vehicle to be parked is represented by a blue rectangle, with a green dot on top indicating the forward direction of the vehicle. Surrounding red lines depict the LiDAR beams. The target goal area, where the vehicle aims to park, is shown as a green rectangle, with a green dot signifying the desired parking orientation. Dark blue rectangles represent other parked vehicles, while brown lines and squares denote building structures. The trajectory of the vehicle is depicted by a yellow line, showing the path taken by the vehicle as it navigates toward the goal area. This trajectory provides visual insight into the navigation decisions of the agent. The simulation displays the simulation time, along with the current speed and steering angle of the vehicle, as on-screen text.

The default values for the entire simulation are as follows:

- Map size: 30m by 30m
- Vehivle size: 4m by 2m; Wheelbase: 2.5m
- Goal area size: 5m by 2.5m
- LiDAR sensor: 36-point virtual LiDAR with 10m range
- Maximum speed: 5m/s; Minimum speed: -3m/s
- Maximum acceleration: 1m/s^2
- Maximum steering angle: $\pm 40°$
- Maximum steering speed: 10°/s

These values can be adjusted according to user preference for specific simulation needs.

The motion of a vehicle is implemented based on the kinematic bicycle model, which simplifies the dynamics by assuming no slip between the tires and the ground. The model equations are given as follows:

$$\dot{x} = v\cos(\theta),\ \dot{y} = v\sin(\theta),\ \dot{\theta} = \frac{v}{L}\tan(\delta),\ \dot{v} = a, \qquad (1)$$

where x and y represent the position of the vehicle, θ is the heading angle, v is the velocity, δ is the steering angle, L is the wheelbase, and a is the acceleration. This model enables efficient simulation of vehicle trajectories in response to control inputs.

The simulation is structured across four progressively challenging levels, each increasing the environmental complexity to gradually develop the skills of the agent in navigation, obstacle avoidance, and precise control required for parking tasks. Each level introduces unique configurations and obstacles, pushing the agent to adapt its strategy and learn complex behaviors.

Fig 2 illustrates sample initial states for four levels. Each level is defined as follows:

- **AutonomousParking-v0:** A position and orientation of the target parking space are randomly selected within a 20m $\times$ 20m 2D area situated inside a larger 30m $\times$ 30m 2D space, which is surrounded by walls. A vehicle to be parked, represented as a blue rectangle, begins at a random position and orientation within the 30m $\times$ 30m space. The objective is to navigate and park the vehicle accurately within the designated goal rectangle.
- **AutonomousParking-v1:** A position and orientation of the target parking space are randomly selected within a 20m $\times$ 20m 2D area situated inside a larger 30m $\times$

30m 2D space, which is surrounded by walls. Four disorderly parked vehicles, represented as black rectangles, are randomly placed within the environment, each with a random orientation, ensuring that they do not overlap with the goal area. A vehicle to be parked begins at a random position and orientation within the 30m × 30m space, ensuring that it does not overlap with any of the parked vehicles. The objective is to navigate and park the vehicle accurately within the designated goal rectangle.

- **AutonomousParking-v2:** A position and orientation of the target parking space are randomly selected within a 20m × 20m 2D area situated inside a larger 30m × 30m 2D space, which is surrounded by walls. Each corner of the outer space has a 75% probability of containing a building, represented as a brown square with a randomly determined size. A vehicle to be parked begins at a random position and orientation within the 30m × 30m space, ensuring that it does not overlap with any of the buildings. The objective is to navigate and park the vehicle accurately within the designated goal rectangle.
- **AutonomousParking-v3:** A position and orientation of the target parking space are randomly selected within a 20m × 20m 2D area situated inside a larger 30m × 30m 2D space, which is surrounded by walls. Each corner of the outer space has a 75% probability of containing a building, represented as a brown square with a randomly determined size. Additionally, four disorderly parked vehicles are randomly placed within the environment, each with a random orientation, ensuring that they do not overlap with the goal and the buildings. A vehicle to be parked begins at a random position and orientation within the 30m × 30m space, ensuring that it does not overlap with any of the parked vehicles or buildings. The objective is to navigate and park the vehicle accurately within the designated goal rectangle.

This incremental-level design provides a structured training approach where the agent can build foundational skills in simpler environments before advancing to complex, real-world-like scenarios. By exposing the agent to progressively challenging tasks, this multi-level design promotes skill generalization, allowing the agent to tackle increasingly congested environments with dynamic obstacles and irregularly parked vehicles. Through these levels, the agent learns to navigate various parking conditions, eventually acquiring robust, adaptable strategies essential for parking in realistic, irregular environments.

