Targeted Attack on Deep RL-based Autonomous Driving with Learned Visual Patterns

Prasanth Buddareddygari[1], Travis Zhang[2], Yezhou Yang[1], and Yi Ren[3]

***Abstract*— Recent studies demonstrated the vulnerability of control policies learned through deep reinforcement learning against adversarial attacks, raising concerns about the application of such models to risk-sensitive tasks such as autonomous driving. Threat models for these demonstrations are limited to (1) targeted attacks through real-time manipulation of the agent's observation, and (2) untargeted attacks through manipulation of the physical environment. The former assumes full access to the agent's states/observations at all times, while the latter has no control over attack outcomes. This paper investigates the feasibility of targeted attacks through visually learned patterns placed on physical objects in the environment, a threat model that combines the practicality and effectiveness of the existing ones. Through analysis, we demonstrate that a pre-trained policy can be hijacked within a time window, e.g., performing an unintended self-parking, when an adversarial object is present. To enable the attack, we adopt an assumption that the dynamics of both the environment and the agent can be learned by the attacker. Lastly, we empirically show the effectiveness of the proposed attack on different driving scenarios, perform a location robustness test, and study the tradeoff between the attack strength and its effectiveness. Code is available at https://github.com/ASU-APG/Targeted-Physical-Adversarial-Attacks-on-AD**

I. Introduction

"Attack is the secret of defense; defense is the planning of an attack."

– Sun Tzu, The Art of War, 5th century BC

Deep reinforcement learning (RL) has grown tremendously in the past few years, producing close-to-human control policies on various tasks [1], [2], [3], [4] including solving Atari games [2], [5], robot manipulation [6], autonomous driving [7], [8], and many others [3]. However, deep neural networks (DNNs) are vulnerable to adversarial attacks, with demonstrations in real world applications such as computer vision [9], [10], [11], [12], natural language processing [13], and speech [14]. Recent studies showed that deep RL agents, due to their adoption of DNNs for value or policy approximation, are also susceptible to such attacks [15], [16], [17].

The threat models in the deep RL domain form two categories: the first assumes that the attacker can directly manipulate the states/observations or actions of the agent, while the second performs the attack through physical objects placed in the environment. Among the first category, Huang et al. [15] proposed to directly perturb the agent's observations at *all* time steps during a roll-out. Similarly, Lin et al. [16] proposed to attack during a chosen subset of time steps. Applications of this type of attacks to autonomous driving have been shown to be effective [17], [18]. Weng et al. [19] showed that learning the dynamics of agents and environments improves the efficacy of the attack in comparison to model-free methods. This category of threats, however, are not practical as they require **direct access to the agents' perception modules to modify their observations**. Such a strong prerequisite condition to launch attacks significantly limits the power of these threats.

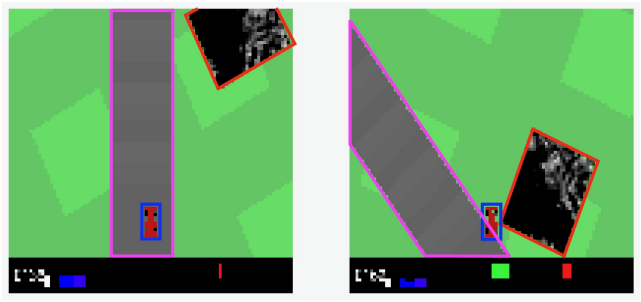


Fig. 1: Targeted adversarial attack on autonomous driving agent using an object fixed to the ground. The attack formulation incorporates the dynamics of the object subject to the pretrained policy of the agent and the object itself. **Left:** initial state. **Right:** achieved target state. The red, blue, and pink bounding boxes indicate the learned adversarial visual patterns, car, and road track, respectively.

For a more feasible approach, researchers studied attacks where adversarial objects are placed in the environments to fool DNNs. Such attacks have been proven effective in general applications such as image classification [20], [21], [22] and face recognition [23]. Specific to deep RL, Kong et al. [24] and Yang et al. [25] demonstrated the existence of physical adversaries, in the form of advertising sign boards and patterns painted on the road respectively, that can successfully mislead autonomous driving systems. While their models are more practical, most of the existing attempts of this type are **not targeted** towards reaching a certain goal state. Instead, they seek to maximize the deviation of actions in the presence of adversaries from the benign policy. These loose-end attacks would only be considered effective when the final state turns out to be disastrous. This is not guaranteed and thus the attack results vary.

