Adversarial Policy Generation in Automated Parking

Alessandro Pighetti[1,2] Francesco Bellotti[1] Riccardo Berta[1] Andrea Cavallaro[2,3,4]

Luca Lazzaroni[1] Changjae Oh[2]

[1]University of Genoa, Italy
[2]Queen Mary University of London, UK
[3]Idiap Research Institute, Switzerland
[4]EPFL, Switzerland

Abstract

Automated driving (AD) systems require rigorous testing to ensure safety and robustness, especially in corner-case scenarios, before real-world deployment. Deep reinforcement learning (DRL) is a promising approach for decision-making in AD, enabling dynamic learning through trial and error. Adversarial agents can be used to expose DRL systems to critical corner-cases, but reward functions solely opposing the AD agent's objectives can lead to unrealistic behaviors, such as overly incentivizing crashes. This paper explores an automated parking (AP) scenario where an adversarial agent disrupts a parking agent exiting an adjacent slot—a common but under-explored corner-case challenge. We propose a more balanced adversary reward function, aiming for realistic yet disruptive behavior compared to the baseline approach. The results show promising improvements in correspondence with the operational design domain (ODD) of AP systems, encouraging further investigation into system performance after several victim-adversary training iterations.

1. Introduction

Development of automated driving functions (ADFs) face compelling challenges, primarily due to strict requirements in safety and security [5] in heterogeneous decision-making contexts [7], [6]. A significant application case is Automated Parking (AP) [10], a system that autonomously drives a vehicle in a target parking slot.

Recent works focus on assessing and strengthening the robustness of AD systems in safety-critical, yet realistic challenging scenarios. This evaluation problem has been recently tackled by iterative search of adversarial trajectories (i.e., aimed at disrupting the functionalities of an AD system) and realistic scenario generation [8], also in the DRL context [4]. The idea of DRL-based adversarial policies has been introduced in zero-sum games between simulated humanoid robots [3]. Here, the victim agent is a DRL model, pre-trained through self-play and then frozen. To effectively investigate challenging scenario adaptation in AD, Chen *et al.* [1] achieve promising results by guiding a fleet of adversarial vehicles through non-zero-sum assumption and introducing a traffic law-guided reward function to generate realistic trajectories in a lane-change scenario.

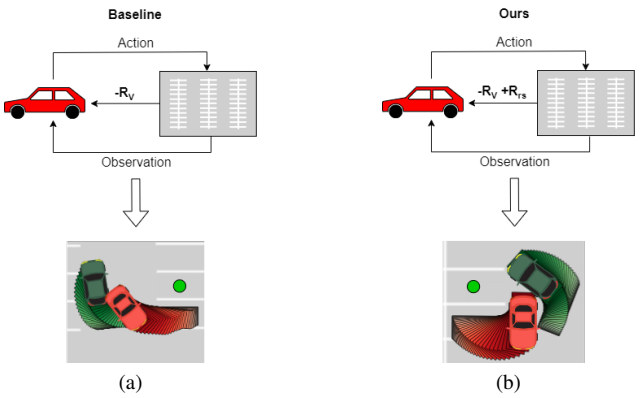


Figure 1. Overview of the proposed method. Trajectories of victim (green) and adversarial (red) agents in (a) the baseline and (b) ours. The adversarial agent in (a) intentionally chases the victim agent and makes a collision, while the one in (b) blocks the victim from reaching the target parking slot.

This manuscript investigates the generation of an effective adversarial policy to validate the performance of a

DRL-based AP system in simulation contexts. Fig. 1 depicts a simplified overview of the proposed method. This approach aims to smooth out a basic adversarial strategy obtained through a reward function that is the plain negation of the reward intake of the AP agent.

The main contributions of this paper include: (i) we investigate an AP scenario in which one of the parked vehicles (i.e., the adversary) is exiting from its parking slot while another (i.e., the victim) is entering an adjacent slot (ii) we introduce a safety-guided training procedure for the adversary to induce a balanced trade-off between realistic and adversarial behavior; (iii) we demonstrate the benefit of the proposed adversarial training by comparing the trajectories generated by the AP agent before and after the deployment of the policy fine-tuned through the interaction with the adversary.

2. Method

We deploy a pre-trained DRL-based AP model derived from previous work as our victim. The details of its training and testing procedure, are reported in [6]. The initial parking scene is populated by environment vehicles (Es) and a victim vehicle (V) We define the complete state of the environment as $S = D_n$, where D_n refers to the position, longitudinal and lateral acceleration, speed and angular velocity of all vehicles in the scene.

