CRASH: Challenging Reinforcement-Learning Based Adversarial Scenarios For Safety Hardening

Amar Kulkarni, Shangtong Zhang, Madhur Behl

University of Virginia
{ark8su, shangtong, madhur.behl}@virginia.edu

Abstract

Ensuring the safety of autonomous vehicles (AVs) requires identifying rare but critical failure cases that on-road testing alone cannot discover. High-fidelity simulations provide a scalable alternative, but automatically generating realistic and diverse traffic scenarios that can effectively stress test AV motion planners remains a key challenge. This paper introduces CRASH - Challenging Reinforcement-learning based Adversarial scenarios for Safety Hardening - an adversarial deep reinforcement learning framework to address this issue. First CRASH can control adversarial Non Player Character (NPC) agents in an AV simulator to automatically induce collisions with the Ego vehicle, falsifying its motion planner. We also propose a novel approach, that we term safety hardening, which iteratively refines the motion planner by simulating improvement scenarios against adversarial agents, leveraging the failure cases to strengthen the AV stack. CRASH is evaluated on a simplified two-lane highway scenario, demonstrating its ability to falsify both rule-based and learning-based planners with collision rates exceeding 90%. Additionally, safety hardening reduces the Ego vehicle's collision rate by 26%. While preliminary, these results highlight RL-based safety hardening as a promising approach for scenario-driven simulation testing for autonomous vehicles.

1. Introduction

Autonomous vehicles (AVs) are increasingly being deployed in cities, showcasing the potential of AV technology (Kusano et al. 2023). Yet, these systems are far from perfect. Failures continue to occur, ranging from AVs getting stuck in traffic leading to frustrated commuters; all the way to catastrophic events with life-threatening consequences. A stark example is the recent Cruise pedestrian collision in San Francisco, where a pedestrian was struck by a human driven vehicle and flung in to the path of the AV which further dragged the pedestrian, leading to public outrage and the suspension of Cruise's operating license (Exponent 2024). These failures are not isolated incidents-they expose a deeper issue. AVs struggle with rare and complex situations, known as "long-tail" events, which lie beyond the scope of everyday driving scenarios. Compounding the problem is a pervasive lack of transparency. When
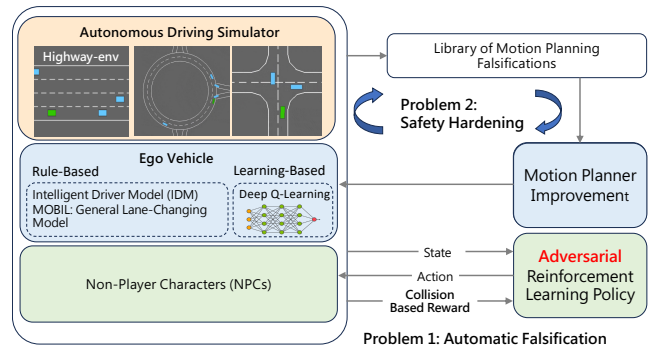
Figure 1: **CRASH** is a framework that iteratively tests and improves an AV motion planner. First, Automatic Falsification uses adversarial NPCs, guided by a reinforcement learning policy designed to induce collisions with the Ego vehicle. Then, Safety Hardening retrains the Ego motion planner to enhance its robustness against these adversarial scenarios.

accidents occur, little to no information is shared about their root causes and whether meaningful steps are being taken to prevent similar failures in the future. Reports from the California DMV and NHTSA (Team 2021; NHTSA 2024) provide limited insights, often describing only surface-level details. The current approach to AV safety development relies heavily on manual post-incident analysis, where developers retrospectively analyze failure scenarios and craft expert-designed rules to mitigate similar incidents. This process is not only reactive but also highly inefficient, requiring vast amounts of driving time to encounter rare safety-critical situations that expose system vulnerabilities (Kalra and Paddock 2016). Moreover, the reliance on on-road testing raises ethical concerns, as cities and public roads are effectively used as testing grounds, putting lives at risk.

Consequentially, simulation has become a critical tool for addressing challenges in AV testing, providing a controlled environment for exploring rare safety-critical scenarios without risking lives. However, generating meaningful scenarios is complex and requires addressing three major challenges: *saliency*, ensuring scenarios are realistic and relevant by avoiding unrealistic dynamics or overly simple interactions that fail to uncover weaknesses; *curse of dimensionality*, managing the vast configuration space of vari-

ables like vehicle behaviors, road layouts, and environmental factors, which grow exponentially and complicate prioritization; and *curse of rarity*, efficiently identifying critical long-tail scenarios that are infrequent but essential for comprehensive testing (FARS 2023). Simulation must overcome these challenges of scenario generation to effectively target and evaluate meaningful edge cases.

A targeted approach is needed to simulate scenarios where AVs are likely to fail. While researchers have explored methods such as perturbing logged trajectories, recreating real-world accidents in simulation, and importance sampling to search for edge cases, these techniques often focus solely on identifying failure scenarios. Critically, they do not incorporate these scenarios into the process of improving the AV's safety.

To address this gap, we propose CRASH, a novel framework that combines Automatic Falsification and Safety Hardening to systematically test and improve AV motion planners (shown in Figure 1). CRASH not only identifies failure-inducing scenarios through adversarial reinforcement learning but also integrates them into an iterative process to enhance the Ego vehicle's robustness. This work focuses exclusively on motion planning, excluding considerations of perception in scenario generation. The contributions of this paper are as follows:

1. We develop a DQN-based method to train NPC agents capable of automatically falsifying both rule-based and learning-based Ego motion planners.

2. We introduce the novel concept of Safety Hardening, formulated as a bi-level optimization, and we investigate three methods to improve the safety of Ego motion planners, including local hardening and model pool-based methods with uniform and prioritized sampling.