C. Training Procedure

In the Autonomous Parking environment, the GCRL framework is described as follows:

- **States:** The state consists of the observation o and the AG ag. The observation is an array of 41 floats, which include information about the vehicle as well as LiDAR data. The information about the vehicle encompasses its position, orientation, speed, and steering angle. The LiDAR data is obtained from a virtual 36-point LiDAR system. The AG is an array of 3 floats, indicating the position and orientation of the vehicle.
- **Goals:** The goal is the position and orientation of the designated parking area. The goal is sampled in a 3D space, represented by x, y, and θ. The achievement of the goal, the success of the episode, is defined by a function checking whether the vehicle is located within the designated parking area with the right orientation.
- **Rewards:** A binary and sparse reward is used. The reward is 0 if the vehicle is in the designated parking area with the right orientation; otherwise, the reward is -1.
- **Actions:** The action is in the 2-dimensional action space, with each action value ranging between -1 and 1. These two values represent the target speed and steering angle of the vehicle, respectively. Each action is scaled to the maximum allowable speed and steering angle. The current speed and steering angle of the vehicle are updated incrementally, respecting predefined limits on acceleration and steering rate, ensuring smooth and controlled adjustments in response to each action input.

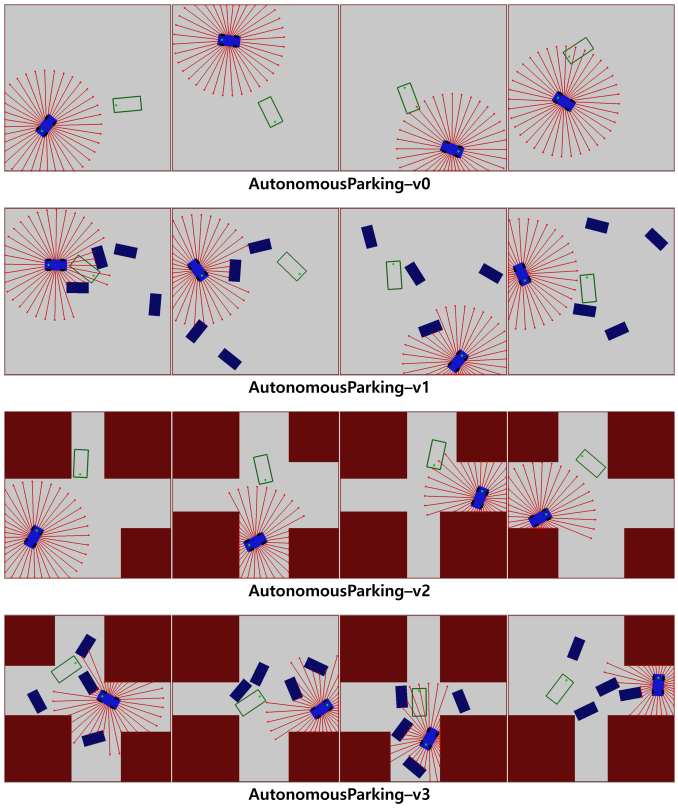

Fig. 2. Illustrations of sample initial states for AutonomousParking-v0, v1, v2, and v3.

To evaluate the effectiveness of using the GCRL approach to address the autonomous parking task in complex and unstructured environments, three RL algorithms are implemented and compared.

- **Soft Actor-Critic (SAC):** A model-free and off-policy algorithm that optimizes a stochastic policy for continu-

ous action spaces, balancing exploration and exploitation. SAC is used as a baseline algorithm due to its robustness in handling complex control tasks by maximizing both expected reward and policy entropy, which encourages exploratory behavior necessary for adapting to diverse parking scenarios.

- **SAC with Hindsight Experience Replay (SAC+HER):** SAC is combined with HER to address sparse reward challenges inherent in autonomous parking tasks. HER enhances learning by replaying unsuccessful experiences as if the agent was aiming to reach a different, achievable goal. This additional feedback is particularly useful in environments where direct successes are rare, providing the agent with more opportunities to learn successful strategies.
- **SAC with Failed goal Aware HER (SAC+FAHER):** To further improve learning efficiency, an enhanced HER variant, termed FAHER, is used with SAC. FAHER implements cluster-based sampling, grouping experiences by achieved goals and sampling from clusters to emphasize harder-to-reach goals. This approach helps the agent learn more effectively in challenging scenarios by focusing on experiences that push the ability of the agent to navigate obstacles and achieve precise parking goals.

These algorithms are integrated within the GCRL framework, allowing for a comprehensive evaluation of the effectiveness of each algorithm in achieving accurate parking in complex and varying environments.

