

## 731 Appendix

### 732 Impact Statement

733 Our work introduces SHIFT, a novel approach to perturbation attacks in reinforcement learning (RL)  
734 by altering the semantics of the true states while remaining stealthy from both static and dynamic  
735 perspectives. SHIFT demonstrates outstanding performance, successfully compromising all state-of-  
736 the-art defense methods. This highlights the urgent need for more sophisticated defense mechanisms  
737 that are resilient to semantic uncertainties.

738 This research raises important safety concerns, as adversaries could exploit these semantic-changing  
739 attacks to cause significant harm in real-world RL applications, such as autonomous driving. The  
740 ability to alter the perception of critical systems like self-driving cars could lead to catastrophic  
741 consequences. As such, our findings underscore the necessity of further research into robust defenses  
742 capable of withstanding such advanced and subtle attack strategies.

### 743 A Related Work 18

744	A.1 State Perturbation Attacks and Defenses . . . . .	18
745	A.2 Attacks and Defenses Beyond State Perturbations . . . . .	18
746	A.3 Diffusion Models and RL . . . . .	19
747	A.4 Diffusion Models in Adversarial Examples . . . . .	19
748	A.5 Relation with Constrained Diffusion Model . . . . .	19
749	A.6 GAN-based Semantic Adversarial Attacks. . . . .	20

### 750 B Proof of Theorem 3.6 20

### 751 C Implementation Details and Algorithms 22

752	C.1 Two Stage Attacks Pipelines . . . . .	22
753	C.2 Visualization of SHIFT-O and SHIFT-I in the Freeway Environment and Doom . .	22
754	C.3 Score-Based Diffusion Model . . . . .	23
755	C.4 EDM Model and ODE formulation . . . . .	24
756	C.5 EDM as a conditional diffusion model . . . . .	25
757	C.6 Training and testing stage algorithms for SHIFT . . . . .	25
758	C.7 Hyper-parameters Setting . . . . .	26

### 759 D More Evaluation Results 27

760	D.1 Additional Attack Results in Atari Environments . . . . .	27
761	D.2 Attack Performance on Continuous Action Space Environment and PPO Policy . .	27
762	D.3 Detection Results . . . . .	28
763	D.4 Ablation on Realism Guidance . . . . .	28
764	D.5 Ablation on Policy Guidance Strength . . . . .	28
765	D.6 Ablation on Attack Frequency . . . . .	28
766	D.7 Ablation on DDPM and EDM diffusion architectures . . . . .	29
767	D.8 Performance and Stealthiness of High-Sensitivity Direction Attacks in Korkmaz [31]	29
768	D.9 Time Complexity Comparison . . . . .	31

## A Related Work

### A.1 State Perturbation Attacks and Defenses

State perturbation attacks on RL policies were first introduced in [24], where the *MinBest* attack was proposed to minimize the probability of selecting the best action by iteratively adding  $l_p$ -norm constrained noise calculated through  $-\nabla_{\tilde{s}_t} \pi(\pi(\cdot|s_t), \tilde{s}_t)$ . Building on this, Zhang et al. [59] showed that when the agent’s policy is fixed, finding the optimal adversarial policy can be framed as an MDP, and the attacker can find the optimal attack policy by applying RL techniques. This was further improved in [47], where a more efficient algorithm for finding optimal attacks, called *PA-AD*, was introduced. Instead of searching perturbed states in the original state space, *PA-AD* trained a director through RL to find the optimal attack direction, and the trained director directs the designed actor to generate perturbed states, which decreases the searching space of RL. More recently, illusory attack [16] is proposed by requiring a perturbed trajectory to follow the same distribution as the normal trajectory, making it difficult to detect. However, this approach does not scale to high-dimensional image input. Korkmaz [31] recognized the limitation of  $l_p$  norm constrained attacks and proposed a policy-independent attack by following high sensitive directions, leading to attacks such as changing brightness and contrast, image blurring, image rotation and image transform. These types of attacks are imperceptible when the amount of manipulation applied is small and can compromise SA-MDP defense. However, they can barely alter the domain-specific semantics of image input according to our Definition 3.3. For example, in the Pong game, the pong ball’s relative distance from the two pads will remain the same after changing brightness and contrast, or transforming the image. As a result, these attacks cannot bypass diffusion-based defenses as shown in Tables 7 and 8.

On the defense side, Zhang et al. [59] demonstrated that a universally optimal policy under state perturbations might not always exist. They proposed a set of regularization-based algorithms (SA-DQN, SA-PPO, SA-DDPG) to train robust RL agents. This was enhanced in [36], where a worst-case Q-network and state importance weights were incorporated into the regularization. A more recent work called CAR-DQN [35] shows that using an  $l_\infty$  norm can further improve the policy’s robustness, and they theoretically capture the optimal robust policy (ORP) under  $\epsilon$  constrained state perturbation attacks, although this method incurs high computational costs. Another line of work by [54] proposed an autoencoder-based detection and denoising framework to identify and correct perturbed states. Korkmaz and Brown-Cohen [32] proposed SO-INRD, which uses the local curvature of the cross-entropy loss between the action distribution  $\pi(a|s)$  given by a policy  $\pi$  and a target action distribution to detect adversarial directions. He et al. [20] showed that when the initial state distribution is known, it is possible to find a policy that optimizes the expected return under state perturbations. Diffusion-based defenses have also been utilized to generate more robust agent policies. DMBP [56] utilized a conditional diffusion model to recover actual states from perturbed states and Sun and Zheng [46] used the diffusion model as a purification tool to generate a belief set about the actual state and perform a pessimistic training to generate a robust policy. Nie et al. [40] found that the performance loss of a policy under an  $l_\infty$  norm-bounded state perturbations is bounded by the KL divergence between the action distribution under the true state and that under the perturbed state, and utilized a new network architecture called SortNet [58] to train a robust RL policy. More recently, a game-theoretical defense method (GRAD) [37] was proposed to address temporally coupled attacks by modeling the temporally coupled robust RL problem as a partially observable zero-sum game and deriving an approximate equilibrium of the game. Another important recent defense is PROTECTED [39], which iteratively searches for a set of non-dominated policies during training and adapts these policies during testing to address different attacks. However, both GRAD [37] and PROTECTED [39] focus on MuJoCo environments and are already computationally intensive (both take more than 20 hours) to train on the relatively simple environments. Without further adaptation, it will be computationally prohibitive to apply these two methods to environments with image input as we consider in this paper.

### A.2 Attacks and Defenses Beyond State Perturbations

As demonstrated by [25], altering the reward signal can significantly disrupt the training process of Q-learning, causing the agent to adopt a policy that aligns with the attacker’s objectives. Additionally, [62] introduced an adaptive reward poisoning technique that can induce a harmful policy in a number of steps that scales polynomially with the size of the state space  $|S|$  in the tabular setting. In

a similar vein, Zhang et al. [62] developed an adaptive reward poisoning method capable of achieving a malicious policy in polynomial steps based on the size of the state space  $|S|$ .

Moving beyond reward manipulation, Lee et al. [33] proposed two techniques for perturbing the action space. Among them, the *Look-Ahead Action Space* (LAS) method was found to deliver better performance in reducing cumulative rewards in deep reinforcement learning by distributing attacks across both the action and temporal dimensions. Another line of research focuses on adversarial policies within multi-agent environments. For example, [17] showed that in a zero-sum game, a player using an adversarial policy can easily beat an opponent using a well-trained policy.

Attacks targeting an RL agent’s policy network have also been explored. Inference attacks, as described by [10], aim to steal the policy network parameters. On the other hand, poisoning attacks, as discussed in [23], focus on directly manipulating the model parameters. Specifically, Huai et al. [23] proposed an optimization-based method to identify an optimal strategy for poisoning the policy network.

### A.3 Diffusion Models and RL

Diffusion models have recently been utilized to solve RL problems by exploiting their state-of-the-art sample generation ability. In particular, diffusion models have been utilized to generate high quality offline data in solving offline RL problems. Offline RL training is known as a data-sensitive process, where the quality of the data has a huge influence on the training result. To deal with this problem, many studies [19, 2, 27] have shown that diffusion models can learn from a demo dataset and then generate high reward trajectories for learning or planning purposes. In addition, conditional diffusion models have been directly used to model RL policies. A conditional diffusion model can generate actions through a denoising process with states and other useful information as conditions. Several studies [28, 48] have shown state-of-the-art performance in various offline RL environments when using a diffusion model as a policy, which leads to a promising research direction.

Furthermore, Black et al. [8] shows that the denoising process can be viewed as a Markov Decision Process (MDP). Thus, Black et al. [8] trains a diffusion model with the help of RL by maximizing a user-specific reward function, which connects the generative models and optimization methods.

### A.4 Diffusion Models in Adversarial Examples

Diffusion models have recently gained significant attention in generating adversarial examples due to their superb performance. They can generate high-quality adversarial examples that deceive target classifiers while remaining imperceptible to human observers.

Since the images generated by diffusion models inherently lack adversarial effects, a widely used approach is to use diffusion models along with existing methods of generating adversarial examples. The idea is to combine the generated samples from the diffusion model with perturbed samples from other attack methods such as PGD attacks during the attack process to generate high quality and imperceptible adversarial examples [55, 11].

Another promising direction is to use a (surrogate) classifier to guide the diffusion model generating samples that meet attacker specified goals by using gradient information from the classifier during the testing stage [38, 13, 18]. Also, Chen et al. [9] used the classifier guidance during the training stage of the diffusion model along with self and cross attention mechanisms.

Further, Beerens et al. [6] showed that poisoning the training set can produce a deceptive diffusion model that will generate adversarial samples without any guidance.