Above all, this work attempts to lay the foundation for a unified methodology that addresses inefficiency and limitations in failure scenario generation and establishes a pathway for systematically improving AV safety through simulation.

2. Related Work

AV safety is a multifaceted problem, but surprisingly, there are no agreed-upon safety metrics. Standards like ISO 26262 (iso 2018) focus on functional safety, addressing risks from software errors, and hardware faults, but they do not account for the planning or decision-making capabilities of AVs. Similarly, UL 4600 (ULSE 2020) provides a framework for creating safety cases but struggles with the challenge of enumerating all possible traffic scenarios. Formal methods for defining AV safety face challenges from the uncertainty and complexity of traffic environments (Mehdipour et al. 2023), making scenario parametrization tedious and verification impractical for dynamic, unbounded scenarios. Rather than providing formal guarantees, we focus on evaluating safety through collision rates, a widely accepted and practical metric in autonomous driving (Westhofen et al. 2021).

Many simulation-based scenario generation studies rely on naturalistic driving data (NDD) to construct datasets of realistic scenarios, which can then be searched for dangerous events (Sun et al. 2021; Neelofar and Aleti 2022). NDD

also aids in building learning-based behavior models for vehicles and pedestrians (Yan et al. 2023). The Naturalistic Adversarial Driving Environment (NADE), combines decision trees generated from NDD with deep reinforcement learning (DRL) to generate rare scenarios (Feng et al. 2023, 2021). While NDD ensures realism, it limits the scope of generated scenarios to maneuvers observed in the dataset, potentially missing the full dynamic capabilities of AVs. In contrast, our method directly generates collision trajectories, avoiding reliance on pre-existing data.

Actively probing systems for vulnerabilities is well-researched in autonomous driving. For instance, Robust Adversarial Reinforcement Learning (RARL) applies adversarial disturbances to improve controller robustness (Pinto et al. 2017), with variants like Risk Averse RARL (Pan et al. 2019) focusing on vehicle control, whereas we focus on motion planning.

Table 1: Comparison of CRASH with closest related work

Feature	NADE	RARL	STRIVE	CRASH
Adversarial Generation	✓	✓	✓	✓
Iterative Improvement	✗	✗	✗	✓
Focus on Motion Planning	✓	✗	✓	✓
Naturalistic Data Dependence	High	None	High	None

Methods in adversarial trajectory generation control NPCs using techniques such as Particle Swarm Optimization (Turlej and Pankiewicz 2021) or Monte Carlo Tree Search (Klischat and Althoff 2023). STRIVE optimizes adversarial trajectories in the latent space of VAE model (Rempe et al. 2022). While some approaches employ DRL, they typically stop after a single iteration of AV improvement (Sharif and Marijan 2022; Karunakaran et al. 2022). While these focus solely on falsification, CRASH iteratively discovers failure cases and uses them to enhance AV robustness. To the best of our knowledge, this is the first framework to integrate both adversarial scenario generation and safety improvement into a unified methodology (Table 1).

3. Problem Setup and Formulation

Problem Setup: We model the traffic scenario as a Markov Decision Process (MDP), a framework well-suited for decision-making in dynamic, stochastic environments such as traffic. The MDP is described by the tuple (S, S_0, A, P, R), where S is the state space, S_0 is the set of intial states, A the action space, P the transition model, and R the reward function. The probabilistic transition relation $P : S \times A \times S \to \mathbb{R}$ defines the probability of transitioning from one state to another given an action, governed by the simulator dynamics. The state of each vehicle $s_t^i \in S_t$ at time t is represented by its position (x_t, y_t), heading ψ_t, and velocity v_t along the heading direction. The scenario consists of $N + 1$ vehicles, where the Ego vehicle V^0 is controlled by the motion planner, and the remaining N vehicles are Non-Player Characters (NPCs), denoted by V^i. Figure 2(a) illustrates a reference coordinate system for a 2-lane, 2-agent scenario, showing the positions and headings of the Ego and an NPC vehicle. Figure 2(c) shows the example of the set of intial states S_0 for the same configuration comprising of 8 possible initial configurations between the Ego and
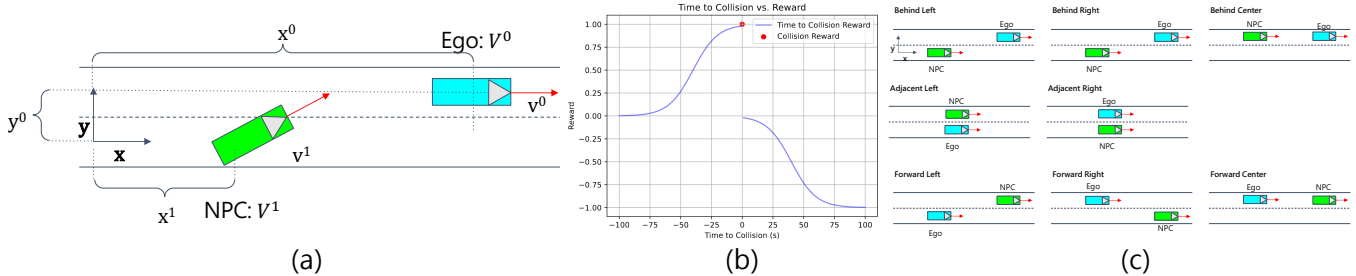
Figure 2: (a) Coordinate system: the blue vehicle (V_0) is the Ego, and the green vehicle (V_1) is an NPC on a 2-lane highway. (b) Reward function: positive when the NPC approaches the Ego, negative when moving away. (c) Initial configurations: the NPC (green) spawns at 8 relative positions around the Ego (blue).

NPC vehicles, based on their relative positions. The motion planning for the Ego agent V^0, can be dictated by either a rule-based motion planner, or a learning-based motion planner (e.g. DQN) For the learning-based planner, the reward function encourages collision avoidance and forward motion (Leurent 2018). This setup provides the foundation for formulating the two key problems addressed in this paper.

