# Autonomous Parking Using Deep RL

**Dylan Fulop** **Ovee Manolkar** **Nishad Kane** **Alifiya Iqbal Pithawala**

**Vihang Pancholi (Leader)**
Arizona State University
vpancho1@asu.edu

## Abstract

In the context of advancing autonomous driving, the persistent challenge of parking necessitates exploration into Reinforcement Learning (RL). This project addresses the pertinent problem of parking difficulty despite driving competence, aiming to determine if RL can effectively facilitate autonomous parking within a simulated environment. Leveraging Unity's ML-Agents toolkit and simulated vehicles, the project's solution involved RL-based simulations. Despite initial obstacles with Sol and Carla, the transition to Unity yielded notable advancements. The developed RL strategy exhibited consistent parking performance across various starting locations, showing promise in addressing the parking problem. However, limited generalization and underutilization of the second parking space were observed. Further exploration into hyperparameters, reward functions, and alternative algorithms like SAC is essential for refinement. Notably, the project's model required an extensive training period to achieve consistent and satisfactory parking efficiency.

## 1 Introduction

The automotive industry has experienced a revolution caused by advancements in autonomous driving technologies. However, achieving complete autonomy still requires overcoming the barrier of accurate and efficient parking. Enhancing safety, convenience, and overall efficiency in urban environments is contingent upon a vehicle's capacity to navigate and park autonomously in a variety of scenarios. Even with advancements in autonomous driving, parking remains a challenging task, especially in highly congested and intricate urban environments.

This project explores the field of Reinforcement Learning (RL) especially as it relates to autonomous parking. Reinforcement Learning (RL), a subfield of machine learning, has showed that it can allow autonomous systems to communicate with their surroundings and also learn to make judgements. In this project, the central question that we try to answer is whether reinforcement learning (RL) can be used to teach cars to park themselves in a virtual setting. Because of the emphasis on simulations, parking behaviours can be trained and improved in a controlled yet dynamic environment without posing any risks to real life.

Proximal Policy Optimisation (PPO), the RL algorithms used in the development of reliable parking strategies for autonomous vehicles, is thoroughly explored in this project. The project attempts to handle the challenges of parking by utilising Unity's ML-Agents toolkit in simulated environments. Its goal is to close the gap between theoretical developments and real-world applications in autonomous parking systems. The results of this project have the potential to completely transform the field of autonomous driving by overcoming the enduring difficulty of accurate and flexible parking in real-world situations.

## 2 Framework

### 2.1 Imitation Learning

Imitation learning[10] is a machine learning technique where an agent learns by observing and imitating expert behavior rather than solely relying on trial and error. It involves learning from demonstrations or examples provided by a human or an expert agent. This method is often used when direct exploration in a complex environment might be time-consuming or risky.

### 2.2 Proximal Policy Optimization(PPO)

PPO[6] is a reinforcement learning algorithm used for training agents to make decisions in environments, especially in scenarios like games or simulations. It belongs to the family of policy gradient methods and focuses on optimizing policies that control an agent's behavior. PPO aims to balance the exploration of new strategies and the exploitation of known effective strategies by updating policies based on experiences gathered during training. It's known for its stability and effectiveness in various environments.

### 2.3 Reward Function

The reward functions play a pivotal role in steering the reinforcement learning process for autonomous parking. As demonstrated in Table 1, the diverse range of reward and penalty values assigned to specific actions or scenarios guides the learning algorithm in comprehending desirable behaviors. For instance, positive rewards, such as "+5" for successfully entering a parking space or "+100" for achieving successful parking, incentivize the system to emulate these favorable actions. Conversely, penalties like "-50" for collisions with walls or other cars at varying speeds discourage undesirable behaviors. The accompanying explanations in the comments column offer deeper insights into the nuanced conditions and constraints associated with each reward function. Functions like "Angle difference while in a space" and "Direction change" highlight specific calculations and constraints crucial for optimal parking maneuvers. This intricate reward structure serves as a fundamental guide, steering the autonomous system toward learning and exhibiting efficient and safe parking behaviors.