However, these works only care about static stealthiness in a supervised learning setting, while SHIFT also takes dynamic stealthiness into consideration.

### A.5 Relation with Constrained Diffusion Model

While diffusion models have been utilized to generate adversarial examples in the supervised learning setting (see Appendix A.4 for a review), their application in adversarial state perturbations in RL has not been considered before. We remark that our problem can be viewed as sampling from a diffusion model with constraints on realism and history alignment. However, existing approaches for

constrained diffusion [12] cannot be applied directly to our setting, as they require constraints such as physical rules to be explicitly given and easily evaluable. In our setting, it is difficult to identify the projection onto the valid states  $S^*$ , making these approaches less suitable.

## A.6 GAN-based Semantic Adversarial Attacks.

Generative adversarial networks (GANs) have been widely used to generate adversarial examples that go beyond traditional  $p$ -norm constraints in image classification. Notable works include Adv-GAN [52], which learns to generate perturbations that mislead classifiers while maintaining perceptual similarity, and Natural Adversarial Examples [63], which leverage latent space manipulations within VAE-GAN frameworks to produce realistic semantic shifts. Other approaches explore latent adversarial attacks [64] or direct image synthesis [4] to create imperceptible or highly transferable adversarial examples. While these methods succeed in the supervised setting, they do not transfer directly to reinforcement learning due to the absence of sequential structure, policy conditioning, or temporal consistency in GANs. Moreover, aligning GANs or other generative methods with reinforcement learning (RL)-specific objectives—such as minimizing the expected cumulative reward under a given policy—would require substantial modifications to both architecture and loss functions. As such, adapting these methods to generate semantics-aware perturbations in RL constitutes a distinct line of research and is not directly comparable to our approach.

## B Proof of Theorem 3.6

The following proof is adapted from the proof in Appendix H of Dhariwal and Nichol [14]. We show that in the RL state perturbation attacks setting, we could combine classifier-free and gradient guidance. Let  $\pi$  denote the victim’s policy,  $Q^\pi$  the state-action value function, and  $\tau_{t-1} = \{s_{t-1}, a_{t-1}, \dots, s_{t-k}, a_{t-k}\}$  the sequence of the last  $k$  observations and actions up to time  $t$ . We first define a conditional Markovian process  $\hat{q}$  similar to  $q$  as follows.

$$\begin{aligned} \hat{q}(\tilde{s}_t^0 | \tau_{t-1}) &:= q(\tilde{s}_t^0 | \tau_{t-1}) \\ \hat{q}(Q^\pi | \tilde{s}_t^0, \tau_{t-1}) &\text{ is known for every } (\tilde{s}_t^0, \tau_{t-1}) \\ \hat{q}(\tilde{s}_t^{i+1} | \tilde{s}_t^i, Q^\pi, \tau_{t-1}) &:= q(\tilde{s}_t^{i+1} | \tilde{s}_t^i), \quad \forall i \\ \hat{q}(\tilde{s}_t^{1:T} | \tilde{s}_t^0, Q^\pi, \tau_{t-1}) &:= \prod_{i=1}^T \hat{q}(\tilde{s}_t^i | \tilde{s}_t^{i-1}, Q^\pi, \tau_{t-1}), \end{aligned} \tag{4}$$

where  $q(\tilde{s}_t^0 | \tau_{t-1}) = P(\tilde{s}_t^0 | s_{t-1}, a_{t-1})$  is the conditional distribution of the original state  $\tilde{s}_t^0$  given the history  $\tau_{t-1}$ . Next we show that the joint distribution  $\hat{q}(\tilde{s}_t^{0:T}, Q^\pi | \tau_{t-1})$  given  $\tau_{t-1}$  is well defined.

$$\begin{aligned} \hat{q}(\tilde{s}_t^{0:T}, Q^\pi | \tau_{t-1}) &= \hat{q}(\tilde{s}_t^{1:T} | \tilde{s}_t^0, Q^\pi, \tau_{t-1}) \hat{q}(\tilde{s}_t^0, Q^\pi | \tau_{t-1}) \\ &= \prod_{i=1}^T \hat{q}(\tilde{s}_t^i | \tilde{s}_t^{i-1}, Q^\pi, \tau_{t-1}) \hat{q}(Q^\pi | \tilde{s}_t^0, \tau_{t-1}) \hat{q}(\tilde{s}_t^0 | \tau_{t-1}) \\ &= \prod_{i=1}^T \hat{q}(\tilde{s}_t^i | \tilde{s}_t^{i-1}, Q^\pi, \tau_{t-1}) \hat{q}(Q^\pi | \tilde{s}_t^0, \tau_{t-1}) \hat{q}(\tilde{s}_t^0 | \tau_{t-1}) \\ &= \prod_{i=1}^T \hat{q}(\tilde{s}_t^i | \tilde{s}_t^{i-1}, Q^\pi, \tau_{t-1}) \hat{q}(Q^\pi | \tilde{s}_t^0, \tau_{t-1}) \hat{q}(\tilde{s}_t^0 | \tau_{t-1}). \end{aligned}$$

Following essentially the same reasoning as in Appendix H of Dhariwal and Nichol [14] with the trivial extension of including the condition  $\tau_{t-1}$ , we have

$$\begin{aligned} \hat{q}(\tilde{s}_t^i | \tilde{s}_t^{i-1}, \tau_{t-1}) &= \hat{q}(\tilde{s}_t^i | \tilde{s}_t^{i-1}) \\ \hat{q}(\tilde{s}_t^{i-1} | \tilde{s}_t^i, \tau_{t-1}) &= q(\tilde{s}_t^{i-1} | \tilde{s}_t^i, \tau_{t-1}) \end{aligned}$$

899 Next, we show  $\hat{q}(Q^\pi|\tilde{s}_t^i, \tilde{s}_t^{i-1}, \tau_{t-1})$  does not depend on  $\tilde{s}_t^i$ .

$$\begin{aligned}
\hat{q}(Q^\pi|\tilde{s}_t^i, \tilde{s}_t^{i-1}, \tau_{t-1}) &= \frac{\hat{q}(\tilde{s}_t^{i-1}, \tilde{s}_t^i, Q^\pi, \tau_{t-1})}{\hat{q}(\tilde{s}_t^i, \tilde{s}_t^{i-1}, \tau_{t-1})} \\
&= \hat{q}(\tilde{s}_t^i|\tilde{s}_t^{i-1}, Q^\pi, \tau_{t-1}) \frac{\hat{q}(\tilde{s}_t^{i-1}, Q^\pi, \tau_{t-1})}{\hat{q}(\tilde{s}_t^{i-1}, \tilde{s}_t^i, \tau_{t-1})} \\
&= \hat{q}(\tilde{s}_t^i|\tilde{s}_t^{i-1}) \frac{\hat{q}(Q^\pi|\tilde{s}_t^{i-1}, \tau_{t-1})}{\hat{q}(\tilde{s}_t^i|\tilde{s}_t^{i-1}, \tau_{t-1})} \\
&= \hat{q}(\tilde{s}_t^i|\tilde{s}_t^{i-1}) \frac{\hat{q}(Q^\pi|\tilde{s}_t^{i-1}, \tau_{t-1})}{\hat{q}(\tilde{s}_t^i|\tilde{s}_t^{i-1})} \\
&= \hat{q}(Q^\pi|\tilde{s}_t^{i-1}, \tau_{t-1}). \tag{5}
\end{aligned}$$

900 We can now derive the reverse process that combines both classifier-free and gradient-guided methods.

$$\begin{aligned}
\hat{q}(\tilde{s}_t^{i-1}|\tilde{s}_t^i, Q^\pi, \tau_{t-1}) &= \frac{\hat{q}(\tilde{s}_t^{i-1}, \tilde{s}_t^i, Q^\pi, \tau_{t-1})}{\hat{q}(\tilde{s}_t^i, Q^\pi, \tau_{t-1})} \\
&= \frac{\hat{q}(\tilde{s}_t^{i-1}, \tilde{s}_t^i, \tau_{t-1}) \hat{q}(Q^\pi|\tilde{s}_t^{i-1}, \tilde{s}_t^i, \tau_{t-1})}{\hat{q}(\tilde{s}_t^i, Q^\pi, \tau_{t-1})} \\
&= \frac{\hat{q}(\tilde{s}_t^{i-1}|\tilde{s}_t^i, \tau_{t-1}) \hat{q}(\tilde{s}_t^i, \tau_{t-1}) \hat{q}(Q^\pi|\tilde{s}_t^{i-1}, \tilde{s}_t^i, \tau_{t-1})}{\hat{q}(\tilde{s}_t^i, Q^\pi, \tau_{t-1})} \\
&= \frac{\hat{q}(\tilde{s}_t^{i-1}|\tilde{s}_t^i, \tau_{t-1}) \hat{q}(Q^\pi|\tilde{s}_t^{i-1}, \tilde{s}_t^i, \tau_{t-1})}{\hat{q}(Q^\pi|\tilde{s}_t^i, \tau_{t-1})} \\
&\stackrel{(a)}{=} q(\tilde{s}_t^{i-1}|\tilde{s}_t^i, \tau_{t-1}) \frac{\hat{q}(Q^\pi|\tilde{s}_t^{i-1}, \tau_{t-1})}{\hat{q}(Q^\pi|\tilde{s}_t^i, \tau_{t-1})},
\end{aligned}$$