Problem 1: Automatic Falsification The goal of automatic falsification is to discover failure-inducing scenarios by optimizing the behavior of NPC agents to maximize collisions with the Ego vehicle. This involves finding an optimal policy $\pi^*_{\text{NPC}}(S_t, A_t)$ that maps states S_t to actions A_t to maximize the cumulative collision count:

$$\pi^*_{\text{NPC}}(S_t, A_t) = \arg\max_{A_t}(\sum_{j=0}^{E} \phi) \quad (1)$$

Here, ϕ is a binary indicator variable set to 1 whenever a collision is detected by the simulator and 0 otherwise, and E denotes the total number of episodes. The simulator evaluates interactions between the NPC agents and the Ego vehicle, updating ϕ for each collision.

Problem 2: Safety Hardening The outcome of a successful automatic falsification is an adversarial NPC agent that can falsify the Ego vehicle and result in several collisions. Given an adversarial NPC agent, the objective is to try and find an optimal policy $\pi^*_{\text{Ego}}(S_t, A_t)$ for the Ego, that minimizes the cumulative number of crashes across E episodes.

$$\pi^*_{\text{Ego}}(S_t, A_t) = \arg\min_{A_t}(\sum_{e=0}^{E} \phi) \quad (2)$$

Notably, the optimization objectives for the Ego and NPC policies, π_{Ego} and π_{NPC}, are dual problems: while the NPC aims to maximize collisions (Eq. (1)), the Ego aims to minimize them (Eq. (2)). Safety hardening combines these dual objectives into a unified framework as a bi-level optimization problem, where the Ego and NPC policies are optimized sequentially. This is represented as:

$$\min_{\sum_{e=0}^{E} \phi} \max_{\sum_{e=0}^{E} \phi} f(\pi^*_{\text{Ego}}(S_t, A_t), \pi^*_{\text{NPC}}(S_t, A_t)) \quad (3)$$

Here, f is a function that evaluates both policies, with the NPC attempting to exploit the Ego's weaknesses while the Ego adapts to improve its robustness. This setup is similar to the adversarial training structure of Generative Adversarial Networks (GANs) (Goodfellow et al. 2014), where one agent (NPC) generates adversarial scenarios and the other (Ego) learns to counteract them.

We make the following assumptions in our problems: The state transition model P_t is treated as a black-box, governed by the simulation's underlying dynamics. The state S_t^i of all vehicles is assumed to be fully known, without sensor noise or uncertainty. The set of all possible initial states is finite and enumerable. Finally, the Ego vehicle can be controlled using either a rule-based or learning-based motion planner, with the latter assumed to be re-optimizable and re-trainable.

4. CRASH: Adversarial Falsification and Safety Hardening

CRASH operates in two key stages: Automatic Falsification, where we train Non-Player Character (NPC) agents to discover failure-inducing scenarios for the Ego vehicle, and Safety Hardening, where the Ego motion planner is iteratively retrained to counter these adversarial scenarios, improving its robustness. We describe each of these next.

4.1 DQN Based Automatic Falsification

In simulation-based autonomous vehicle research, reinforcement learning is typically employed to improve the behavior of the Ego vehicle. However, in the context of falsification, we shift focus: rather than controlling the Ego, we are given an Ego motion planner and aim to train an NPC agent to learn a policy that maximizes collisions with the Ego. This adversarial setup forms the foundation of the automatic falsification in CRASH, where we leverage Deep Q-Networks (DQN) (Mnih et al. 2015) to train the NPC.

Adversarial Reward Design: To incentivize the NPC to collide with the Ego, we design an adversarial reward function tailored to this objective. The total reward R_t combines multiple components to guide the NPC's behavior:

$$r_t^i(s_t^i, a_t^i) = w_1 r_c + w_2 r_x + w_3 r_y \quad (4)$$

Here, r_c, r_x, and r_y serve distinct roles: r_c is the boolean collision reward set to 1 when a collision ϕ occurs, providing
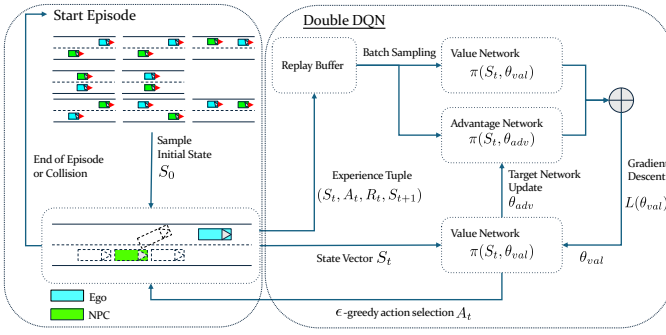
Figure 3: **Automatic Falsification in CRASH**: Training architecture for the adversarial NPC agent using Double DQN: simulator states (S_t) are input to the value network, which selects actions via the ϵ-greedy method
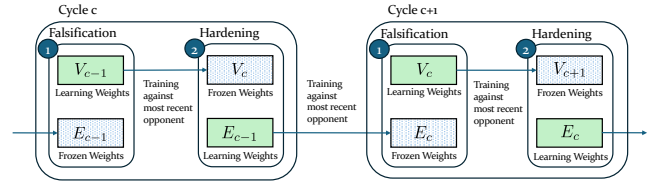

Figure 4: Local Safety Hardening over two cycles: In cycle c, the NPC V_{c-1} (green) is trained against the fixed Ego E_{c-1} (blue), producing V_c. Then, V_c is fixed, and E_{c-1} is trained to produce E_c. This process repeats in cycle $c+1$

the primary signal for adversarial behavior. However, since collisions terminate the episode, r_c is a sparse reward, presenting challenges for reinforcement learning (Rengarajan et al. 2022). To address the sparsity of r_c, we incorporate continuous rewards based on time-to-collision (TTC) along the longitudinal (x) and lateral (y) axes. These components, r_x and r_y, reward the NPC for decreasing the distance to the Ego while penalizing motion away from it. The TTC, λ, is computed as: $\lambda_x = \frac{\Delta x}{\Delta v_x}$ and $\lambda_y = \frac{\Delta y}{\Delta v_y}$. Unlike traditional TTC methods that focus only on positive values (collision proximity), we use the sign of λ to encode the NPC's approach direction. This enhances the NPC's spatial awareness relative to the Ego.