[1] PB and YY are with the Active Perception Group at the School of Computing and Augmented Intelligence, Arizona State University, Tempe, AZ 85287, USA {pbuddare, yz.yang}@asu.edu

[2] TZ is with the College of Engineering, Cornell University, Ithaca, NY, 14853, USA. This work was done while TZ was an intern at the Active Perception Group, Arizona State University. tz98@cornell.edu

[3] YR is with the School for Engineering of Matter, Transport, and Energy, Arizona State University, Tempe, AZ 85287, USA yiren@asu.edu

Launching targeted attacks without direct access to the agent's perception modules remains an open challenge. To achieve this, we use the assumptions that the attacker can learn differentiable dynamical models that predict the transition of the environment and has access to the dynamics of the agent's own states with respect to the agent's actions. We argue that these assumptions are reasonable since the environment (e.g., a particular segment of the highway) is accessible to all, including the attacker, and agent dynamics (e.g., for vehicles) is common knowledge. Lastly, since we focus on the existence of policy vulnerability, we assume the agent's policy model to be white-box.

To the best of our knowledge, our work presented in this paper is the first to investigate the existence of *targeted* attacks on deep RL models using adversarial objects in the environment. Specifically, we examine the existence of static and structured perturbations on the environment so that within a time window, the agent is led to a state that is specified by the attacker (Fig. 1). If successful, the study will expose real-world threat models. For instance, a hacker could place an adversarial billboard sign next to the road to cause self-driving cars to veer off the track without directly modifying the cars' observations.

The contributions of our paper are as follows:

- Our presented attack algorithm generates a **static perturbation** that can be directly realized through an object placed in the environment to mislead the agent towards a pre-specified state within a time window.
- We perform ablation studies to show that the choices of the time window and the attack target are correlated. Therefore, fine-tuning of the loss function of the attack with respect to the time window is necessary for identifying successful attacks.
- We study the **robustness of the derived attacks** with respect to their relative locations to the agent, and show that moving the attack object partially out of the sight of the agent will reduce the effect of the attack.

II. Related Work

A. Adversarial attacks on RL agents

Adversarial attacks in RL, especially in the deep RL domain, have gained attention [15], [16], [17] following those for DNNs [9], [26], [12]. In the Atari environment, Lin et al. [16] proposed a strategically timed attack which focuses on finding the right time when an adversarial attack needs to be performed, and an enchanting attack, a targeted attack that generates adversarial examples in order to find actions that lead to a target state. Kos et al. [27] proposed methods for generating value function-based adversarial examples and Behzadan et al. [28] studied adversarial attacks on deep Q networks (DQN) along with transferability to different DQN models. Further, Pattanaik et al. [29] proposed a gradient-based attack on double deep Q networks (DDQN) and deep deterministic policy gradient (DDPG), and developed a robust control framework through adversarial training. Weng et al. [19] proposed model-based adversarial attacks on MuJoCo [30] domains using a target state as the attack goal similar to the enchanting attack presented by Lin et al. [16]. More recently, Zhang et al. [31] proposed a state-adversarial Markov decision process and studied adversarial attacks on model-free deep RL algorithms. While all these aforementioned works have shown that deep RL systems are vulnerable to adversarial attacks, few have explored a target-controlled attack using a dynamical model as presented in this work.