901 where (a) follows from Equation (5). Note that  $q(\tilde{s}_t^{i-1}|\tilde{s}_t^i, \tau_{t-1})$  can be learned through a history  
902 conditioned diffusion model  $p_\theta$  and we will use  $Q^\pi(s_t, \pi(\tilde{s}_t^i))$  to approximate  $\hat{q}(Q^\pi|\tilde{s}_t^i, \tau_{t-1})$ ,  $\forall i$ .  
903 We notice that the  $Q^\pi(s_t, \pi(\tilde{s}_t^i))$  term guides our reverse process to generate samples that lead to a  
904 lower state value. Thus, the attacker should use the victim's policy  $\pi$  here to gain better guidance  
905 through  $Q^\pi(\tilde{s}_t^i, \pi(\tilde{s}_t^i))$ . Plugging them back into the above equation, we have

$$\begin{aligned}
\hat{q}(\tilde{s}_t^{i-1}|\tilde{s}_t^i, Q^\pi, \tau_{t-1}) &\approx p_\theta(\tilde{s}_t^{i-1}|\tilde{s}_t^i, \tau_{t-1}) \frac{Q^\pi(s_t, \pi(\tilde{s}_t^{i-1}))}{Q^\pi(s_t, \pi(\tilde{s}_t^i))} \\
&\approx p_\theta(\tilde{s}_t^{i-1}|\tilde{s}_t^i, \tau_{t-1}) e^{\log Q^\pi(s_t, \pi(\tilde{s}_t^{i-1})) - \log Q^\pi(s_t, \pi(\tilde{s}_t^i))}. \tag{6}
\end{aligned}$$

906 Using the Taylor expansion, we get

$$\log Q^\pi(s_t, \pi(\tilde{s}_t^{i-1})) - \log Q^\pi(s_t, \pi(\tilde{s}_t^i)) \approx (\tilde{s}_t^{i-1} - \tilde{s}_t^i) \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)).$$

907 We also have

$$p_\theta(\tilde{s}_t^{i-1}|\tilde{s}_t^i, \tau_{t-1}) \propto \mathcal{N}(\tilde{s}_t^{i-1}; \mu_i, i), \sigma_i^2 \mathbf{I}) \propto \exp\left(-\frac{(\tilde{s}_t^{i-1} - \mu_i)^2}{2\sigma_i^2}\right).$$

908 where  $\mu_i$  comes from  $\epsilon_i = \Gamma \epsilon_\theta(s_t^i, i, \tau_{t-1}) + (1 - \Gamma) \epsilon_\theta(s_t^i, i)$ , as given by (2), and  $\sigma_i^2$  is determined  
909 by the noise scheduler  $\beta_i$ .

910 Substituting them back into (6), we have

$$\begin{aligned}
& p_\theta(\tilde{s}_t^{i-1}|\tilde{s}_t^i, \tau_{t-1}) e^{\log Q^\pi(s_t, \pi(\tilde{s}_t^{i-1})) - \log Q^\pi(s_t, \pi(\tilde{s}_t^i))} \\
& \propto \exp \left( -\frac{(\tilde{s}_t^{i-1} - \mu_i)^2}{2\sigma_i^2} + (\tilde{s}_t^{i-1} - \tilde{s}_t^i) \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)) \right) \\
& = \exp \left( -\frac{(\tilde{s}_t^{i-1} - \mu_i)^2 - 2\sigma_i^2 (\tilde{s}_t^{i-1} - \tilde{s}_t^i) \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i))}{2\sigma_i^2} \right) \\
& = \exp \left( -\frac{(\tilde{s}_t^{i-1} - \mu_i)^2 - 2\sigma_i^2 (\tilde{s}_t^{i-1} - \mu_i) \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)) + (\sigma_i^2 \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)))^2}{2\sigma_i^2} \right) \\
& \times \exp \left( \frac{2\sigma_i^2 (\mu_i - \tilde{s}_t^i) \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)) + (\sigma_i^2 \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)))^2}{2\sigma_i^2} \right) \\
& = \exp \left( -\frac{((\tilde{s}_t^{i-1} - \mu_i) - \sigma_i^2 \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)))^2}{2\sigma_i^2} \right. \\
& \quad \left. + \frac{2\sigma_i^2 (\mu_i - \tilde{s}_t^i) \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)) + (\sigma_i^2 \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)))^2}{2\sigma_i^2} \right) \\
& \propto \exp \left( -\frac{(\tilde{s}_t^{i-1} - (\mu_i + \sigma_i^2 \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i))))^2}{2\sigma_i^2} \right). \tag{7}
\end{aligned}$$

911 Equation (7) implies that the reverse process when sampling from a history-conditioned DDPM  
912 model guided by the victim's state value function can be represented as

$$p(\tilde{s}_t^{i-1}|\tilde{s}_t^i, Q^\pi, \tau_{t-1}) = \mathcal{N}(\tilde{s}_t^{i-1}; \mu_i + \sigma_i^2 \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)), \sigma_i^2 \mathbf{I}).$$

913 Noticed that we want to guide the generated state  $\tilde{s}_t^0$  lead to a lower state-action value, thus we change  
914 sign of the gradient guidance to minus.

$$p(\tilde{s}_t^{i-1}|\tilde{s}_t^i, Q^\pi, \tau_{t-1}) = \mathcal{N}(\tilde{s}_t^{i-1}; \mu_i - \sigma_i^2 \nabla_{\tilde{s}_t^i} \log Q^\pi(s_t, \pi(\tilde{s}_t^i)), \sigma_i^2 \mathbf{I}).$$

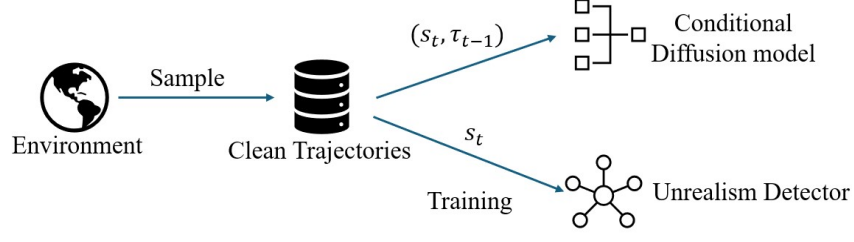
## 915 C Implementation Details and Algorithms

### 916 C.1 Two Stage Attacks Pipelines

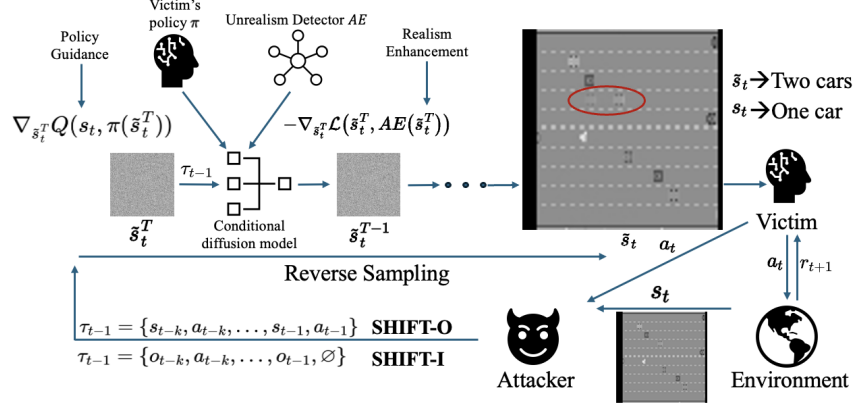
917 Figure 3 gives an overview of our two-stage diffusion-based attack including all the major components  
918 involved.

### 919 C.2 Visualization of SHIFT-O and SHIFT-I in the Freeway Environment and Doom

920 Figure 5 gives an illustration of SHIFT-O and SHIFT-I in the Atari Freeway environment. Figure 4  
921 provides additional comparison between SHIFT-O/I and PGD and rotation attack from Korkmaz  
922 [31] in Atari Freeway and Doom.



(a) Training Stage



(b) Testing Stage

Figure 3: Pipelines of SHIFT’s two stages. a) shows the training stage where the attacker uses clean data to train a history-conditioned diffusion model and an autoencoder-based anomaly detector. b) shows the testing stage where the attacker perturbs the true state through the reverse sampling process of the pre-trained conditional diffusion model guided by the gradient of the victim’s policy and that of the autoencoder’s reconstruction loss.

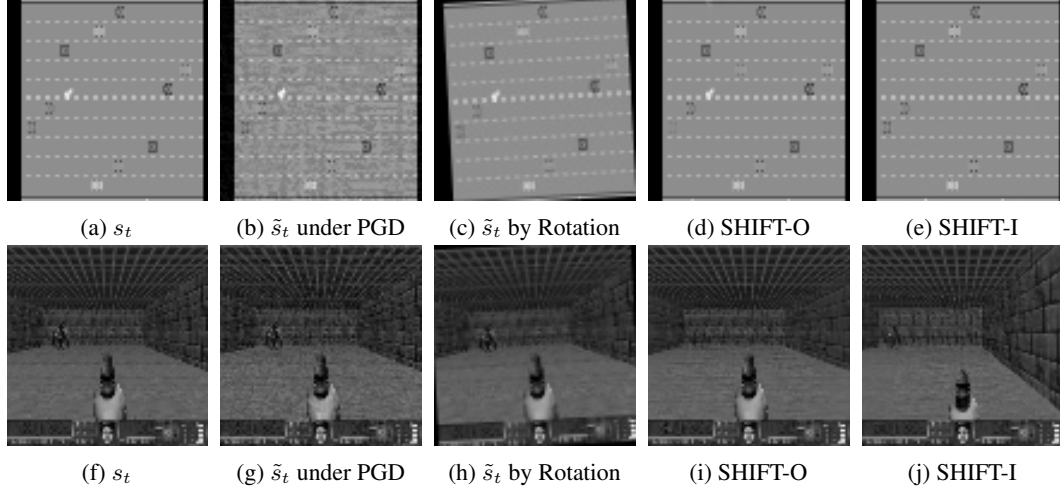


Figure 4: Extra examples of true and perturbed states of Atari Freeway and Doom. a) is the true state. b) and c) are the perturbed states under the PGD attack with a  $l_\infty$  budget of  $\frac{15}{255}$  and through rotation [31] by 3 degrees counterclockwise, respectively. d) and e) are the perturbed states generated by our SHIFT-O and SHIFT-I attacks, respectively. Neither PGD nor Rotation attacks can alter the decision-related semantics.