To ensure smooth gradients for learning, r_x and r_y are computed using sigmoid functions applied to $\lambda_{t,x}^i$ and $\lambda_{t,y}^i$.

$$r_x = \frac{-\text{sign}(\lambda_{t,x}^i)}{1 + e^{a - b\lambda_{t,x}^i}} \quad r_y = \frac{-\text{sign}(\lambda_{t,y}^i)}{1 + e^{a - b\lambda_{t,y}^i}} \quad (5)$$

Here, a and b are scaling parameters tuned to balance the reward's sensitivity to distance and velocity. The weights w_1, w_2, and w_3 govern the relative importance of each reward component and are tuned through reward shaping. Figure 2(b) illustrates the continuous TTC-based rewards.

DQN-Based NPC Training: With the adversarial reward function, R_t, defined, we train the NPC to approximate the optimal action-value function $Q^*(S_t, A_t)$. For stability and efficiency, we use Double Q-Learning (van Hasselt, Guez, and Silver 2015) with prioritized experience replay (Schaul et al. 2016). The NPC training architecture, shown in Figure 3, consists of two networks: A value network parameterized by θ_{val} to estimate $Q(S_t, A_t; \theta_{\text{val}})$; and a target network parameterized by θ_{adv} for stable updates. The value network is trained by minimizing the Bellman loss:

$$\begin{aligned} L(\theta_{val}) = (R_t &+ \gamma \max_a Q^*(S_{t+1}, A_{t+1}; \theta_{\text{val}}) \\ &- Q(S_t, A_t; \theta_{\text{adv}}))^2 \end{aligned} \quad (6)$$

To address overestimation bias in Q-learning, the target network parameters, θ_{adv}, are updated using an exponential

moving average: $\theta_{\text{adv}} \leftarrow \tau\theta_{\text{val}} + (1 - \tau)\theta_{\text{adv}}$ where τ is the smoothing factor.

The training process begins by uniformly sampling initial states S_0 to prevent overfitting to a narrow set of scenarios. At each time step t, the simulator provides the current state S_t, which is fed into the value network to compute Q-values for all possible actions. An action A_t is selected using an ϵ-greedy strategy to balance exploration and exploitation, executed in the simulator, and results in a reward R_t and next state S_{t+1}. The experience tuple (S_t, A_t, R_t, S_{t+1}) is stored in a prioritized replay buffer. Once the buffer reaches a certain capacity, experiences are sampled to update the value network using the loss function in Equation (6). During evaluation, actions are selected greedily based on the highest Q-value, without exploration. Trajectories resulting in collisions are stored as falsifications, collected during both training and evaluation. This approach allows the NPC to systematically discover failure modes in the Ego motion planner.

4.2 Safety Hardening

Building on the automatic falsification process, Safety Hardening aims to enhance the Ego agent's safety by iteratively training it against the adversarial NPC policies. After training the optimal NPC $\pi_{\text{NPC}}^*(S_t, A_t)$ to cause collisions against the current Ego policy, $\pi_{\text{Ego}}(S_t, A_t)$, we train the Ego policy $\pi_{\text{Ego}}^*(S_t, A_t)$ to mitigate these failures. We sequentially train π_{NPC} and π_{Ego}, one at a time, over a fixed number of cycles C. Each cycle comprises of a Falsification simulation, during which the NPC is trained to improve π_{NPC} by updating the NPC policy parameters $\theta_{\text{npc},c}$. This is followed by a Hardening simulation where the weights of the NPC DQN are frozen and the Ego policy parameters $\theta_{\text{ego},c}$ are updated to improve π_{Ego} to learn to avoid collisions against an adversarial NPC. We denote $V_c = \pi_{\text{NPC}}(S_t, A_t; \theta_{\text{npc},c})$ as the NPCs trained in the falsification simulation in cycle c and $E_c = \pi_{\text{Ego}}(S_t, A_t; \theta_{\text{ego},c})$ as the Ego trained during the hardening simulation in cycle c.

Local Safety Hardening The simplest approach to Safety Hardening is Local Safety Hardening, where each NPC is trained exclusively against the most recent Ego agent from the previous cycle. As illustrated in Figure 4, during cycle c, the NPC agent V_c is trained in a Falsification simulation to maximize collisions with E_{c-1}, whose weights remain fixed. Following this, E_c is trained during a Hardening simulation to minimize collisions with V_c, whose weights are

now frozen. This alternation between falsification and hardening repeats for each cycle. While straightforward, Local Safety Hardening can lead to overfitting, as agents are optimized only against their immediate counterparts, limiting their ability to generalize to a broader range of opponents.