B. Physical Adversarial Attacks

There are a few recent works that focused on physical adversarial attacks [21], [23]. With respect to multi-agent environments, Gleave et al. [32] primarily focused on training an adversarial agent to exploit the weaknesses of traditionally-trained deep RL agents. However, their study, being in a multi-agent setting, does not allow for physical objects to be placed in the environment and is different from the threat model proposed in this paper. Kong et al. [24] proposed a generative model that takes a 3D video slice as input and generates a single physical adversarial example. More recently, the method proposed by Yang et al. [25] optimizes physical perturbations on a set of frame sequences and places them directly on the environment using a differentiable mapping of the perturbations in 2D space to the 3D space. However, both of these methods do not consider a target state for the agent to reach in the presence of physical adversarial examples. Boloor et al. [33] showed a targeted attack on autonomous driving systems called a hijacking attack, where the agent takes a targeted path of actions pre-specified by the attacker. However, our approach differs by letting the attacker choose a final target state and using our attack algorithm to internally learn the path of actions to reach the target.

III. Targeted Attack via Learned Visual Patterns of Physical Objects

We formulate our task as attacking a deep RL system with the adversarial object to be continuously effective at misguiding the agent, while the agent is moving in the environment due to the dynamics of the agent. This is a key difference we claim from existing deep RL attacks that prior works consider. Such a requirement is necessary for the goal of manipulating the agent in a non-trivial way, leading to a guaranteed effective attack. Moreover, unlike perturbations in the state or actions spaces in existing attacks, we perturb a static rectangular area fixed to the environment. In the following section, we introduce the notation, problem statement, and technical details towards a solution.

A. Notations and preliminaries

Let $o_t \in [0,1]^{w \times h \times c}$ be a grey-scale image with width w, height h, and channel size c, that represents the state (scenes) of the underlying Markov decision process (MDP). In the experiment, o_t is the stack of the last four top-down views of a driving scene, resembling a simplification of data obtained through LIDAR. We use o_t as the most recent image of the stack. $a_t \in [0,1]^n$ is the action vector chosen by the agent at time t, and n is the number of continuous actions to be determined. In the experiment, the actions include the

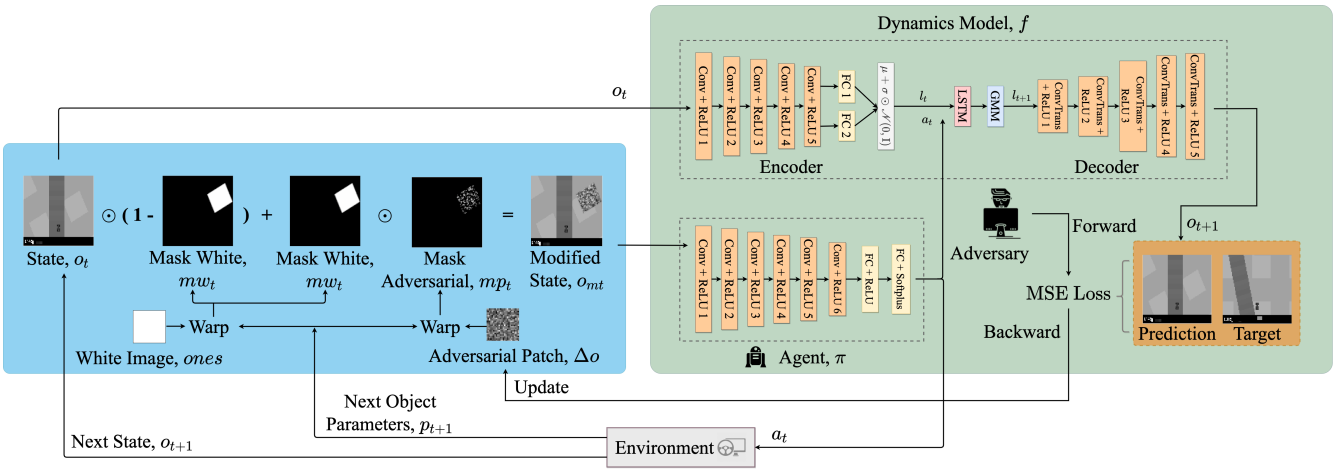


Fig. 2: Illustration of physical adversarial attack in OpenAI Gym's CarRacing-v0 environment. The blue panel shows the crafting of modified observation by adversary through planting and updating the physical object. Adversary creates a new dynamics model and it is assumed that pre-trained agent policy is known as shown in the green panel. The orange panel shows the optimization performed by adversary to perform physical adversarial attack by minimizing the loss between prediction from dynamics model and the predefined target observation.

normalized braking and acceleration rates and the change of steering angle.