### 923 C.3 Score-Based Diffusion Model

924 As shown in Song et al. [45], a diffusion process  $\{\mathbf{x}(i)\}_{i \in [0, T]}$  can be represented as the solution to  
 925 a standard stochastic differential equation (SDE):

$$d\mathbf{x} = \mathbf{f}(\mathbf{x}, i)di + \mathbf{g}(i)d\mathbf{w},$$

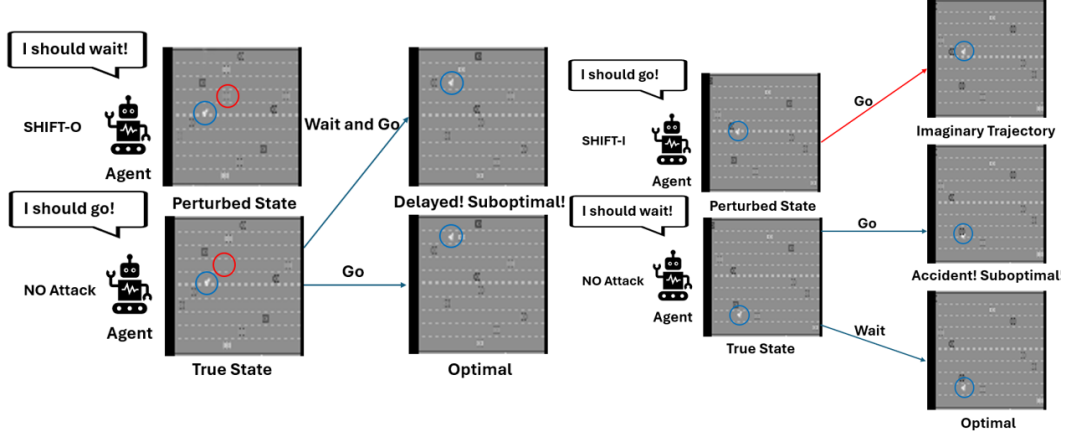


Figure 5: Visualization of **SHIFT-O** and **SHIFT-I** in Freeway. The goal of the Freeway game is to control the agent (blue circle) to cross the road quickly and safely. **SHIFT-O** injects an extra car (red circle) into the agent’s observations to slow down the agent. **SHIFT-I** injects a segment of imaginary trajectory into the agent’s observations that misleads the agent to move forward, despite a car being present in front of the agent in the real state, resulting in a collision.

where  $\mathbf{f}$  represents the drift coefficient, which models the deterministic part of the SDE and determines the rate at which the process changes over time on average.  $g(i)$  is called the diffusion coefficient, which represents the random part of the SDE and determines the magnitude of the noise. Finally,  $\mathbf{w}$  represents a Brownian motion so that  $g(i)d\mathbf{w}$  is the noising process.

We can let the diffusion process have  $\mathbf{x}_0 \sim p_0$  and  $\mathbf{x}_T \sim p_T$ , where  $p_0 = p_{\text{data}}$  is the data distribution and  $p_T$  is a Gaussian noise distribution independent of  $p_0$ . Then we could run the reverse-time SDE to recover a sample from  $p_0$  by the following process:

$$d\mathbf{x} = [\mathbf{f}(\mathbf{x}, i) - g(i)^2 \nabla_{\mathbf{x}} \log p_i(\mathbf{x})] di + g(i)d\bar{\mathbf{w}},$$

where  $\nabla_{\mathbf{x}} \log p_i(\mathbf{x})$  is the score function and  $\bar{\mathbf{w}}$  is a Brownian motion that flows back from time  $T$  to 0. The training objective for the score matching function  $\mathbf{s}_\theta$  for the SDE is then given by:

$$\arg \min_{\theta} \mathbb{E}_i \left[ \lambda(i) \mathbb{E}_{\mathbf{x}(0)} \mathbb{E}_{\mathbf{x}(i)|\mathbf{x}(0)} \left[ \left\| \mathbf{s}_\theta(\mathbf{x}(i), i) - \nabla_{\mathbf{x}(i)} \log p_{0i}(\mathbf{x}(i) | \mathbf{x}(0)) \right\|_2^2 \right] \right],$$

where  $\lambda(i)$  is a positive weighting function and  $i$  is uniformly sampled from  $[0, T]$ . The objective can be further simplified since  $p_{0i}$  is a known Gaussian distribution.

#### C.4 EDM Model and ODE formulation

Inspired by Song et al. [45], EDM [29] proposes an ODE formulation of the diffusion model by having a scheduler  $\sigma(t)$  to schedule the noise added at each time step  $t$ . The score function correspondingly changes to  $\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma)$ , which does not depend on the normalization constant of the underlying density function  $p(\mathbf{x}, \sigma)$  and is much easier to evaluate. To be specific, if  $D(\mathbf{x}; \sigma)$  is a denoiser function that minimizes:

$$\mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} \mathbb{E}_{\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})} \|D(\mathbf{x} + \mathbf{n}; \sigma) - \mathbf{x}\|_2^2 \quad (8)$$

then

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma) = (D(\mathbf{x}; \sigma) - \mathbf{x})/\sigma^2$$

We usually train a neural network  $\theta$  to learn the denoising function  $D(\mathbf{x}, \sigma)$  by using the simplified training objective in Equation (8). Utilizing this finding, EDM only requires a small number of reverse sampling steps to generate a high quality sample. However, EDM needs more preconditioning parameters such as scaling  $\mathbf{x}$  to an approximate dynamic range, as further discussed below.

## 948 C.5 EDM as a conditional diffusion model

949 In this paper, we follow the approach in Alonso et al. [3] to train an EDM-based diffusion model  
 950 conditioned on a history  $\tau_{t-1}$  to predict the next state  $s_t$ . Taking the network preconditioning  
 951 parameters used in EDM into account, we have the denoising function changed to:

$$\mathbf{D}_\theta(s_t^i, c_{\text{noise}}^i, \tau_{t-1}) = c_{\text{skip}}^i s_t^i + c_{\text{out}}^i \mathbf{F}_\theta(c_{\text{in}}^i s_t^i, c_{\text{noise}}^i, \tau_{t-1}),$$

952 where  $\mathbf{F}_\theta$  is the neural network to be trained, the preconditioners  $c_{\text{in}}$  and  $c_{\text{out}}$  scale the network's input  
 953 and output magnitude to keep them at unit variance for any noise level  $\sigma(i)$ ,  $c_{\text{noise}}^i$  is an empirical  
 954 transformation of the noise level, and  $c_{\text{skip}}^i$  is determined by the noise level  $\sigma(i)$  and the standard  
 955 deviation of the data distribution  $\sigma_{\text{data}}$ . The detailed expressions are given below:

$$c_{\text{in}}^i = \frac{1}{\sqrt{\sigma(i)^2 + \sigma_{\text{data}}^2}} \quad (9)$$

$$c_{\text{out}}^i = \frac{\sigma(i)\sigma_{\text{data}}}{\sqrt{\sigma(i)^2 + \sigma_{\text{data}}^2}} \quad (10)$$

$$c_{\text{noise}}^i = \frac{1}{4} \log(\sigma(i)) \quad (11)$$

$$c_{\text{skip}}^i = \frac{\sigma_{\text{data}}^2}{\sigma_{\text{data}}^2 + \sigma^2(i)} \quad (12)$$

956 where  $\sigma_{\text{data}} = 0.5$ . The noise parameter  $\sigma(i)$  is sampled to maximize the effectiveness during training  
 957 by setting  $\log(\sigma(i)) = \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$ , where  $P_{\text{mean}} = -0.4$ ,  $P_{\text{std}} = 1.2$ . Refer to Karras et al. [29]  
 958 for a detailed explanation.

959 The training objective of  $\mathbf{F}_\theta$  changes correspondingly to

$$\mathcal{L}(\theta) = \mathbb{E}[\|\mathbf{F}_\theta(c_{\text{in}}^i s_t^i, c_{\text{noise}}^i, \tau_{t-1}) - \frac{1}{c_{\text{out}}^i} (s_t - c_{\text{skip}}^i s_t^i)\|^2] \quad (13)$$

960 In our implementation, we change the residual block layers from [2,2,2,2] to [2,2] and the denosing  
 961 steps to 5, and set the drop condition rate to 0.1. We keep other hyperparameters the same as Alonso  
 962 et al. [3].

## 963 C.6 Training and testing stage algorithms for SHIFT

---

### Algorithm 1: History-Aligned Conditional Diffusion Model Training

---

**Input:** Training data  $O = \{(s_t, \tau_{t-1})\}_{i=1}^N$ , condition dropping rate  $\alpha_{\text{drop}}$ ,  $P_{\text{mean}}$ ,  $P_{\text{std}}^2$ ,  $\sigma$ , learning  
 rate  $\eta$ , history length  $k$ .