Uniform Model Pool-Based Safety Hardening: To address the overfitting issue inherent in Local Safety Hardening, we introduce Model Pool-Based Safety Hardening, which maintains a pool of all previously trained agents. Specifically, we maintain two model-pools - one for the set of all previously trained Egos $E^P = \{E_c\}, \forall c \in C$ and another for the set of all NPCs $V^P = \{V_c\}, \forall c \in C$. After training, each agent's weights are frozen, and it is added to the respective pool to preserve its behavior. During Falsification and Hardening simulations, agents are sampled from the model pool to serve as opponents for the current episode. The key question is: what sampling criteria should be used? As the name suggests, uniform model pool-based sampling selects opposing agents uniformly from the corresponding pool. Figure 5 depicts this process. For NPC training at cycle c, agents are sampled from the Ego pool E^P_{c-1} using uniform probability, ensuring exposure to a diverse range of opponents and mitigating the risk of forgetting how to be effective against previous agents, whether that is for falsification in the case of NPCs, or for hardening in the case of the Ego vehicle. Similarly, for Ego training at cycle c, agents are sampled uniformly from the NPC pool V^P_c. The uniform model pool-based sampling approach promotes generalization by exposing agents to varied strategies and reducing the risk of overfitting to a single opponent.

Prioritized Model Pool-Based Safety Hardening: While uniform sampling improves generalization, it effectively trains an agent to handle the average skill level of the pool rather than trying to iteratively building stronger agents. To address this, the prioritized model pool-based sampling can be employed, where agents are sampled based on their skill levels, determined by an Elo rating system (Elo 1967). Each new NPC or Ego enters the pool and starts with an Elo rating of $R = 1000$. The expected score $\mathbb{E}_{\text{Agent}}$ for an agent is computed using the rating difference with its opponent:

$$\mathbb{E}_{\text{Agent}} = \frac{1}{1 + \exp\left(\dfrac{R_{\text{Opponent}} - R_{\text{Agent}}}{\zeta}\right)} \quad (7)$$
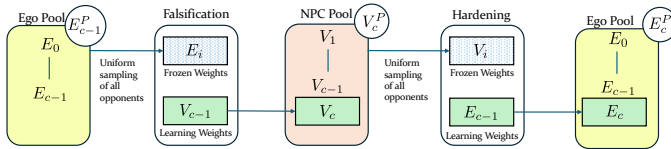
Figure 5: Uniform Sampling for model pool-based safety hardening: The NPC V_{c-1} is trained against Ego agents sampled uniformly from the pool E^P_{c-1}, and the resulting NPC, V_c, is added to the NPC pool V^P_c. The newest Ego is then trained against NPCs sampled uniformly from V^P_c.
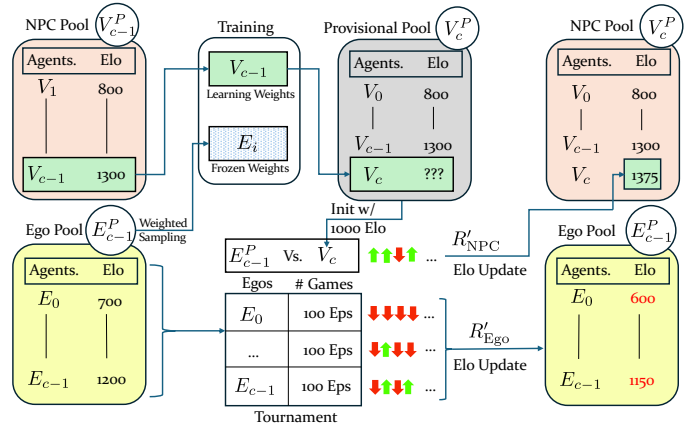
Figure 6: Example of cth safety hardening cycle with prioritized sampling and tournament evaluation: (1) NPC V_{c-1} is trained against E^P_{c-1}, producing V_c. (2) V_c is initialized with 1000 Elo and competes in a round-robin tournament with E^P_{c-1} to update scores. (3) V_c is added to the updated NPC pool V^P_c.

Here ζ is a scaling factor to control the sensitivity to rating differences. The agent score is updated each episode, where NPCs win if a collision happens, and Egos win if not. An agents score is updated iteratively $R'_{\text{Agent}} = R_{\text{Agent}} + K(\phi - \mathbb{E}_{\text{Agent}})$, where K is a gain. The sampling probability for each agent in the model pool is determined based on its expected score. For agent A_i, the probability $P(A_i)$ of being selected is given by:

$$P(A_i) = \frac{w(A_i)}{\sum_j w(A_j)}, \quad \text{where} \quad w(A_i) = \left[\mathbb{E}_{A_i}\right]^{\beta} \quad (8)$$

Here, β controls prioritization strength, with higher values favoring stronger agents. This method ensures that NPCs and Egos are trained against the most challenging opponents, promoting improvement. In our traffic scenario, outcomes are binary: win or a loss. To ensure Elo ratings accurately reflect performance, we define distinct win criteria for NPCs and Ego based on their respective objectives.

Tournament Based Evaluation To mitigate biases introduced during training due to exploration (e.g., ϵ-greedy actions), we implement a Tournament-Based Evaluation system. After training each new NPC or Ego agent, a round-robin tournament is conducted where the new agent competes against all agents in the opposing model pool (e.g., the new NPC V_c competes against all Egos in E^P). During these tournaments, agents act greedily without exploration, ensuring that their performance reflects true proficiency. Elo ratings are updated based on the tournament results, providing an accurate assessment of agent skill. Figure 6 illustrates this process for NPC evaluation. This procedure is similarly applied during Ego training, ensuring that the updated pools V^P and E^P reflect accurate agent rankings and behaviors.
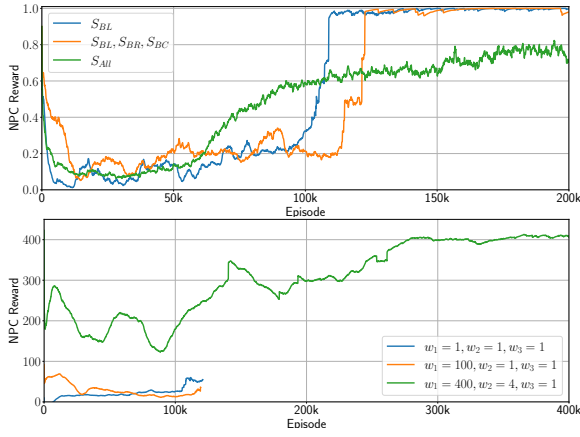
Figure 7: Accumulated rewards for NPC across sparse [Top] and continuous [Bottom] reward shaping experiments.