2.1. Adversarial policy learning

To assess the robustness of the presented victim model in a realistic and challenging scenario, we propose to learn an adversarial policy that disrupts the functionalities of the parking agent, simulating a situation in which the parking agent tries to park in the designated parking slot while another agent is exiting from an adjacent one. Therefore, we set the starting point of the adversarial agent to be one of the slots near to the victim agent. Fig. 2 depicts the proposed DRL-guided pipeline. The training agent is just the adversary, while the frozen victim is treated as part of the environment. In this context, the observable state space of the adversarial agent is defined by the tuple:

$$O_A = \{dist(V, tgt), dist(A, V), D_A\}, \tag{1}$$

where $dist(V, tgt)$ and $dist(A, V)$ are the Euclidean distance between the victim agent and the target position and between the victim agent and the opponent position, respectively. D_A refers to the subset of the complete state S, only containing the features describing the adversarial vehicle. Regarding the adversarial action space, we define $\mathcal{A}_A$ = throttle, brake, steering angle, reverse, the same as for V [6]. In this context, we deploy the Proximal Policy Optimization (PPO) algorithm [9]. By iteratively adjusting the policy parameters based on a clipped surrogate objective, it strikes a balance between exploration and exploitation while maintaining stability and dependable performance.

2.2. Adversarial reward

To obtain a successful adversarial policy against our pre-trained victim, the underlying structure of the reward signal R_A must be designed to be disruptive to the victim policy, whose objective is defined by R_V. Existing work [3], learn adversarial policies by formulating the problem as zero-sum game, which can be directly modeled as:

$$R_A = -R_V \tag{2}$$

Although this baseline approach may produce highly disruptive adversarial policies, they are not realistic and do not translate to the safety-critical scenario of automated driving. Deploying the zero-sum adversarial reward function can lead to undesired and overly aggressive adversarial behaviors. Inspired by [1], we relax the zero-sum assumption and add a regularization reward R_{rs}, expanding Eq. 2 as:

$$R_A = -R_V + R_{rs}, \tag{3}$$

where R_{rs} encourages the adversarial policy to generate realistic and safety-driven trajectories by penalizing the collision (R_r) and the blocking of the parking area (R_s):

$$R_{rs} = R_r + R_s \tag{4}$$

To limit the disruptive adversarial collision strategy, the learning agent is penalized through a sparse contribution, R_r, in the following way:

$$R_r = \begin{cases} -1, & \text{for } A \text{ collision,} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Because of the large time horizon faced by the learning agent during a training episode, we add the dense contribution R_s which is given at every simulation step. We define this term as:

$$R_s = \begin{cases} -0.05, & \text{if } A \text{ is in } Ar_{tgt}, \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

where Ar_{tgt} refers to a predefined area, dynamically changing a position based on the location of the current target parking slot, as depicted in Fig. 2. R_s is introduced to prevent A from physically obstructing the victim's target parking slot.

3. Experimental results

The experimental part of the proposed work is conducted using the CARLA [2] simulator, an open-source software

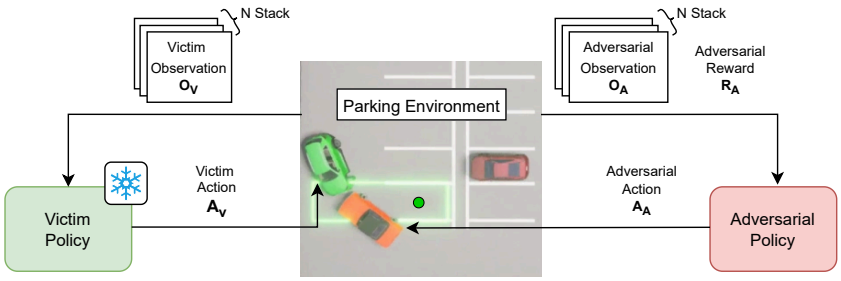


Figure 2. Proposed adversarial policy training pipeline and environment. Adversarial and victim action spaces are continuous: {throttle, brake, steering angle, reverse}. Both observations O_V and O_A are given as a sequence of N observations at a time for the current state s_t and the previous $[s_{t-1}, \cdots, s_{t-(N-1)}]$ states of the environment. For our experiments $N = 5$. The scene includes the victim agent (green), adversarial agent (orange), and a static environment vehicle (red), parked next to the target. The green dot is the target parking slot, and the green rectangle is Ar_{tgt}, in Eq. 6.