The hyperparameters used in experiments are adopted from [17]. All hyperparameters are described in detail in [16] and [17]. The hyperparameters are as follows:

- Actor and critic networks: 3 layers with 256 units each and ReLU non-linearities
- ADAM optimizer [19] with 10^{-1} of learning rate for both actor and critic
- Action L2 norm coefficient:1.0
- Polyak-averaging coefficient: 0.95
- Number of epochs: 200
- Number of cycles per epoch: 50
- Number of batches per cycle: 40
- Number of workers: 1
- Number of rollouts per worker: 4
- Observation clipping: [-200,200]
- Probability of random actions: 0.3
- Scale of additive Gaussian noise: 0.2
- Buffer size: 10^6 transitions
- Batch size: 256
- Probability of HER experience replay: 0.8
- Number of clusters: 16
- Failed goal buffer size: 150

IV. Experimental Results

The performance of three RL algorithms is evaluated in terms of success rate over the course of training. Figures illustrate the progression of success rates across training epochs, while tables provide a summary of the final success

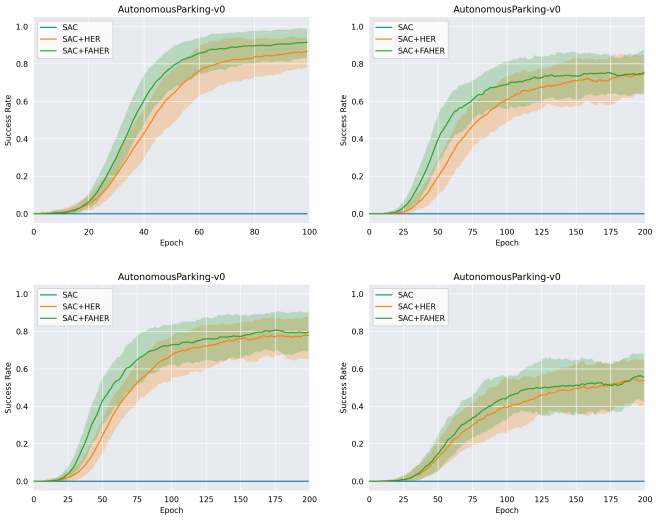

Fig. 3. Success rates obtained while training SAC, HER, and FAHER.

rates achieved after training. The success rate at each epoch of training is calculated based on performance across 20 test episodes, with training spanning a total of 200 epochs. To enhance the robustness and reliability of the reported outcomes, the entire training and evaluation process is repeated using five distinct random seeds, and the results are averaged accordingly.

In the graphs, a solid line represents the average success rate across the five repetitions at each epoch, with the shaded area indicating the range between the minimum and maximum success rates. To smooth the epoch-wise results, a moving average of the previous 20 success rates is calculated and depicted in the figures, providing a clearer trend over time.

Figure 3 compares the performance of SAC, SAC+HER, and SAC+FAHER across different levels of the AutonomousParking environment. In the graphs, a solid line represents the average success rate across the five repetitions at each epoch, with the shaded area indicating the range between the minimum and maximum success rates. To smooth the epoch-wise results, a moving average of the previous 20 success rates is calculated and depicted in the figures.

In all levels (v0 to v3), the SAC baseline consistently shows a success rate of 0, demonstrating its ineffectiveness in the AutonomousParking environment without the GCRL approach. Conversely, SAC+FAHER consistently achieves a higher success rate more quickly than SAC+HER, especially in the earlier levels.

In the v0 level, SAC+FAHER outperforms SAC+HER in achieving a higher success rate at a faster pace. Approximately after 50 epochs, SAC+FAHER reaches a success rate exceeding 0.8, whereas SAC+HER shows a slower ascent to a similar success rate. In this phase, SAC+FAHER demonstrates quicker convergence and a higher final success rate compared to SAC+HER.

In the v1 level, SAC+FAHER reaches a higher success

rate more rapidly than SAC+HER. After about 100 epochs, SAC+FAHER achieves a success rate above 0.7, while SAC+HER improves at a slower rate. During the early to mid-training phases of v1, SAC+FAHER exhibits superior performance compared to SAC+HER. However, the performance gap narrows in the final success rate.

Similarly, in the v2 level, SAC+FAHER continues to rise in success rate more quickly than SAC+HER. By around 60 epochs, SAC+FAHER achieves a success rate of 0.8, whereas SAC+HER shows a slower increase thereafter, eventually reaching a comparable level. Throughout this phase, SAC+FAHER consistently outperforms SAC+HER in terms of training speed and the final success rate.

In the final level (v3), SAC+FAHER exhibits a rapid increase in success rate compared to SAC+HER. After approximately 150 epochs, the increase in success rate stagnates for both algorithms, with maximum success rates plateauing around 0.55. This level appears to present a more challenging environment for both algorithms, indicating that while SAC+FAHER shows faster learning initially, the final performance does not exhibit a significant difference.

Overall, across all levels, SAC+FAHER tends to reach a higher success rate more quickly than SAC+HER. However, as the levels progress, the performance gap between the two algorithms diminishes. Particularly in the v3 level, the increased difficulty of the environment restricts performance improvements for both algorithms.