5. Results

5.1 Experiment Setup

We use the `highway-env` simulator (Leurent 2018), a gymnasium environment for testing RL algorithms in traffic scenarios. The setup consists of 1 Ego vehicle (V^0) and 1 NPC (V^1) on a two-lane highway, controlled by either a rule-based or learning-based planner. Each episode starts from one of 8 initial configurations (Figure 2(c)), chosen uniformly. The rule-based planner for the Ego uses the Intelligent Driver Model (IDM) (Treiber, Hennecke, and Helbing 2000) for longitudinal control and MOBIL (Kesting, Treiber, and Helbing 2007) for lane changes. The learning-based planner employs a DQN, while NPC agents use the reward function in Eq (4) to prioritize causing collisions and minimizing time-to-collision. Both value and advantage networks are 3-layer perceptrons with 256 hidden units. The Adam optimizer (Kingma and Ba 2014) is used with a learning rate of 5×10^{-4}. We primarily measure the average Crash Rate(CR) over the last 100 episodes $CR = \sum_{100}^{i=0} \frac{\phi_i}{100}$, as well as the accumulated reward over each episode.

5.2 Automatic Falsification Results

To evaluate the effectiveness of reward shaping for the NPC adversarial agent, we examined its performance using sparse and continuous rewards to falsify the rule-based Ego motion planner. Figure 7 illustrates the accumulated rewards for the NPC across these two experiments: sparse rewards [Top] and continuous rewards [Bottom]. Initially, only the sparse reward is used, where w_2 and w_3 in Eq (4) are set to zero. We test the sparse reward's effectiveness under varying initial configurations: (1) the NPC spawns behind and to the left of the Ego ($S_0 = S^{BL}$), (2) the NPC spawns in any "behind" state ($S_0 = S^{BL}, S^{BR}, S^{BC}$), and (3) the NPC spawns uniformly across all possible states. Results in Figure 7[Top] show that in simple scenarios, where the NPC spawns behind the Ego, it quickly maximizes the collision reward by speeding up and causing a rear collision. However, as more initial states are introduced, the learning rate drops significantly, indicating that sparse rewards alone are
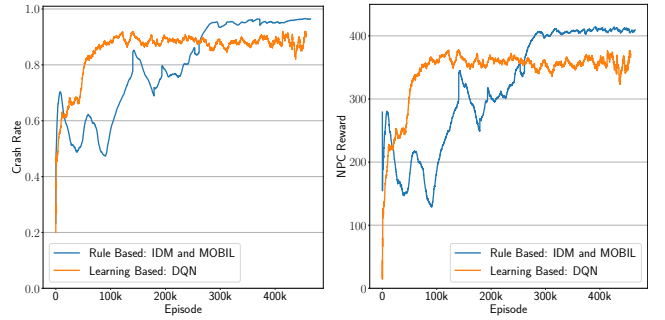

Figure 8: Average reward and crash rates for Automatic Falsification of rule-based and learning-based planners.

insufficient for the NPC to learn effective falsification strategies across diverse scenarios. Figure 7[Bottom] presents results when the continuous time-to-collision reward is integrated into the reward function, with varying weights w_1, w_2, and w_3 governing the emphasis on sparse rewards (w_1), longitudinal rewards (w_2), and lateral rewards (w_3). These trials reveal that the reward weights $w_1 = 400$, $w_2 = 4$, and $w_3 = 1$ enable the most effective learning for the NPC across all scenarios. The weights are thus retained for all subsequent experiments.

Using the tuned reward function, we trained the NPC to falsify both the rule-based and learning-based Ego motion planners. Ten trials, each with 10M transitions, were conducted, and the average crash rate and accumulated reward are shown in Figure 8 [Left] and [Right], respectively. As the reward saturates, the NPC achieves an average crash rate of 97% against the rule-based (IDM & MOBIL) and 90% against the learning-based (DQN) planner. These results highlight the effectiveness of the NPC reward function and the DQN implementation in Section for automatic falsification of both planners in the two-lane, two-agent scenario.

5.3 Safety Hardening Results

To evaluate Safety Hardening, we train the Ego and NPC policies iteratively over five cycles. Initially, the NPC policy V_1 is trained against the baseline Ego policy E_0. Subsequently, the Ego policy E_1 is retrained to counter V_1, and this process alternates through five cycles, with V_c being trained against E_{c-1} and E_c against V_c. We apply this training procedure to evaluate the three different Safety Hardening strategies: (1) Local Safety Hardening, (2) Uniform Model Pool-Based Safety Hardening, and (3) Prioritized Model Pool-Based Safety Hardening. Table 2 summarizes the results of these experiments. Each row corresponds to an Ego agent (E_c), showing its average crash rates evaluated over 100 episodes against the directly trained NPC counterpart (V_c) and all previously trained NPCs (V_{1-4}), for each method. Similarly, each NPC (V_c) is evaluated against its immediate Ego counterpart (E_{c-1}) as well as all previously trained Egos (E_{1-4}). We first examine the trends in local falsification and hardening. In the first cycle, V_1 is trained against E_0, resulting in a crash rate of 86% ↑. Once V_1's weights are frozen, E_1 is trained against V_1, reducing