Table 1. Victim model robustness evaluation over 1000 episodes. The model is evaluated by measuring the success rate, mean alignment error (A_{err}), expressed in degrees, and the mean number of reverse gear changes per episode (N_{rev}).

Environment	**Success rate**	A_{err}	N_{rev}
Training	97%	4.41	1.61
Random agent	96%	6.35	1.71
Baseline Adversarial	**9%**	45.02	2.64
Safe Adversarial	**32%**	44.24	2.21

Table 2. Effect of R_{rs}. The model is evaluated by measuring the mean number of overall collisions (C_{all}) and collisions with E or the walls (C_E), collision rate against V (CR_V), and the mean of time steps when A is in Ar_{tgt} (T_{block}). The scores are the average of 1000 episodes.

Reward (R_A)	C_{all}	C_E	CR_V	T_{block}
$-R_V$	2.1	1.55	**87%**	9.43
$-R_V + R_{rs}$	**0.84**	**0.26**	62%	**0.71**

featuring realistic vehicle and environment dynamics, created to support the development of AD systems. Fig. 2 depicts the environment setup, populated by V (in green) along with a static E and A (in orange), both parked close the target. The green dot marks the target parking slot, the green rectangle refers to Ar_{tgt}, in Eq. 6.

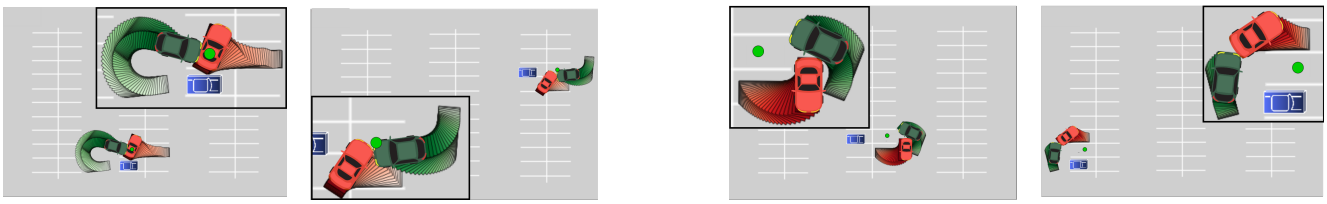
For the experimental settings, we deployed the pre-trained victim parking model, reaching a final success rate of 97% , in a scenario with 0-2 Es, parked around the randomly changing target location. Here, a parking attempt is considered successful if V reaches the target tgt. Alternatively, the episode terminates when a timeout occurs after 120 seconds or if V collides with any object or vehicle in the scene. At each episode, the alignment of the target is randomized as well, between front and rear parking.

3.1. Victim policy

We first evaluated the efficacy of our initial adversarial policy, trained with the reward function described in Eq. 2. The goal is to assess the robustness of the victim policy to extreme, not-safe-guarded attacks by A and generate a baseline to compare with a more realistic adversary. Table 1 shows the evaluation parameters for V in the scene. In this context, we do not explicitly report the collision rate of V, as it is simply the complement of the success rate, as no episode timeout was observed in the test phase. We tested the scenario for 1,000 episodes in the original training environment, along with a random agent and both against the baseline and a safe As. The success rate, alignment error and the number of gear changes shows how the adversary is able to disrupt the normal victim capabilities when compared to the non-adversarial environment. However, Fig. 3a shows that the sample adversarial trajectory generated by this preliminary phase is not indicative of a true failure of V policy, since the adversary's behavior neglects safety. The goal of A is solely achieved through physical target blocking and by actively seeking a direct collision with V, which is also reflected by the metrics presented in Table 2. This behavior would not be regarded as realistic in the AD context and is completely outside the ODD of a typical AP system. This spurred us to seek a finer reward.