**Output:** Trained EDM model parameters  $\theta$

**Initialize:** EDM model parameters  $\theta$

**for** number of training iterations **do**

    Sample a data point  $(s_t, \tau_{t-1}) \sim O$ ;  
 //  $\tau_{t-1} = \{s_{t-k}, a_{t-k}, \dots, s_{t-1}, a_{t-1}\}$  is the **true** history

964     Sample  $\log(\sigma) \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$ ;  
     Calculate preconditioners  $c_{\text{in}}, c_{\text{out}}$  based on  $\sigma$  according to (9) and (10);  
     Generate noisy data  $s_t^i \sim \mathcal{N}(s_t, \sigma^2 \mathbf{I})$ ;  
     **if** random  $> \alpha_{\text{drop}}$  **then**  
         | Compute generated state  $\tilde{s}_t = \mathbf{D}_\theta(s_t^i, c_{\text{noise}}^i, \tau_{t-1})$   
     **else**  
         | Compute generated state  $\tilde{s}_t = \mathbf{D}_\theta(s_t^i, c_{\text{noise}}^i)$   
     Compute loss  $\mathcal{L}(\theta)$  based on Equation (13);  
     Update  $\theta$  using gradient descent:  $\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}(\theta)$ ;

**end**

---

---

**Algorithm 2:** Testing Stage Sampling with History, Policy and Realism Guidance

---

**Input:** History conditioned diffusion model  $\mathbf{D}_\theta$ , victim’s policy  $\pi$ , number of denoising steps  $T$ , autoencoder-based unrealism detector  $\mathbf{AE}$ , agent state-action value function  $Q^\pi$ , given **true** history  $\tau_{t-1} = \{s_{t-k}, a_{t-k}, \dots, s_{t-1}, a_{t-1}\}$  for SHIFT-O or **observed** history  $\tau_{t-1} = \{o_{t-k}, a_{t-k}, \dots, o_{t-1}, \emptyset\}$  for SHIFT-I, classifier-free guidance strength  $\Gamma_1$ , classifier guidance strength  $\Gamma_2$ , attack rate  $\xi$ , true state  $s_t$ , history length  $k$ .

**Output:** Generated sample  $\tilde{s}_t$

**Initialize:**  $\tilde{s}_t^T \sim \mathcal{N}(0, \mathbf{I})$ ;

Calculate state importance  $\omega(s) = \max_{a_1 \in A} Q^\pi(s, a_1) - \min_{a_2 \in A} Q^\pi(s, a_2)$ ;

**if**  $\omega(s_t)$  in top  $\xi$  percentile of previous states and total attacked times  $< t \times \xi$  **then**

    total attack times += 1;

    // Inject SHIFT Attack

**for**  $i = T$  **to** 1 **do**

        // Calculate proposed output  $\hat{s}_t$  based on  $\tilde{s}_t^i$

$\hat{s}_t = \tilde{s}_t^i$ ;

**for**  $j = i$  **to** 1 **do**

$\hat{s}_t = \mathbf{D}_\theta(\hat{s}_t, c_{\text{noise}}^j, \tau_{t-1})$

**end**

        Calculate policy guidance gradient  $g \leftarrow \nabla_{\hat{s}_t} Q^\pi(s_t, \pi(\hat{s}_t))$ ;

        Inject policy guidance  $\tilde{s}_t^i = \tilde{s}_t^i - \Gamma_2 g$ ;

        Generate next sample  $\tilde{s}_t^{i-1} = \Gamma_1(i) \mathbf{D}_\theta(\tilde{s}_t^i, c_{\text{noise}}^i, \tau_{t-1}) + (1 - \Gamma_1(i)) \mathbf{D}_\theta(\tilde{s}_t^i, c_{\text{noise}}^i)$ ;

**if**  $i \neq 1$  **then**

            Conduct a gradient descent based on the reconstruction error from the unrealism detector

$$\tilde{s}_t^{i-1} = \tilde{s}_t^{i-1} - \nabla_{\tilde{s}_t^{i-1}} \mathcal{L}(\tilde{s}_t^{i-1}, \mathbf{AE}(\tilde{s}_t^{i-1}));$$

**end**

**end**

**end**

---

## C.7 Hyper-parameters Setting

**EDM Diffusion Model Training Parameters.** As mentioned before, we only change the residual block layers from [2,2,2,2] to [2,2] and the denosing steps to 5, and set the drop condition rate to 0.1. We keep other hyperparameters the same as Alonso et al. [3] for training the EDM diffusion model.

**Testing Stage Parameters and Testbench Specification.** We schedule the classifier-free guidance scale as  $\Gamma_1(i) = \max(\frac{T-i}{T}, 0.3)$ , where  $T$  is the number of reverse steps and  $i$  is the current reverse step. We set the policy guidance strength  $\Gamma_2$  differently in each environment under each defense. In the Pong environment, we set  $\Gamma_2 = 3.5$  for DQN, DP-DQN, and DMBP and  $\Gamma_2 = 2$  for all other defenses. In the Freeway environment, we set  $\Gamma_2 = 6$  for DQN, DP-DQN and DMBP and  $\Gamma_2 = 4.5$  for all other defenses. In the BankHeist environment, we set  $\Gamma_2 = 4$  for all defenses. In the RoadRunner environment, we set  $\Gamma_2 = 6$  for all defenses. In the Doom environment, we set  $\Gamma_2 = 4.5$  for DQN, DP-DQN and DMBP and  $\Gamma_2 = 2.5$  for all other defenses. For the Airsim autonomous driving simulator, we use CITY, which is a complicated environment that simulate real city traffic situation. We set  $\Gamma_2 = 0.5$  for all defenses in Airsim.

We conduct all of our experiments on a workstation equipped with an Intel I9-12900KF CPU, an RTX 3090 GPU, and 64GB system RAM.

**Pre-processing Atari, Doom and Airsim Environments.** We have used the same environment wrappers as in Zhang et al. [59], which convert an RGB image to a gray-scale image and resize the image to reduce its resolution from  $210 \times 160$  to  $84 \times 84$  for Atari environments and  $320 \times 240$  to  $84 \times 84$  for Doom. We also follow Zhang et al. [59] to center crop images using the same shifting parameters as in Zhang et al. [59], where we set the cropping shift to 10 for Pong, 20 for Roadrunner, and 0 for Freeway, Bankhesit and Doom. We also convert the front view camera snapshot of the autonomous agent in Airsim to  $84 \times 84$  grayscale image. We do not stack frames in our pre-processing.

## D More Evaluation Results

Table 3: Attack results on Atari BankHeist and RoadRunner Environments under SHIFT-I and SHIFT-O attacks with 1.0 and 0.25 attack frequency.

Model	BankHeist				RoadRunner			
	SHIFT-O-1.0	SHIFT-O-0.25	SHIFT-I-1.0	SHIFT-I-0.25	SHIFT-O-1.0	SHIFT-O-0.25	SHIFT-I-1.0	SHIFT-I-0.25
<b>DQN-No Attack</b>	680.0±0.0	680.0±0.0	680.0±0.0	680.0±0.0	13500.0±0	13500.0±0	13500.0±0	13500.0±0
<b>DQN</b>	0.0±0.0	50.0±45.6	0.0±0.0	0.0±0.0	0.0±0.0	160.0±151.7	0.0±0.0	0.0±0.0
<b>SA-DQN</b>	3.0±3.5	80.0±14.1	0.0±0.0	0.0±0.0	480.0±521.5	2820.0±1690.3	0.0±0.0	440.0±559.5
<b>WocAR-DQN</b>	6.0±5.5	66.0±27.0	0.0±0.0	0.0±0.0	120.0±83.7	1080.0±807.5	0.0±0.0	100.0±70.7
<b>CAR-DQN</b>	12.0±14.1	24.0±15.2	0.0±0.0	6.0±8.9	1100.0±447.2	920.0±44.7	260.0±313.0	540.0±320.9
<b>DP-DQN</b>	15.0±12.0	32.0±29.2	0.0±0.0	0.0±0.0	220.0±130.4	3900.0±2640.1	0.0±0.0	0.0±0.0
<b>DMBP</b>	20.0±18.2	54.0±53.7	0.0±0.0	0.0±0.0	240.0±134.2	4300.0±1433.5	0.0±0.0	80.0±83.7

Table 4: Attack results of our methods with {0.15,0.25,0.5,1.0} attack frequencies against DMBP defenses.

Environment		SHIFT-O					SHIFT-I			
Frequency	No Attack	0.15	0.25	0.50	1.00	0.15	0.25	0.50	1.00	
Pong	21.0±0.0	-0.6±3.0	-9.8±4.5	-12.6±3.8	-14.0±4.9	-20.4±1.3	-20.2±0.8	-20.6±0.9	-20.6±0.9	
Freeway	34.1±0.1	31.6±0.5	31.0±1.2	28.8±1.5	22.0±0.7	13.8±2.2	14.6±2.5	14.6±1.1	19.2±2.9	
Doom	75.4±4.4	71.4±5.1	65.4±9.8	65.2±7.1	-184.8±231.7	-101.6±200.3	-256.2±180.4	-239.8±174.4	-302.0±2.7	
AirSim	40.3±0.5	28.1±7.2	22.8±8.0	19.2±6.6	20.8±12.0	11.7±3.5	9.4±4.9	6.6±2.5	6.2±0.7	

Table 5: Attack performance on the Highway environment (with a continuous action space) and the Pong environments (with a discrete action space), under PPO policies with/without DMBP defense. For the continuous action environment Highway, we compare with the MAD attack from Zhang et al. [59]. For the discrete action environment Pong, we compare with the PGD attack.