### 2.4 Methodology

In our approach, we opted for Unity with ML-Agents due to installation issues with Carla. Unity's established ML-Agents plugin compared to Unreal's provided a more dependable framework.

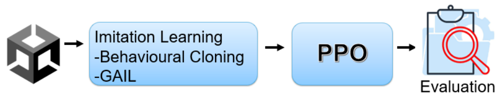

Figure 1: System Pipeline

We utilized the ML-Agents package, functioning within Unity and Python, streamlining reinforcement learning agent creation. Leveraging an existing car controller and the ML-Agents' "Agent" class in Unity, we extended functionality for learning-capable agents. Our tasks involved Unity configuration, resource allocation, and programming reward calculations and observations. The remaining tasks were managed through the Unity editor or templates. Our work drew heavily from a GitHub tutorial[12], providing initial guidance on hyper-parameters and core reward mechanisms.

Figure 2 shows that the car was given 14 ray-perception sensors which detect the distance of an object in a particular direction, as long as there is an object less than some specified distance away.

Our environment featured a compact parking lot with six spaces on each side. The car agent, spawned randomly, interacted with our reward system based on its actions. We opted for the PPO learning algorithm, configuring parameters like batch size, buffer size, and learning rate to facilitate effective training.

Implementing a linear learning rate initially, we later switched to a constant schedule for more efficient training resumption. Resuming training took significantly less time with the constant

Table 1: Reward function

| Reward Function | Value | Comments |
|---|---|---|
| Enter a space | +5 | |
| Leave a space | -5 | |
| Angle difference while in a space = $\lvert\Theta_C - \Theta_S\rvert$ | $-\frac{1}{45} \times \lvert\Theta_C - \Theta_S\rvert + 1$ | Only active while in space. Angles are changed to be within 90 of each other |
| Successful parking | +100 | |
| Collide with wall | -50 | |
| Collide with car at speed $\lvert v\rvert$ | $-\lvert v\rvert \times 50 - 5$ | |
| Current target space reward | $\max_i RS_i$ | Take the max reward from the reward from each space as explained below |
| Direction change | $clip((\Delta d_{l1}(car, S_i)) \times 10) = DC_i$ | Use l1 distance for simplicity. Clip to be between -0.5,0.5. Only added to RSi |
| Total Distance | $\left(1 - \frac{d_{l1}(car,S_i)}{N}\right)/20 = TD_i$ | N is used to normalize so that value is always less than 1, then divide by 20. Only added to RS$_i$ |
| Reward for space $i$ | $RS_i = DC_i + TD_i$ | Only added to R if maximum over i |

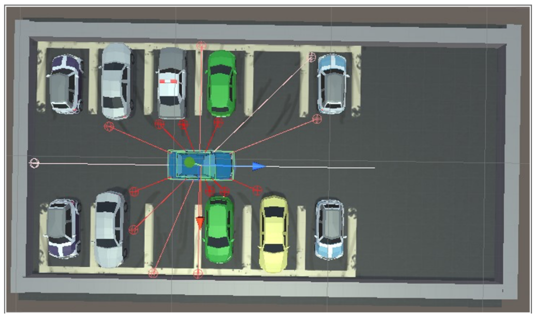

Figure 2: Ray Perceptron Sensors

schedule, resolving potential errors encountered with the linear schedule. Despite some delay in resuming, potentially related to a bug within ML-Agents, our neural network architecture comprised normalization and three layers with 264 hidden units.

Imitation learning became part of our strategy, seamlessly integrated with ML-Agents. Using the "demonstration recorder" in Unity, we manually parked the car numerous times, integrating these demos into our training setup with behavioral cloning and GAIL techniques.

## 3 Results

The goal of our project is to solve the persistent problem that people face while parking even when they are skilled drivers. Utilizing Unity's ML-Agents toolkit, the simulated environment offered a controlled and dynamic environment to investigate the effectiveness of Reinforcement Learning (RL) in enabling autonomous parking.