3.2. Adversarial policy

To enhance the adversary behavior in terms of realism w.r.t. a typical AP system ODD, we introduced the R_{rs} safety term, presented in Eq. 3, to the training procedure of A. We did so by loading the pre-trained model weights from the first purely adversarial phase and continuing the training procedure after adding the R_{rs} contribution. We then,



(a) (b)

Figure 3. Trajectories of the victim (green) and adversarial (red) agents using the baseline (a) and the proposed (b) adversarial policy. The green dot is the parking target. (a) shows the adversarial agent simply *blocking the parking slot*, making the victim's parking impossible, while (b) shows the adversarial agent not intentionally colliding with the victim but exiting the parking slot.

again, evaluated both V and A policies simultaneously. Results are reported in both Table 1 and 2. When compared to the results of the first adversary, the re-trained A drastically reduced the mean number of collisions per episode against V and Es and walls surrounding the parking environment. The new adversarial policy also guided the vehicle outside of Ar_{tgt} and avoided target blocking completely. This results suggest that the behavior exhibited by A is more safety-aware and more realistic overall. This is also visible from the generated sample trajectory, shown in Fig. 3b, where A limits its disruptive behavior by exiting the parking slot at a certain angle, depending on the initial simulation scene, and then braking. This behavior allows to assess the response of V, which often led to a policy failure and not seldom to a collision caused by itself rather than actively by A, highlighting gaps on V generalization capability in such challenging scenarios. This is also confirmed by the collision rate of V, which is significantly higher than in the non-adversarial testing environment. Table 1 shows that, the change in the adversarial policy led V to have a higher success rate and lower collision rate then the baseline, which is negative from the adversary's point of view. However, the generated adversarial behavior and trajectories are more realistic, as shown by the analysis of both the metrics presented in Table 2 and the trajectory reported in Fig. 3b, and can be accounted for when deploying such end-to-end AP system in a real-world scenario.

4. Conclusion

This study applied a DRL-based framework to implement a safety-aware, realistic adversarial policy in an AP context, effectively uncovering vulnerabilities in the system under real-world, safety-bounded adversarial conditions that mimic corner-case scenarios. Such integration serves as a first step towards making AP systems more robust against potential corner cases, thus extending their ODD. However, the current adversarial behavior and corner-case generation system presents some limitations. For instance, the proposed scenario is limited to the interaction between the ego vehicle and a single adversarial vehicle agent. This is insufficient to properly model the variety of situations that could lead to victim failure cases. Moreover, the behavior of the adversarial agent is limited to exiting an adjacent parking slot, which limits the variety of the generated corner-case scenarios. To expand the investigation domain, future work will, therefore, include multi-agent systems composed of several vehicle and non-vehicle (e.g. pedestrian) agent guided by an adversarial policy. Furthermore, a proper assessment of the overall system's capabilities (also in comparison to other corner-case generation techniques) will require several iterations of victim and adversary model retraining.

References

[1] Baiming Chen, Xiang Chen, Qiong Wu, and Liang Li. Adversarial evaluation of autonomous vehicles in lane-change scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 23(8):10333–10342, 2022. 1, 2

[2] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017. 2

[3] Adam Gleave, Michael Dennis, Cody Wild, Neel Kant, Sergey Levine, and Stuart Russell. Adversarial policies: Attacking deep reinforcement learning. In *International Conference on Learning Representations*, 2019. 1, 2

[4] Zimin He, Jiawei Zhang, Danya Yao, Yi Zhang, and Huaxin Pei. Adversarial generation of safety-critical lane-change scenarios for autonomous vehicles. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 6096–6101, 2023. 1

[5] Philip Koopman and Michael Wagner. Autonomous vehicle safety: An interdisciplinary challenge. *IEEE Intelligent Transportation Systems Magazine*, 9(1):90–96, 2017. 1

[6] Luca Lazzaroni, Alessandro Pighetti, Francesco Bellotti, Alessio Capello, Marianna Cossu, and Riccardo Berta. Automated parking in carla: A deep reinforcement learning-based approach. In *Applications in Electronics Pervading Industry, Environment and Society*, 2024. 1, 2

[7] Qianqian Liu, Fengying Dang, Xiaofan Wang, and Xiaoqiang Ren. Autonomous highway merging in mixed traffic using reinforcement learning and motion predictive safety controller. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1063–1069, 2022. 1

[8] Shreyas Ramakrishna, Baiting Luo, Christopher B. Kuhn, Gabor Karsai, and Abhishek Dubey. Anti-carla: An adversarial testing framework for autonomous vehicles in carla. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2620–2627, 2022. 1

[9] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 2

[10] Kotaro Yamamoto, Rui Teng, and Kenya Sato. Simulation evaluation of vehicle movement model using spatio-temporal grid reservation for automated valet parking. *IEEE Open Journal of Intelligent Transportation Systems*, 4:261–266, 2023. 1