Let $\pi : [0,1]^{w\times h\times c} \rightarrow [0,1]^n$ be a deterministic policy learned on the MDP with c equaling 1 to represent grayscale images, and let

$$f : [0,1]^{w\times h\times c} \times [0,1]^n \rightarrow [0,1]^{w\times h\times c}, \tag{1}$$

be the dynamics model of the environment that gives the next state o_{t+1} when action a_t is taken. We note that the agent, as a dynamical system, has its own state defined by normalized $\delta_t \in [0,1]^k$, where k is the number of properties. In the experiment, δ_t is represented by the position, velocity, and steering angle of the vehicle. We denote the dynamics of the agent as:

$$g : [0,1]^k \times [0,1]^n \rightarrow [0,1]^k. \tag{2}$$

We consider attacks in the form of a grey-scale image (perturbation) in a fixed rectangular area of the environment. This image, without transformation, is denoted by Δo, and its global coordinates by Φ. To integrate this image into the scene (o_t), the following differentiable procedure is programmed:

(1) The relative position of the adversarial rectangle in the scene, denoted by p_t, is first calculated based on the agent dynamics, g, the object's global coordinates, Φ, and a transformation function, ψ as $p_t = \psi(\delta_t, \Phi)$, where $\delta_t = g(\delta_{t-1}, a_{t-1})$.

(2) Let $ones$ be a matrix of ones. A mask $mw_t \in \{0,1\}^{w\times h}$ is created based on p_t and $ones$ using homography estimation, realized through the $warp$ function in Kornia [34]. mw_t only has 1s within the rectangle.

(3) A transformed adversarial image $mp_t \in [0,1]^{w\times h}$ is created based on p_t and Δo, again using homography estimation.

(4) Lastly, we integrate the adversarial image into the view:

$$o_{mt} = o_t \odot (1 - mw_t) + mw_t \odot mp_t, \tag{3}$$

where $\odot$ is the element-wise product. Although the homography estimation and warping procedure described above are similar to [25], our unique differentiable layer implementation allows solving through gradient-based methods rather than the local linearization approach presented in [25].

B. Problem statement

Given the initial state o_0 (which contains duplicates of the initial scene), the initial agent state δ_0, the pretrained policy π, the dynamical models $f(\cdot,\cdot)$ and $g(\cdot,\cdot)$, and the transformation function, $\psi(\delta, \Phi)$, we search for an image Δo, with $||\Delta o||_\infty \leq \epsilon$, that leads the agent to a specific target o_{target} within the time window $[0, T]$. Formally:

$$\begin{aligned} \min_{||\Delta o||_\infty \leq \epsilon} \quad & \sum_{t=1}^{T} ||o_t - o_{target}||_2^2. \\ s.t. \quad & a_t = \pi(o_{mt}), \\ & o_{mt} = o_t \odot (1 - mw_t) + mw_t \odot mp_t \\ & mw_t = warp(ones, p_t), mp_t = warp(\Delta o, p_t), \\ & p_t = \psi(\delta_t, \Phi), o_{t+1} = f(o_t, a_t), \delta_{t+1} = g(\delta_t, a_t). \end{aligned} \tag{4}$$

The dependency of variables involved in this problem is visualized in Fig. 2. The loss function of Eq. (4) accepts early convergence of the agent's state to the target. Notice that we use scenes without the adversarial perturbation in evaluating the loss, since the target state is specified before the attack problem is solved. The use of the learned dynamics model, agent's dynamics and a differentiable implementation of $warp$ together make this problem differentiable with respect to the perturbation Δo, allowing the problem to be solved using gradient-based methods.

C. Learning dynamics of the environment

Here we introduce the procedure for learning a differentiable dynamical model of the environment, which is an essential step to enable a gradient-based attack. We believe that addition of this dynamical model explicitly accounts for

state evolution in the attack generation and also the plan of actions leading to target state. This makes our targeted attack more feasible and easier by letting the attacker specify a target state rather than how to reach that target state.