|	V$_1$			V$_2$			V$_3$			V$_4$			V$_5$			Mean		
	Local	Uniform	Prioritized	Local	Uniform	Prioritized	Local	Uniform	Prioritized	Local	Uniform	Prioritized	Local	Uniform	Prioritized	Local	Uniform	Prioritized
E$_0$	0.86 ↑	0.80	0.83	0.79	0.74	0.67	0.36	0.40	0.37	0.28	0.67	0.49	0.69	0.57	0.44	0.60	**0.64**	0.56
E$_1$	0.40 ↓	0.32	0.24	0.87 ↑	0.38	0.76	0.61	0.47	0.38	0.64	0.49	0.58	0.38	0.48	0.43	0.58	0.43	0.48
E$_2$	0.32	0.24	0.38	0.29 ↓	0.31	0.27	0.67 ↑	0.44	0.44	0.17	0.47	0.36	0.21	0.44	0.32	0.33	0.38	0.35
E$_3$	0.66	0.38	0.56	0.75	0.32	0.75	0.32 ↓	0.38	0.28	0.70 ↑	0.58	0.63	0.50	0.57	0.36	0.59	0.45	0.52
E$_4$	0.83	0.33	0.55	0.80	0.36	0.55	0.74	0.37	0.52	0.17 ↓	0.24	0.15	0.77 ↑	0.34	0.63	0.66	0.33	0.48
E$_5$	0.62	0.31	0.52	0.58	0.43	0.92	0.61	0.28	0.56	0.58	0.50	0.63	0.20 ↓	0.39	0.26	0.52	**0.38**	0.58
Mean	0.62	0.40	0.51	0.68	0.42	0.65	0.55	0.39	0.43	0.42	0.49	0.47	0.46	0.47	0.41	0.49	0.43	0.49

Table 2: Crash Rates of each Ego E evaluated against each NPC V for 100 episodes over 5 cycles of Falsification and Safety Hardening. Each method of Local Safety Hardening, Uniform model pool based sampling, and Prioritized model pool based sampling is shown.

the crash rate to 40% ↓, demonstrating effective hardening. Subsequently, V_2 is trained against E_1 (with E_1's weights frozen), increasing the crash rate to 87% ↑, indicating successful falsification. This is followed by E_2 reducing the crash rate to 29% ↓ during hardening. These trends persist across cycles, with crash rates consistently increasing during falsification (V_c vs. E_{c-1}) and decreasing during hardening (E_c vs. V_c). However, examining the final Ego agent, E_5, reveals limitations of local training. While E_5 achieves a low crash rate of 20% ↓ against the most recent NPC, V_5, its crash rate against NPCs from earlier cycles (V_{1-4}) remains significantly higher ($> 58\%$). This indicates that repeated training cycles did not build upon the skills learned in earlier iterations, and newly trained Egos tend to forget how to counter older NPCs. The mean crash rate further shows no consistent decrease across NPC versions, highlighting the inability of local falsification to generalize across iterations.

Figure 9 illustrates the average crash rates for each Ego and NPC across cycles. Uniform model pool sampling exhibits the most consistent improvement, achieving a 26% reduction (from 64% to 38%) in the average crash rate for the Ego across five cycles, outperforming both local falsification and prioritized model pool sampling. For NPC crash rates (Figure 9[Right]), no clear trend emerges to indicate which method is most effective. However, uniform model pool sampling shows the highest overall increase in NPC crash rates across the five cycles, suggesting it balances the effectiveness of both adversarial falsification and hardening better than the other methods.

6. Conclusions and Discussion

In this work, we introduce CRASH, a novel framework that tackles the dual challenges of automatic falsification and safety hardening for AV motion planners. For automatic falsification, we demonstrate that a DQN-based adversarial NPC policy can effectively falsify both rule-based (crash rate 97%) and learning-based Ego motion planners (crash rate 90%) in a two-vehicle, two-lane highway scenario. Additionally, we propose three methods for iterative safety hardening, showing that the uniform model pool-based approach outperforms both local hardening and prioritized sampling. Specifically, it achieves a 26% reduction in Ego crash rates over five hardening cycles, highlighting its effectiveness in mitigating overfitting and improving the generalization of trained agents. In conclusion, this preliminary work shows the potential of the CRASH framework
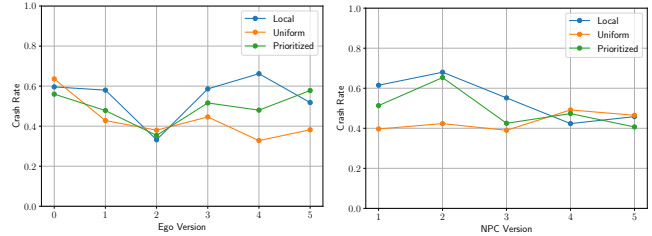
(a) Mean Crash Rate across Ego Versions

(b) Mean Crash Rate across NPC Versions

Figure 9: The average crash rate across each variant of Safety Hardening. In the left plot, the average crash rate of all NPCs evaluated against each Ego version is shown. In the right plot, it is the average crash rate of all Egos evaluated against each NPC version.

in developing adversarial agents for safety testing in autonomous driving and highlights the rich problem space in implementing safety hardening.

Limitations and Future Work: While our results demonstrate the efficacy of CRASH, we acknowledge the simplicity of the two-vehicle, two-lane highway scenario. For instance, for this scenario, the set of eight initial states can be fully enumerated a priori. Therefore, we anticipate when we scale this approach to include more NPCs and more complex road geometries, we will need to relax the assumption associated with the knowledge of all the initial states. Another problem of interest with multiple NPCs would be whether they can collude to cause more sophisticated falsifications for the Ego vehicle. This attribute is not examined yet by our approach and would likely necessitate exploring multi-agent deep reinforcement learning methods that could train all NPCs at once. Secondly, our current definition for falsification is simply that a collision occurs between the NPC and the Ego vehicle but not all collisions are equally informative; future investigations will explore collision-specific rewards to focus on generating failures where the AV is clearly at fault, offering more actionable insights into planner vulnerabilities. Despite these simplifications, our approach provides a valuable foundation for systematically testing and improving AV motion planners, addressing key challenges in safety-critical scenario generation and mitigation.