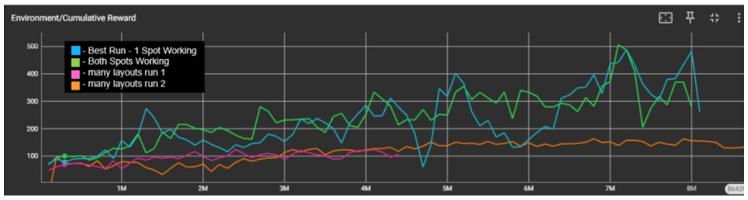

Figure 3: Reward vs Timestep Plot

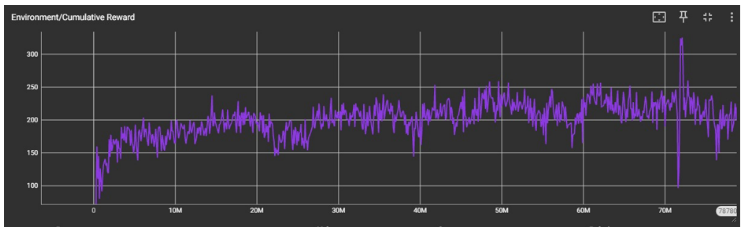

Figure 4: Reward vs Timestep Plot

Figure 4 shows the reward vs timestep plot of the most recent run in the environment where we attempt to train a generalized agent. While it only reaches 220, it is clearly improving and given more time may have been able to successfully park.

The video at `https://www.youtube.com/watch?v=GX7BGDN3KYs` shows our trained agent which can successfully park. After roughly 70 hours of training, our attempted generalized solution is improving very slowly. This video at `https://www.youtube.com/watch?v=PYHdERzc84I` shows that attempted solution and where we were able to get it within the time constraints.

## 4 Discussion

In Unity, persistent challenges emerged during our experiments. Initially, the car showed a consistent preference for parking in one space due to an issue with the orientation of "spaces" objects. Generalization was limited, requiring multiple environment copies, yet progress toward proficient parking was slow. Despite extensive training of about 200 million iterations, the reward plateaued at 210, well below our target of 400.

While Unity allowed consistent parking regardless of the initial position, substantial challenges persist, demanding further exploration. To improve performance, we're exploring hyper-parameters, diverse reward functions, and the necessity of techniques like Behavioral Cloning (BC) or Generative Adversarial Imitation Learning (GAIL) through ablation studies. The complexity of consistently utilizing the second parking space highlights the ongoing effort needed to overcome this challenge.

## 5 Conclusion

In conclusion, the project's pivot to Unity proved pivotal in achieving notable progress in reinforcement learning for parking scenarios. The successful implementation of a dual-target parking strategy showcased adaptability and effectiveness in complex environments. However, further refinement is imperative. Despite achieving consistent parking abilities within the designated environment, ongoing efforts in fine-tuning hyperparameters, reward functions, and exploring alternative algorithms remain crucial. The persistent challenge of generalization, exemplified by the model's difficulty in utilizing the second parking space consistently, signals the need for continued research and optimization in reinforcement learning methodologies for parking scenarios, delineating clear pathways for future development.

## References

[1] Takehara, R., & Gonsalves, T. (2021). *Autonomous Car Parking System using Deep Reinforcement Learning*. In 2021 2nd International Conference on Innovative and Creative Information Technology (ICITech) (pp. 85-89). Salatiga, Indonesia. doi: 10.1109/ICITech50181.2021.9590169.

[2] Thunyapoo, B., Ratchadakorntham, C., Siricharoen, P., & Susutti, W. (2020). *Self-Parking Car Simulation using Reinforcement Learning Approach for Moderate Complexity Parking Scenario*. In 2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON) (pp. 576-579). Phuket, Thailand. doi: 10.1109/ECTI-CON49241.2020.9158298.

[3] Junzuo, L., & Qiang, L. (2021, April). *An Automatic Parking Model Based on Deep Reinforcement Learning*. Journal of Physics: Conference Series, 1883(1), 012111. https://dx.doi.org/10.1088/1742-6596/1883/1/012111

[4] Zhang, J., Chen, H., Song, S., & Hu, F. (2020). *Reinforcement Learning-Based Motion Planning for Automatic Parking System*. IEEE Access, 8, 154485-154501. doi: 10.1109/ACCESS.2020.3017770.