1) Data collection: We first collect data in the format of (state, action, next state) through multiple rollouts of the environment. Note that a successfully attacked rollout will encounter states different from those experienced through the benign policy, e.g., agent moving out of the highway. To collect a representative dataset, we perform rollouts using the pretrained policy with noise of variable strength τ added to the actions, i.e, $a_t = a_t + \mathcal{N}(0,1) * \tau$ similar to [25]. The noisy actions help explore the environment, allowing the adversary to predict the environment dynamics correctly when approaching the target. The resultant dataset is denoted by $\mathcal{D} = \{(o_i, a_i, o_{i+1})\}_{i=1}^{N}$. We note that such data collection is achievable when launching a real-world attack, as long as the attacker can sample the state transitions towards the specified target by using a vehicle with dynamics similar to the attacked agent.

2) Learning the environment dynamics: Since the environment state contains rich information (e.g., time-variant track and surroundings), feed forward neural networks fail to generalize well on the dataset. Here we follow Ha et al. [35] to construct a dynamical model using a variational autoencoder (VAE) and a mixture-density recurrent neural network (MD-RNN), denoted by $\hat{f}(\cdot,\cdot;w)$, which takes in the environment state and action, and predicts the next environment state. w are trainable parameters. As in [35], we use the same combination of mean square error and Kullback–Leibler divergence as the loss for training the VAE, and the Gaussian mixture loss for training the MD-RNN.

D. Optimization details

We use Alg. 1 to solve the attack problem (Eq. (4)). During each iteration, we obtain the state containing the adversarial image o_{mt} as described in Eq. 4 by computing mp_t and mw_t. To respect the observation limits seen by the agent, we clip o_{mt} between 0 and 1 so that a valid image is yielded. The agent then performs an action on o_{mt} to get a_t. Using the dynamics model, f, future prediction $o_{t+1}^{\dagger}$ is obtained to compute the loss. Finally, we backpropagate the sum of losses within the time window $[0, T]$ in order to update perturbation Δo.

Algorithm 1: Optimization for Targeted Physical Adversarial attack

Input: Number of Iterations, I, environment env, Attack length, T, pretrained policy π, dynamics model, f, target state o_{target}
Output: learned physical perturbation example, Δo

```
i ← 0, seed ← random seed
Δo ← N(0, 1)
while i < I do
    total_loss ← 0, t ← 0
    env.seed(seed)
    o_t = env.reset()
    δ_t ← initial agent state
    while t < T do
        p_t = ψ(δ_t, Φ)
        mw_t, mp_t = warp(ones, p_t), warp(Δo, p_t)
        o_mt = o_t ⊙ (1 − mw_t) + mw_t ⊙ mp_t
        clip o_mt between [0, 1]
        a_t = π(o_mt)
        o†_{t+1} = f̂(o_t, a_t)
        δ_{t+1} = g(δ_t, a_t)
        total_loss += d(o†_{t+1}, o_target)
        o_{t+1} ← env.step(a_t)
        t ← t + 1
    backpropagate total_loss to update Δo
    clip Δo between [−ϵ, ϵ]
    i ← i + 1
Return Δo
```

IV. Experiments

We use the CarRacing-v0 environment [36] in OpenAI Gym to demonstrate the existence of adversarial objects that misguide an otherwise benign deep RL agent. We used a model-free Actor-Critic algorithm [37] to obtain the pretrained policy π. The policy is trained with a batch size of 128 and 10^5 episodes.

For the dynamical model $\hat{f}$ of the environment, the VAE is trained for 10^3 epochs using the Adam optimizer [38]. We set the batch size to 32 and learning rate to 0.001 with decreasing learning rate based on plateau and early stopping. For the MD-RNN, we train for 10^3 epochs using the same optimizer. We set the batch size to 16, the number of Gaussian models to 5, and the learning rate to the same value as the training of VAE.

For the attack, we set the time span T to 25 and the adversarial bound to $\epsilon = 0.9$. An ablation study will be done on these hyperparameters in Sec. V-D. We use the same optimizer as before, and set the learning rate to 0.005 for $I = 10^3$ iterations. We set the adversarial area to be 25 pixels wide and 30 pixels tall.