Attack Type	Highway		Pong	
	PPO	DMBP	PPO	DMBP
<b>No Attack</b>	23.3 ± 9.6	22.5 ± 8.9	21.0 ± 0.0	21.0 ± 0.0
<b>MAD/PGD Attack</b>	2.7 ± 1.3	16.8 ± 3.4	-21.0 ± 0.0	20.8 ± 0.4
<b>SHIFT-O-1.0</b>	2.0 ± 1.8	4.4 ± 0.6	-21.0 ± 0.0	-12.3 ± 7.6
<b>SHIFT-O-0.25</b>	13.7 ± 6.9	15.4 ± 5.2	-20.6 ± 0.9	1.6 ± 4.2
<b>SHIFT-I-1.0</b>	0.84 ± 0.3	2.54 ± 0.2	-21.0 ± 0.0	-20.0 ± 1.2
<b>SHIFT-I-0.25</b>	2.55 ± 0.8	3.80 ± 1.2	-21.0 ± 0.0	-19.6 ± 0.8

Table 6: Automated detection results with MAD threshold set to 5, and CUSUM configured with a drift of 1.5 and a threshold of 3.

Env	Freeway	
Attack Method	MAD Dector	CUSUM Dector
<b>PGD-1/255</b>	Undetected	Undetected
<b>PGD-15/255</b>	Detected	Detected
<b>MinBest-1/255</b>	Detected	Detected
<b>MinBest-15/255</b>	Detected	Detected
<b>PA-AD-1/255</b>	Undetected	Undetected
<b>PA-AD-15/255</b>	Detected	Detected
<b>PA-AD-TC-15/255</b>	Detected	Detected
<b>Rotation Degree 1</b>	Undetected	Undetected
<b>Transform(1,0)</b>	Undetected	Undetected
<b>SHIFT-O-1.0</b>	<b>Undetected</b>	<b>Undetected</b>
<b>SHIFT-I-1.0</b>	<b>Undetected</b>	<b>Undetected</b>

### D.1 Additional Attack Results in Atari Environments

We report additional attack performance results on the remaining two Atari environments: BankHeist and RoadRunner in Table 3.

### D.2 Attack Performance on Continuous Action Space Environment and PPO Policy

Table 5 shows our SHIFT attack can successfully work on PPO policies under both continuous action space environment Highway [34] environment and discrete action space environment Atari Pong.

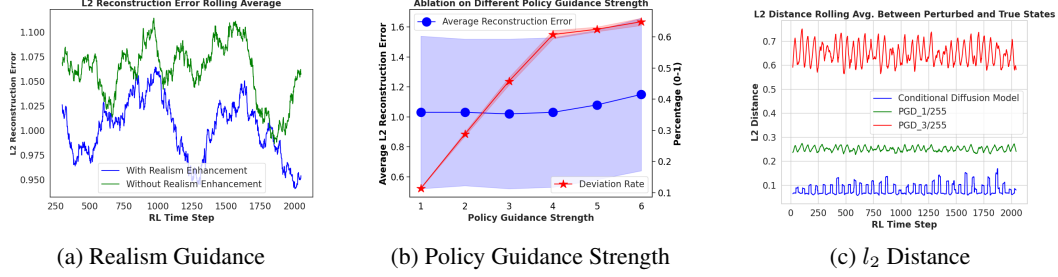


Figure 6: Ablation Study Results. a) shows the rolling average of  $l_2$  reconstruction error (from the autoencoder-based realism detector) of our generated perturbed states with and without the realism enhancement. b) shows the  $l_2$  reconstruction error, deviation rate under different policy guidance strengths with **SHIFT-O** attack. c) shows distance between perturbed and true states ( $84 \times 84$  grayscale images). (a) and (b) use the vanilla DQN policy.

We compared our attack with the Maximal Action Difference (MAD) attack used in [59], which is a simple yet effective attack against the PPO algorithm, both with and without the DMBP defense. Since MAD attack does not apply on discrete action space, we report PGD attack with a budget  $1/255$  as a baseline. The results demonstrate that our attack outperforms MAD/PGD in both cases, further supporting its generalizability to different RL algorithms.

### D.3 Detection Results

Furthermore, inspired by the adversarial attack detection method in Russo and Proutiere [42], we have evaluated our attack under two commonly used anomaly detection methods based on the Wasserstein-1 distance between adjacent perturbed states: (1) the MAD (Median Absolute Deviation) anomaly detector [1], which flags anomaly if three consecutive steps of perturbed trajectories violate clean trajectories' median + threshold  $\times$  MAD, where clean trajectories' median and MAD are both evaluated under Wasserstein-1 distance between adjacent true states. We set threshold = 5 in the table below; and (2) CUSUM [41], a sequential change detection method that identifies persistent deviations (again in terms of Wasserstein-1 distance) from clean trajectories by accumulating small shifts over time. We set the drift to 1.5 and threshold to 3 in CUSUM. We have compared PGD, Minbest, PA-AD attacks (with budgets  $1/255$  and  $15/255$ ), Rotation and Transformation attacks from Korkmaz [31] and our method.

### D.4 Ablation on Realism Guidance

Figure 6a illustrates the  $l_2$  reconstruction error, defined as  $\|s_t - \mathbf{AE}(s_t)\|_2$ , for generated perturbed states both with and without the realism enhancement component. The figure demonstrates that, by incorporating realism enhancement, the  $l_2$  reconstruction error is significantly reduced. This reduction indicates that realism enhancement effectively contributes to the generation of perturbed states that are more stealthy and less likely to be detected.

### D.5 Ablation on Policy Guidance Strength

Figure 6b illustrates the performance of our attack across various levels of policy guidance strength  $\Gamma_2$ . The figure indicates that as the strength increases, the effectiveness of our attack improves, leading to higher manipulation and deviation rates. However, this increased strength also results in a higher  $l_2$  reconstruction error, which negatively impacts the realism of the generated perturbed states. Consequently, there exists a trade-off between attack effectiveness and stealthiness when selecting different policy guidance strengths.

### D.6 Ablation on Attack Frequency

Table 4 illustrates the performance of our attack across different attack frequencies  $\xi$ . The results show that even at low attack frequencies, our method significantly reduces the agent's cumulative reward. Moreover, the reduction becomes more pronounced as the attack frequency increases.

Table 7: Performance of high-sensitivity direction attacks in [31].

Attack	Defense	Pong	Freeway
		Reward↓	Reward↓
B&C	DQN	$-21 \pm 0.00$	$23 \pm 0.00$
	SA-DQN	$11 \pm 0.00$	$25 \pm 0.00$
	DMBP	$20 \pm 1.41$	$27.2 \pm 0.68$
Blurred Observations	DQN	$-21 \pm 0.00$	$18 \pm 0.00$
	SA-DQN	$-20 \pm 0.00$	$27 \pm 0.00$
	DMBP	$20 \pm 0.58$	$33.2 \pm 0.37$
Rotation Degree 1	DQN	$-20 \pm 0.00$	$26.6 \pm 0.45$
	SA-DQN	$-18 \pm 0.00$	$21 \pm 0.00$
	DMBP	$14.6 \pm 2.68$	$27.6 \pm 0.45$
Transform (1,0)	DQN	$-21 \pm 0.00$	$26 \pm 0.00$
	SA-DQN	$-21 \pm 0.00$	$24 \pm 0.00$
	DMBP	$17.8 \pm 2.85$	$27.2 \pm 0.37$

Table 8: Large-scale rotation and shift attacks against fine-tuned diffusion-based defense.

Pong	Defense	Reward↓
Rotation Degree 3	DMBP	$20 \pm 0.71$
Transform (2,1)	DMBP	$18.8 \pm 1.79$

Table 9: Average Reconstruction error, Wasserstein distance, SSIM and LPIPS of the high-sensitivity direction attacks in Korkmaz [31] across a randomly sampled episode. The Wasserstein distance is the Wasserstein-1 distance calculated between the current perturbed state and the previous step’s true state.

Freeway	Reconstruction Error↓	Wass. ( $\times 10^{-3}$ )↓	SSIM↑	LPIPS↓
B&C	$15.30 \pm 0.02$	$0.80 \pm 0.2$	$0.9429 \pm 0.0006$	$0.0455 \pm 0.0031$
Blurred Observations	$5.81 \pm 0.04$	$0.82 \pm 0.2$	$0.5974 \pm 0.0034$	$0.3535 \pm 0.0202$
Rotation Degree 1	$6.41 \pm 0.2$	$0.80 \pm 0.2$	$0.8237 \pm 0.0022$	$0.0351 \pm 0.0041$
Transform (1,0)	$9.32 \pm 0.1$	$0.83 \pm 0.2$	$0.6045 \pm 0.0074$	$0.0741 \pm 0.0068$
SHIFT-O-1.0	$1.02 \pm 0.5$	$0.89 \pm 0.3$	$0.9990 \pm 0.0014$	$0.0008 \pm 0.0014$
SHIFT-I-1.0	$1.05 \pm 0.5$	$0.84 \pm 0.2$	$0.9904 \pm 0.0043$	$0.0175 \pm 0.0124$

Table 10: Ablation studies on EDM and DDPM diffusion architectures.

Pong	DDPM		EDM	
	Reward ↓	Deviation Rate(%)↑	Reward↓	Deviation Rate(%)↑
DQN	$-20.6 \pm 0.5$	$83.6 \pm 1.0$	$-21.0 \pm 0.0$	$90.0 \pm 0.3$
DMBP	$-10.4 \pm 4.8$	$46.1 \pm 0.3$	$-14.0 \pm 6.2$	$47.3 \pm 0.9$
Sampling Time	$\sim 5$ sec		$\sim 0.2$ sec	

## 1030 D.7 Ablation on DDPM and EDM diffusion architectures

1031 We compare DDPM and EDM in terms of attack efficiency and computational cost in Table 10.  
 1032 The results show that EDM and DDPM exhibit similar attack performance. However, DDPM is  
 1033 significantly slower than EDM in terms of sampling time (the average time needed to generate a  
 1034 single perturbed state during testing), making DDPM incapable of generating real-time attacks during  
 1035 testing. This validates the selection of EDM as the diffusion model architecture for constructing our  
 1036 attacks.