V. Conclusion

This study proposes a goal-conditioned reinforcement learning (GCRL) approach to address the problem of autonomous parking in unstructured and complex environments, such as alleyways with irregularly parked vehicles and various obstacles. To support this, a custom simulation environment is developed in Pygame, featuring four progressively challenging levels that simulate realistic parking scenarios.

This study compares the performance of SAC, SAC+HER, and SAC+FAHER to assess the parking success rates of each algorithm. SAC, without the GCRL approach, consistently shows a success rate of 0 across all stages, indicating its ineffectiveness in complex, unstructured environments. In contrast, both SAC+HER and SAC+FAHER demonstrate significantly higher success rates, with SAC+FAHER showing faster convergence and better overall performance across all stages. This emphasizes the importance of incorporating HER-based strategies to enhance goal-reaching capabilities in challenging environments.

Future research could extend this approach by developing a 3D simulation environment using a physics engine to simulate more realistic dynamics and by incorporating dynamic obstacles to further challenge the agent. Additionally, real-world testing will be essential to validate the transferability of learned policies to physical vehicles. This research contributes to the advancement of adaptable and reliable autonomous parking systems capable of operating in complex urban scenarios, bridging the gap between simulated training and practical application.

References

[1] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al., "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[2] R. R. Torrado, P. Bontrager, J. Togelius, J. Liu, and D. Perez-Liebana, "Deep reinforcement learning for general video game AI," in Proc. IEEE Conf. Comput. Intell. Games (CIG), 2018, pp. 1–8.

[3] I. Park, D. Kim, and D. Har, "MAC achieving low latency and energy efficiency in hierarchical M2M networks with clustered nodes," IEEE Sensors Journal, vol. 15, no. 3, pp. 1657-1661, 2015.

[4] T. Kim, L. F. Vecchietti, K. Choi, S. Lee, and D. Har, "Machine learning for advanced wireless sensor networks: A review," IEEE Sensors Journal, vol. 21, no. 11, pp. 12379–12397, 2020.

[5] M. Seo, L. F. Vecchietti, S. Lee, and D. Har, "Rewards prediction-based credit assignment for reinforcement learning with sparse binary rewards," IEEE Access, vol. 7, pp. 118776–118791, 2019.

[6] L. F. Vecchietti, M. Seo, and D. Har, "Sampling rate decay in hindsight experience replay for robot control," IEEE Transactions on Cybernetics, vol. 52, no. 3, pp. 1515–1526, 2020.

[7] B. Singh, R. Kumar, and V. P. Singh, "Reinforcement learning in robotic applications: A comprehensive survey," Artificial Intelligence Review, vol. 55, no. 2, pp. 945–990, 2022.

[8] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," IEEE Trans. Intell. Transp. Syst., vol. 23, no. 6, pp. 4909–4926, 2021.

[9] B. B. Elallid, N. Benamar, A. S. Hafid, T. Rachidi, and N. Mrani, "A comprehensive survey on the application of deep and reinforcement learning approaches in autonomous driving," J. King Saud Univ. Comput. Inf. Sci., vol. 34, no. 9, pp. 7366–7390, 2022.

[10] P. Zhang, L. Xiong, Z. Yu, P. Fang, S. Yan, J. Yao, and Y. Zhou, "Reinforcement learning-based end-to-end parking for automatic parking system," Sensors, vol. 19, no. 18, p. 3996, 2019.

[11] J. Zhang, H. Chen, S. Song, and F. Hu, "Reinforcement learning-based motion planning for automatic parking system," IEEE Access, vol. 8, pp. 154485–154501, 2020.

[12] Z. Du, Q. Miao, and C. Zong, "Trajectory planning for automated parking systems using deep reinforcement learning," Int. J. Automot. Technol., vol. 21, no. 4, pp. 881–887, 2020.

[13] M. Liu, M. Zhu, and W. Zhang, "Goal-conditioned reinforcement learning: Problems and solutions," arXiv preprint arXiv:2201.08299, 2022.

[14] Pygame Development Team, "Pygame: A set of Python modules designed for writing video games," [Online]. Available: https://www.pygame.org/.

[15] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in Proc. Int. Conf. Mach. Learn., pp. 1861–1870, 2018.

[16] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, "Hindsight experience replay," Adv. Neural Inf. Process. Syst., vol. 30, 2017.

[17] T. Kim, T. Kang, H. Jeong, and D. Har, "Clustering-based Failed goal Aware Hindsight Experience Replay," PeerJ Comput. Sci., vol. 10, p. e2588, 2024.

[18] G. Brockman, "OpenAI Gym," arXiv preprint arXiv:1606.01540, 2016.

[19] D. P. Kingma, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.