A. Baselines

To the best of our knowledge, there have been few results on targeted physical attacks on deep RL agents. Although the work of Yang et al. [25] is similar to ours, we believe that their experiment setting is very different from ours and we thus did not use it as a baseline. Therefore, we use a baseline where Δo is drawn uniformly in $[0,1]^{25\times 30}$. By comparing agent state trajectories in the presence of random and optimized Δo, we will show that the proposed attack is more effective than random perturbations.

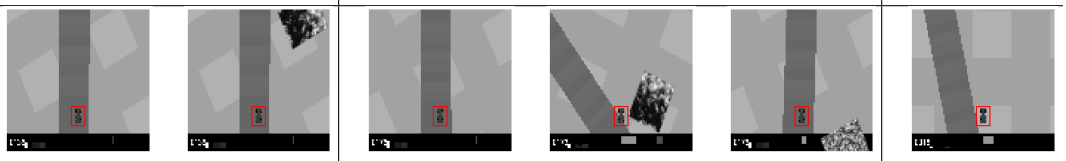
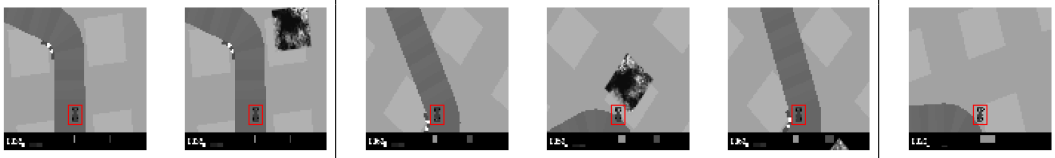
B. Evaluation metrics

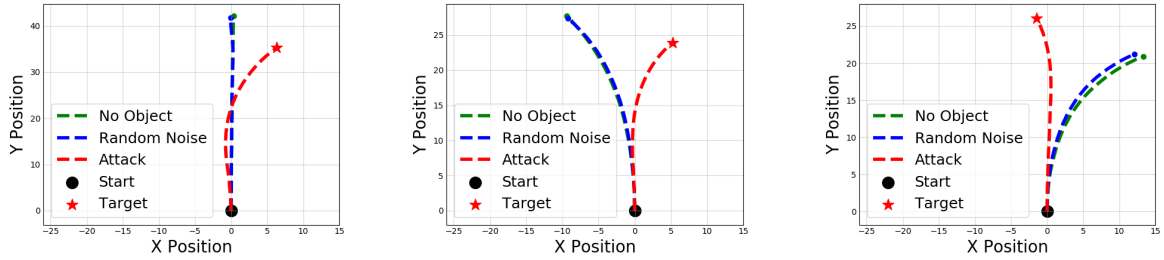
We introduce two metrics to evaluate the effectiveness of an attack: *actions error* and *percentage change of value*. The former is defined as the mean square error between the attacked and benign action values over T timesteps derived from rollouts with and without the adversarial object, respectively. The latter is the percentage change of value from the benign to the attacked rollout, where the value of a policy is the sum of rewards over $[0, T]$.

V. Experiments and Discussion

We evaluate these metrics on three driving scenarios, and compare the trajectories of the agent with and without

TABLE I: Targeted and random attacks in three scenarios. Agent in red boxes. See supplementary video for details.

Scenarios	$(t = 0)$ No attack	Optimal attack	$(t = T)$ No attack	Optimal attack	Random attack	Target state
Straight						
Left turn						
Right turn						



(a) Straight track (b) Left turn track (c) Right turn track

Fig. 3: Trajectories in the three scenarios with no attack, random attack, and optimized attacks.

the attack. Further, we conduct experiments to evaluate the robustness of our attack with respect to different locations of object. Finally, we compare the effectiveness of the attack with varying time span (T) and adversarial bound (ϵ) based on the evaluation metrics.

A. Attack scenarios

We consider three driving scenarios where the agent with the benign policy will go straight, left, and right, respectively. In each of the scenarios, the object is placed at a fixed location in the environment so that it is observable by the agent throughout the attack. We specify the target states as the images shown in Table I.