## 1037 D.8 Performance and Stealthiness of High-Sensitivity Direction Attacks in Korkmaz [31]

1038 Korkmaz [31] proposes various high-sensitivity direction-based attacks that can generate perturbed  
 1039 states that are visually imperceptible and semantically different from the clean states, including

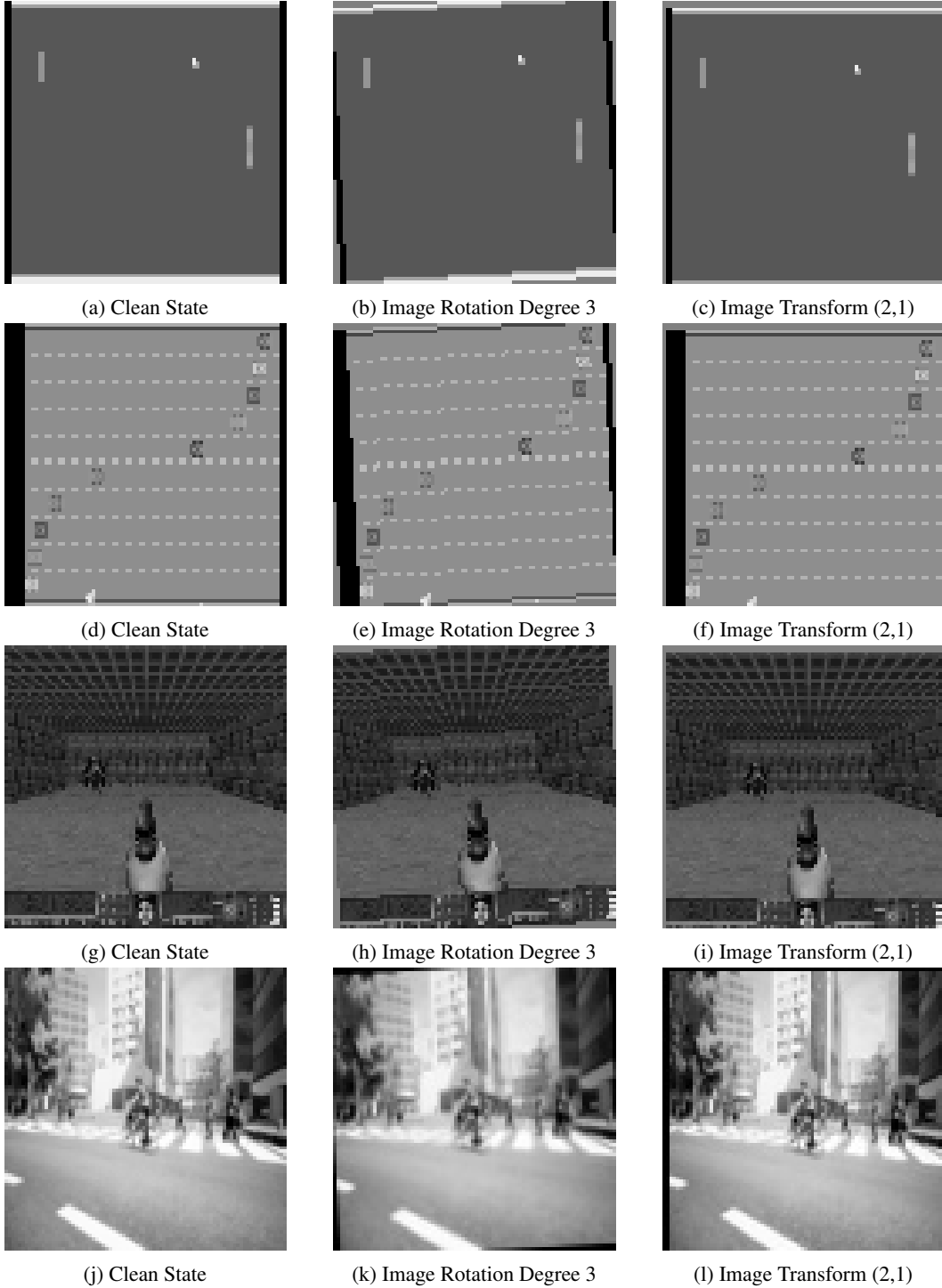


Figure 7: Perceptual impact of large-scale rotation and transform on Atari Pong, Atari Freeway, Doom, and Airsim.

1040 changing brightness and contrast (B&C), image blurring, image rotation and image transform. These  
 1041 attack methods reveal the brittleness of robust RL methods such as SA-DQN, but they mainly target  
 1042 changes in visually significant but not domain-specific semantics. For example, the relative distance  
 1043 between the pong ball and the pad will remain the same after brightness and contrast changes or  
 1044 image transform in the Pong environment. Consequently, the perturbed images generated by these

1045 methods can potentially be purified by a diffusion model. To confirm this, we have conducted new  
 1046 experiments, showing that (1) the DMBP defense with a diffusion model trained from clean data  
 1047 only is able to defend against B&C, blurring, and small scale rotation and transform attacks (see  
 1048 Table 7), and (2) when the diffusion model is fine-tuned by randomly applying image rotations or  
 1049 transform during training, the DMBP defense can mitigate large scale image rotations and transform  
 1050 considered in Korkmaz [31] (see Table 8). In contrast, our diffusion guided attack can change the  
 1051 decision-relevant semantics of the images, such as moving the Pong ball to a different position  
 1052 without changing other elements in the Pong environment as shown in Figure 2). This is the key  
 1053 reason why our attack can bypass strong diffusion-based defense methods.

1054 Furthermore, Korkmaz [31] claims their attacks are imperceptible by comparing the perturbed state  
 1055  $\tilde{s}_t$  and the true state  $s_t$ . However, we found that this only holds for small perturbations. For example,  
 1056 the Rotation attack with degree 3 and Transform attack (1,2) in the Pong environment considered  
 1057 in their paper can be easily detected by humans (see Figure 7). Further, their metric for stealthiness  
 1058 is static and does not consider the sequential decision-making nature of RL. In contrast, our attack  
 1059 method aims to stay close to the set of true states  $S^*$  to maintain static stealthiness and realistic  
 1060 (Definitions 3.2) and align with the history to achieve dynamic stealthiness (Definitions 3.4). These  
 1061 are novel definitions for characterizing stealthiness in the RL context. The static stealthiness and  
 1062 realism are demonstrated through the low reconstruction loss of our method shown in Table 2,9. In  
 1063 contrast, attacks from Korkmaz [31] generate out-of-distribution perturbed states by applying image  
 1064 transformations, which induce high reconstruction error, indicating less realistic than our methods. We  
 1065 further compare the Wasserstein distance between a perturbed state and the previous step’s perturbed  
 1066 state, SSIM and LPIPS metrics to measure the historical-alignment and trajectory-faithfulness in  
 1067 Table 9. The results show that the perturbed states generated by Korkmaz [31] achieve the same  
 1068 level of historical alignment but are less trajectory-faithful than our SHIFT attack.

## 1069 D.9 Time Complexity Comparison

1070 In terms of time complexity, high-sensitivity direction attacks can generate perturbations instantan-  
 1071 eously as they are policy independent, and PGD, PA-AD-TC, MinBest, and PA-AD all take around  
 1072 0.02 seconds to generate a perturbation with 10 iterations. Due to the computational overhead of  
 1073 the reverse process, diffusion-based methods typically require longer generation times. However,  
 1074 by adopting the EDM diffusion paradigm, we reduce the reverse process steps to 5, resulting in  
 1075 a generation time of approximately 0.2 seconds per perturbed state. Although slower than PGD,  
 1076 MinBest, and PA-AD, this still allows our attack to remain feasible for real-time applications.

## References

- [1] Crispin Agar. MAD: Anomaly detection. <https://crispinagar.github.io/blogs/mad-anomaly-detection.html>, 2024.
- [2] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.
- [3] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *arXiv preprint arXiv:2405.12399*, 2024.
- [4] Shumeet Baluja and Ian Fischer. Adversarial examples from adversarial generative networks. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence(IJCAI)*, 2018.
- [5] Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *International Conference on Learning Representations(ICLR)*, 2024.
- [6] Lucas Beerens, Catherine F. Higham, and Desmond J. Higham. Deceptive diffusion: Generating synthetic adversarial examples, 2024. URL <https://arxiv.org/abs/2406.19807>.
- [7] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 2013.
- [8] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations(ICLR)*, 2024.
- [9] Jianqi Chen, Hao Chen, Keyan Chen, Yilan Zhang, Zhengxia Zou, and Zhenwei Shi. Diffusion models for imperceptible and transferable adversarial attack, 2025.
- [10] Kangjie Chen, Shangwei Guo, Tianwei Zhang, Xiaofei Xie, and Yang Liu. Stealing deep reinforcement learning models for fun and profit. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, 2021.
- [11] Xinquan Chen, Xitong Gao, Juanjuan Zhao, Kejiang Ye, and Cheng-Zhong Xu. Advdiffuser: Natural adversarial example synthesis with diffusion models. In *International Conference on Computer Vision (ICCV)*, 2023. doi: 10.1109/ICCV51070.2023.00421.
- [12] Jacob K Christopher, Stephen Baek, and Ferdinando Fioretto. Projected generative diffusion models for constraint satisfaction. *arXiv preprint arXiv:2402.03559*, 2024.
- [13] Xuelong Dai, Kaisheng Liang, and Bin Xiao. Advdiff: Generating unrestricted adversarial examples using diffusion models, 2024. URL <https://arxiv.org/abs/2307.12499>.
- [14] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Advances in neural information processing systems(NeurIPS)*, 2021.
- [15] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. <https://openreview.net/forum?id=BJfvknCqFQ>, 2018.
- [16] Tim Franzmeyer, Stephen Marcus McAleer, Joao F. Henriques, Jakob Nicolaus Foerster, Philip Torr, Adel Bibi, and Christian Schroeder de Witt. Illusory attacks: Information-theoretic detectability matters in adversarial attacks. In *International Conference on Learning Representations(ICLR)*, 2024.
- [17] 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(ICLR)*, 2020.