References

2018. ISO 26262: Road vehicles - Functional safety. International Organization for Standardization.

Elo, A. E. 1967. The Proposed USCF Rating System, Its Development, Theory, and Applications. *Chess Life*, XXII(8): 242–247.

Exponent. 2024. Quinn Emanuel Report re Cruise. Technical report, Exponent. Prepared for Quinn Emanuel Urquhart & Sullivan, LLP.

FARS. 2023. Fatality Analysis Reporting System (FARS) — NHTSA — nhtsa.gov. https://www.nhtsa.gov/research-data/fatality-analysis-reporting-system-fars. [Accessed 06-11-2023].

Feng, S.; Sun, H.; Yan, X.; Zhu, H.; Zou, Z.; Shen, S.; and Liu, H. X. 2023. Dense reinforcement learning for safety validation of autonomous vehicles. *Nature*, 615(7953): 620–627. Number: 7953 Publisher: Nature Publishing Group.

Feng, S.; Yan, X.; Sun, H.; Feng, Y.; and Liu, H. X. 2021. Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment. *Nature Communications*, 12(1): 748. Number: 1 Publisher: Nature Publishing Group.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A. C.; and Bengio, Y. 2014. Generative Adversarial Nets. In *Neural Information Processing Systems*.

Kalra, N.; and Paddock, S. M. 2016. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation Research Part A: Policy and Practice*, 94: 182–193.

Karunakaran, D.; Berrio, J. S.; Worrall, S.; and Nebot, E. 2022. Critical concrete scenario generation using scenario-based falsification. ArXiv:2208.13329 [cs].

Kesting, A.; Treiber, M.; and Helbing, D. 2007. General Lane-Changing Model MOBIL for Car-Following Models. *Transportation Research Record*, 1999(1): 86–94.

Kingma, D. P.; and Ba, J. 2014. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980.

Klischat, M.; and Althoff, M. 2023. Falsifying Motion Plans of Autonomous Vehicles With Abstractly Specified Traffic Scenarios. *IEEE Transactions on Intelligent Vehicles*, 8(2): 1717–1730. Conference Name: IEEE Transactions on Intelligent Vehicles.

Kusano, K. D.; Scanlon, J. M.; Chen, Y.-H.; McMurry, T. L.; yu Chen, R.; Gode, T.; and Victor, T. 2023. Comparison of Waymo Rider-Only Crash Data to Human Benchmarks at 7.1 Million Miles. *ArXiv*, abs/2312.12675.

Leurent, E. 2018. An Environment for Autonomous Driving Decision-Making. https://github.com/eleurent/highway-env.

Mehdipour, N.; Althoff, M.; Tebbens, R. D.; and Belta, C. 2023. Formal methods to comply with rules of the road in autonomous driving: State of the art and grand challenges. *Automatica*, 152: 110692.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533.

Neelofar; and Aleti, A. 2022. Identifying Safety-critical Scenarios for Autonomous Vehicles via Key Features. ArXiv:2212.07566 [cs].

NHTSA. 2024. ADS Crash Report ID 12345. Technical report, U.S. Department of Transportation. Report retrieved from the Standing General Order on Crash Reporting.

Pan, X.; Seita, D.; Gao, Y.; and Canny, J. 2019. Risk Averse Robust Adversarial Reinforcement Learning. arXiv:1904.00511.

Pinto, L.; Davidson, J.; Sukthankar, R.; and Gupta, A. 2017. Robust Adversarial Reinforcement Learning. arXiv:1703.02702.

Rempe, D.; Philion, J.; Guibas, L. J.; Fidler, S.; and Litany, O. 2022. Generating Useful Accident-Prone Driving Scenarios via a Learned Traffic Prior. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Rengarajan, D.; Vaidya, G.; Sarvesh, A.; Kalathil, D.; and Shakkottai, S. 2022. Reinforcement Learning with Sparse Rewards using Guidance from Offline Demonstration. arXiv:2202.04628.

Schaul, T.; Quan, J.; Antonoglou, I.; and Silver, D. 2016. Prioritized Experience Replay. ArXiv:1511.05952 [cs].

Sharif, A.; and Marijan, D. 2022. Adversarial Deep Reinforcement Learning for Improving the Robustness of Multi-agent Autonomous Driving Policies. In *2022 29th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE.

Sun, H.; Feng, S.; Yan, X.; and Liu, H. X. 2021. Corner Case Generation and Analysis for Safety Assessment of Autonomous Vehicles. arXiv:2102.03483.

Team, S. D. C. 2021. Metadata record for: Crash and disengagement data of autonomous vehicles on public roads in California.

Treiber, M.; Hennecke, A.; and Helbing, D. 2000. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2): 1805–1824.

Turlej, W.; and Pankiewicz, N. 2021. Adversarial Trajectories Generation for Automotive Applications. In *2021 25th International Conference on Methods and Models in Automation and Robotics (MMAR)*, 115–120.

ULSE. 2020. *UL 4600: Standard for Evaluation of Autonomous Products*. Standard for safety. Underwriters Laboratories.

van Hasselt, H.; Guez, A.; and Silver, D. 2015. Deep Reinforcement Learning with Double Q-learning. arXiv:1509.06461.

Westhofen, L.; Neurohr, C.; Koopmann, T.; Butz, M.; Schütt, B.; Utesch, F.; Kramer, B.; Gutenkunst, C.; and Böde, E. 2021. Criticality Metrics for Automated Driving: A Review and Suitability Analysis of the State of the Art. *CoRR*, abs/2108.02403.

Yan, X.; Zou, Z.; Feng, S.; Zhu, H.; Sun, H.; and Liu, H. X. 2023. Learning naturalistic driving environment with statistical realism. *Nature Communications*, 14(1): 2037. Number: 1 Publisher: Nature Publishing Group.