B. Comparison with the baseline

We compare the trajectories of the agent under the benign policy, the proposed attack, and the random attacks, with trajectory visualizations in Fig. 3 and final states in Tab. I. For the random attack, we conducted 10 independent simulations for each scenario to derive the mean trajectories. The standard deviations in all three scenarios are negligible. X and Y axes in the figure represent the global coordinates.

Results show that while our approach successfully misguides the agent in all scenarios, the agent is not affected as much by the random attacks. Specifically, in scenario 1, the agent goes straight with and without the presence of a random attack. In the presence of the proposed attack, however, the agent deviates from the benign trajectory to reach the target state. The same happens for scenarios 2 and 3. It is worth noting that by comparing Tab. I and Fig. 3, we see that the agent reaches the target location but does not perfectly match the target orientation. For instance, in the straight track scenario, we can see that the optimal attack after $t = T$ time steps forces the car to turn right, partially going off the road, but in the target state, the car is completely off the road on the right. Further exploration of the attack objective may potentially improve the matching of the orientation. Lastly, Table II quantifies the comparison through the evaluation metrics. The proposed attack outperforms the random ones on both metrics.

C. Robustness to translation

We study the robustness of our attack with respect to different global coordinates of the attack object (Φ) placed in the environment. Specifically, we changed the position of adversarial object iteratively in x and y directions during test time with the same learned adversarial pattern. The experiment is performed on the straight track scenario, with fixed dynamical models. In Fig. 4, the (x, y) coordinates represent position of the attack object relative to the actual

TABLE II: Comparison with random noise baseline in terms of evaluation metrics.

Scenarios	Actions Error	Change of value (%)
Straight + Random	0.064	0
Left turn + Random	0.069	0
Right turn + Random	0.046	-10.72
Straight + Proposed	**0.126**	**-17.70**
Left turn + Proposed	**0.138**	**-32.26**
Right turn + Proposed	**0.062**	**-32.15**

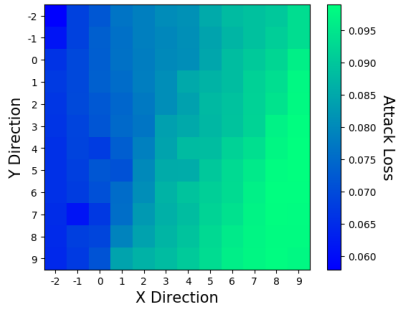


Fig. 4: Attack robustness heatmap on position of physical object in scenario 1

object position, $(0, 0)$ in the original attack (i.e., the one used in the experiments for Tab. I), and the heat map of Fig 4 represents the attack loss of Eq. (4), when the object is moved accordingly. Therefore the blue region represents more successful attacks since the attack loss is lower, whereas the green region represents relatively unsuccessful attacks as attack loss is higher. Note that the range of the figure is bounded by the constraints that the object cannot be on the track and cannot be out of the scene. From this test, if the object is moved towards the track ($-X$ direction in the figure), the attack will still be effective or even better. On the other hand, if the object is moved away from the track and partially out of the scene, the attack becomes less effective, which is reasonable since the agent will have only partial observation of the attack. Further investigating formulations of robust attacks will be valuable.

D. Adversarial bounds and Attack length

TABLE III: Adversarial Bounds *vs* Attack Length

Adversarial Bound ϵ	Attack Length T					
	T = 15		T = 25		T = 30	
	Attack Loss	Actions Error	Attack Loss	Actions Error	Attack Loss	Actions Error
0.1	0.091	0.064	0.090	0.064	0.088	0.063
0.3	0.088	0.078	0.087	0.069	0.085	0.066
0.5	0.086	0.113	0.077	0.107	0.083	0.070
0.9	0.081	0.125	**0.076**	**0.126**	0.078	0.093