[5] Zhang, P., Xiong, L., Yu, Z., Fang, P., Yan, S., Yao, J., & Zhou, Y. (2019). *Reinforcement Learning-Based End-to-End Parking for Automatic Parking System*. Sensors, 19(18), 3996. https://doi.org/10.3390/s19183996
[6] Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., & Wu, Y. (2022). *The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games*. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), Advances in Neural Information Processing Systems (Vol. 35, pp. 24611-24624). Curran Associates, Inc. https://proceedings.neurips.cc/paper$_files/paper/2022/file/9c1535a02f0ce079433344e14d910597 - Paper - Datasets_and_Benchmarks.pdf$

[7] Albilani, M., & Bouzeghoub, A. (2022). *Dynamic Adjustment of Reward Function for Proximal Policy Optimization with Imitation Learning: Application to Automated Parking Systems*. In 2022 IEEE Intelligent Vehicles Symposium (IV) (pp. 1400-1408). Aachen, Germany. doi: 10.1109/IV51971.2022.9827194.

[8] Tiong, T., Saad, I., Teo, K. T. K., & Bin Lago, H. (2022). *Autonomous Valet Parking with Asynchronous Advantage Actor-Critic Proximal Policy Optimization*. In 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 0334-0340). Las Vegas, NV, USA. doi: 10.1109/CCWC54503.2022.9720881.

[9] Shi, J., Li, K., Piao, C., Gao, J., & Chen, L. (2023). *Model-Based Predictive Control and Reinforcement Learning for Planning Vehicle-Parking Trajectories for Vertical Parking Spaces*. Sensors, 23, 7124. https://doi.org/10.3390/s23167124 [10] Chen, X., Zhou, Z., Wang, Z., Wang, C., Wu, Y., & Ross, K. (2020). *BAIL: Best-Action Imitation Learning for Batch Deep Reinforcement Learning*. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, & H. Lin (Eds.), Advances in Neural Information Processing Systems (Vol. 33, pp. 18353-18363). Curran Associates, Inc. https://proceedings.neurips.cc/paper$_files/paper/2020/file/d55cbf210f175f4a37916eafe6c04f0d - Paper.pdf$

[11] Hua, J., Zeng, L., Li, G., & Ju, Z. (2021). *Learning for a Robot: Deep Reinforcement Learning, Imitation Learning, Transfer Learning*. Sensors, 21, 1278. https://doi.org/10.3390/s21041278

[12] Thomas Van Isegham. (2022). *AI-Parking-Unity*. Retrieved from `https://github.com/VanIseghemThomas/AI-Parking-Unity`

[13] Thomas Van Isegham. (2022). *Automated Parking Using RL, a Unity ML-Agents Tutorial.* [Video]. Retrieved from `https://www.youtube.com/watch?v=_Bzw2B-9QkM`

## Contribution

| Group Number | SI. No | Student Name | Email id | Contribution | Contribution Percentage | Signature |
|---|---|---|---|---|---|---|
| 13 | 1 | Vihnag Pancholi | vpancho1@asu.edu | Worked on Project proposal, Literature review, Unity installation, mid presentation, Reward function analysis, Initial training, Report Preparation | 20% | [illegible] |
| | 2 | Dylan Fulap | dfulop@asu.edu | Worked on Project proposal, Running multiple algorithms for mujoco, Unity installation, mid presentation, Implementing pre-trained model and analysis, Initial training | 20% | [illegible] |
| | 3 | Ovee Manolkar | omanolka@asu.edu | Worked on Project proposal, Literature review, Reward function analysis, Initial training, Analysis of first RL model, Report Preparation | 20% | [illegible] |
| | 4 | Nishad Kane | nskane@asu.edu | Worked on Project proposal, Literature review, mid presentation, Reward function analysis, Initial training, Report Preparation | 20% | [illegible] |
| | 5 | Alifiya Iqbal Pithawala | apithawa@asu.edu | Worked on Project proposal, Literature review, Unity installation, Implementing pre-trained model and analysis, Initial training, Analysis of first RL model, Report Preparation | 20% | Alifiya |