- [18] Qi Guo, Shanmin Pang, Xiaojun Jia, Yang Liu, and Qing Guo. Efficient generation of targeted and transferable adversarial examples for vision-language models via diffusion models, 2024. URL <https://arxiv.org/abs/2404.10335>.
- [19] Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xuelong Li. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. In *Conference on Neural Information Processing Systems(NeurIPS)*, 2023.
- [20] Sihong He, Songyang Han, Sanbao Su, Shuo Han, Shaofeng Zou, and Fei Miao. Robust multi-agent reinforcement learning with state uncertainty. *Transactions on Machine Learning Research*, 2023.
- [21] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- [22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in neural information processing systems(NeurIPS)*, 2020.
- [23] Mengdi Huai, Jianhui Sun, Renqin Cai, Liuyi Yao, and Aidong Zhang. Malicious attacks against deep reinforcement learning interpretations. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.
- [24] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. *arXiv:1702.02284*, 2017.
- [25] Yunhan Huang and Quanyan Zhu. Deceptive reinforcement learning under adversarial manipulations on cost signals. In *Decision and Game Theory for Security(GameSec)*, 2019.
- [26] Inaam Ilahi, Muhammad Usama, Junaid Qadir, Muhammad Umar Janjua, Ala I. Al-Fuqaha, Dinh Thai Hoang, and Dusit Niyato. Challenges and countermeasures for adversarial attacks on deep reinforcement learning. *CoRR*, 2020.
- [27] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.
- [28] Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient diffusion policies for offline reinforcement learning. 2024.
- [29] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in neural information processing systems(NeurIPS)*, 2022.
- [30] Bangalore Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A. Al Sallab, Senthil Kumar Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *CoRR*, abs/2002.00444, 2020. URL <https://arxiv.org/abs/2002.00444>.
- [31] Ezgi Korkmaz. Adversarial robust deep reinforcement learning requires redefining robustness. In *Proceedings of the AAAI Conference on Artificial Intelligence(AAAI)*, 2023.
- [32] Ezgi Korkmaz and Jonah Brown-Cohen. Detecting adversarial directions in deep reinforcement learning to make robust decisions. In *International Conference on Machine Learning(ICML)*, 2023.
- [33] Xian Yeow Lee, Sambit Ghadai, Kai Liang Tan, Chinmay Hegde, and Soumik Sarkar. Spatiotemporally constrained action space attacks on deep reinforcement learning agents. In *Proceedings of the AAAI conference on artificial intelligence(AAAI)*, 2020.
- [34] Edouard Leurent. An environment for autonomous driving decision-making. <https://github.com/eleurent/highway-env>, 2018.
- [35] Haoran Li, Zicheng Zhang, Wang Luo, Congying Han, Yudong Hu, Tiande Guo, and Shichen Liao. Towards optimal adversarial robust q-learning with bellman infinity-error. In *International Conference on Machine Learning(ICML)*, 2024.

- [36] Yongyuan Liang, Yanchao Sun, Ruijie Zheng, and Furong Huang. Efficient adversarial training without attacking: Worst-case-aware robust reinforcement learning. In *Advances in Neural Information Processing Systems(NeurlPS)*, 2022.
- [37] Yongyuan Liang, Yanchao Sun, Ruijie Zheng, Xiangyu Liu, Benjamin Eysenbach, Tuomas Sandholm, Furong Huang, and Stephen Marcus McAleer. Game-theoretic robust reinforcement learning handles temporally-coupled perturbations. In *International Conference on Learning Representations(ICLR)*, 2024.
- [38] Jiang Liu, Chun Pong Lau, and Rama Chellappa. Diffprotect: Generate adversarial examples with diffusion models for facial privacy protection, 2023. URL <https://arxiv.org/abs/2305.13625>.
- [39] Xiangyu Liu, Chenghao Deng, Yanchao Sun, Yongyuan Liang, and Furong Huang. Beyond worst-case attacks: Robust rl with adaptive defense via non-dominated policies. In *International Conference on Learning Representations(ICLR)*, 2024.
- [40] Buqing Nie, Jingtian Ji, Yangqing Fu, and Yue Gao. Improve robustness of reinforcement learning against observation perturbations via 1 lipschitz policy networks. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI’24/IAAI’24/EAAI’24, 2025. URL <https://doi.org/10.1609/aaai.v38i13.29360>.
- [41] E. S. PAGE. Continuous inspection schemes. *Biometrika*, 41(1-2):100–115, 06 1954.
- [42] Alessio Russo and Alexandre Proutiere. Balancing detectability and performance of attacks on the control channel of markov decision processes. In *2022 American Control Conference (ACC)*, 2022.
- [43] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017. URL <https://arxiv.org/abs/1705.05065>.
- [44] David Silver, Aja Huang, Christopher J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016.
- [45] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations(ICLR)*, 2021.
- [46] Xiaolin Sun and Zizhan Zheng. Belief-enriched pessimistic q-learning against adversarial state perturbations. In *International Conference on Learning Representations(ICLR)*, 2024.
- [47] Yanchao Sun, Ruijie Zheng, Yongyuan Liang, and Furong Huang. Who is the strongest enemy? towards optimal and efficient evasion attacks in deep rl. In *International Conference on Learning Representations(ICLR)*, 2022.
- [48] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.
- [49] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 2004. doi: 10.1109/TIP.2003.819861.
- [50] Eric Wong, Frank Schmidt, and Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. In *International conference on machine learning(ICML)*, 2019.
- [51] Marek Wydmuch, Michał Kempka, and Wojciech Jaśkowski. ViZDoom Competitions: Playing Doom from Pixels. *IEEE Transactions on Games*, 2019.

- 1218 [52] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Advgan:  
1219 Generating adversarial examples via generative adversarial networks. In *International Joint*  
1220 *Conferences on Artificial Intelligence(IJCAI)*, 2018.
- 1221 [53] Chaowei Xiao, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song. Spatially trans-  
1222 formed adversarial examples. In *International Conference on Learning Representations(ICLR)*,  
1223 2018.
- 1224 [54] Zikang Xiong, Joe Eappen, He Zhu, and Suresh Jagannathan. Defending observation attacks  
1225 in deep reinforcement learning via detection and denoising. In *2022 European Conference on*  
1226 *Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2022.
- 1227 [55] Haotian Xue, Alexandre Araujo, Bin Hu, and Yongxin Chen. Diffusion-based adversarial  
1228 sample generation for improved stealthiness and controllability, 2024. URL <https://arxiv.org/abs/2305.16494>.  
1229
- 1230 [56] Zhihe YANG and Yunjian Xu. DMBP: Diffusion model-based predictor for robust offline  
1231 reinforcement learning against state observation perturbations. In *International Conference on*  
1232 *Learning Representations(ICLR)*, 2024.
- 1233 [57] Weirui Ye, Yunsheng Zhang, Haoyang Weng, Xianfan Gu, Shengjie Wang, Tong Zhang,  
1234 Mengchen Wang, Pieter Abbeel, and Yang Gao. Reinforcement learning with foundation priors:  
1235 Let embodied agent efficiently learn on its own. In *8th Annual Conference on Robot Learning*,  
1236 2024.
- 1237 [58] Bohang Zhang, Du Jiang, Di He, and Liwei Wang. Rethinking lipschitz neural networks  
1238 and certified robustness: A boolean function perspective. In *Advances in Neural Information*  
1239 *Processing Systems(NeurIPS)*, 2022.
- 1240 [59] Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-  
1241 Jui Hsieh. Robust deep reinforcement learning against adversarial perturbations on state  
1242 observations. In *Advances in Neural Information Processing Systems(NeurIPS)*, 2020.
- 1243 [60] Huan Zhang, Hongge Chen, Duane S Boning, and Cho-Jui Hsieh. Robust reinforcement learning  
1244 on state observations with learned optimal adversary. In *International Conference on Learning*  
1245 *Representations(ICLR)*, 2021.
- 1246 [61] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreason-  
1247 able effectiveness of deep features as a perceptual metric. In *IEEE / CVF Computer Vision and*  
1248 *Pattern Recognition Conference(CVPR)*, 2018.
- 1249 [62] Xuezhou Zhang, Yuzhe Ma, Adish Singla, and Xiaojin Zhu. Adaptive reward-poisoning attacks  
1250 against reinforcement learning. In *International Conference on Machine Learning(ICML)*,  
1251 2020.
- 1252 [63] Sheng Zhao, Dheeru Dua, and Sameer Singh. Towards natural and robust adversarial examples.  
1253 *arXiv preprint arXiv:1710.11342*, 2017.
- 1254 [64] Sheng Zhao, Dheeru Dua, and Sameer Singh. Generating adversarial examples with adversarial  
1255 generative nets. *arXiv preprint arXiv:1801.02613*, 2018.
- 1256 [65] Chong Zhou and Randy C. Paffenroth. Anomaly detection with robust deep autoencoders. In  
1257 *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and*  
1258 *Data Mining*, 2017.