Here we perform an ablation study on the attack strength (ϵ) and attack time span (T), by enumerating $\epsilon \in \{0.1, 0.3, 0.5, 0.9\}$ and $T \in \{15, 25, 30\}$. The results in terms of the optimal loss of Eq. (4), and the actions error are summarized in Table III. The experiments are performed on the straight track scenario, with fixed dynamical models. From the results, it is evident that larger ϵ improves the effectiveness of the attacks. Additionally, as we enlarge the time window, the actions error decreases in nearly all cases. Based on our experiments, we believe that if T is smaller, then the attack has a smaller action space to plan on, causing it to alter the actions more aggressively than a bigger T. However, the attack loss increases from $T = 25$ to $T = 30$. By examining the results, we found that this is primarily because the attack object moves out of the scene between $T = 25$ and $T = 30$. As a result of the limited observation of the attack object by the agent, the optimizer struggles to find a way to keep the agent close to the target state, thus the increased loss. This implies that the time window is coupled with the choice of the target state, and its careful selection is important for succeeding in the attack.

VI. Conclusion

Even though autonomous driving agents have been increasingly using deep RL techniques, it is possible that they can be fooled by simply placing an adversarial object in the environment. While previous studies in this domain focused on untargeted attacks without long-term effects, this paper studies the *existence* of static adversarial objects that can continuously misguide a deep RL agent towards a target state within a time window. Using a standard simulator and a pretrained policy, we developed an algorithm that searches for such attacks and showed their existence empirically. For effective search of the attacks, we utilize differentiable dynamical models of the environment, which can be learned through experience collected by the attacker. Our approach has a limitation that the full policy must be known to the attacker (white-box). Additionally, the attack highly depends on the size, position, and pattern of the object. More improvements on these areas are necessary to better understand the practicality of the threat model. Future work will study the existence of robust physical attacks in more complex environments, e.g., with the presence of other agents and with visual or 3D observations. By demonstrating the existence of a new type of attack more practical than digital perturbations, we hope this study can motivate more rigorous research towards robust and safe AI methods for autonomous driving.

VII. Acknowledgments.

This material is based upon work supported by the National Science Foundation under Grant No. #1925403, #2038666 and #2101052. The authors acknowledge support from Amazon AWS Machine Learning Research Award (MLRA), and the Institute of Automated Mobility (IAM), Arizona Commerce Authority. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding entities.

REFERENCES

[1] M. Laskin, A. Srinivas, and P. Abbeel, "CURL: Contrastive unsupervised representations for reinforcement learning," in *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.

[2] A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, Z. D. Guo, and C. Blundell, "Agent57: Outperforming the Atari human benchmark," in *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.

[3] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, *et al.*, "Mastering atari, go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.

[4] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, 2016.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, 2015.

[6] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE international conference on robotics and automation (ICRA)*, 2017, pp. 3389–3396.

[7] S. Wang, D. Jia, and X. Weng, "Deep reinforcement learning for autonomous driving," *arXiv preprint arXiv:1811.11329*, 2018.

[8] X. Liang, T. Wang, L. Yang, and E. Xing, "Cirl: Controllable imitative reinforcement learning for vision-based self-driving," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[10] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[11] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, 2018.

[12] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Trans. Evol. Comput.*, 2019.

[13] R. Jia and P. Liang, "Adversarial examples for evaluating reading comprehension systems," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Sept. 2017.

[14] Y. Qin, N. Carlini, G. Cottrell, I. Goodfellow, and C. Raffel, "Imperceptible, robust, and targeted adversarial examples for automatic speech recognition," in *Proceedings of the 36th International Conference on Machine Learning (ICML)*, Jun 2019.

[15] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel, "Adversarial attacks on neural network policies," *arXiv preprint arXiv:1702.02284*, 2017.

[16] Y.-C. Lin, Z.-W. Hong, Y.-H. Liao, M.-L. Shih, M.-Y. Liu, and M. Sun, "Tactics of adversarial attack on deep reinforcement learning agents," *arXiv preprint arXiv:1703.06748*, 2017.

[17] Y. Huang and S.-h. Wang, "Adversarial manipulation of reinforcement learning policies in autonomous agents," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–8.

[18] J. Sun, T. Zhang, X. Xie, L. Ma, Y. Zheng, K. Chen, and Y. Liu, "Stealthy and efficient adversarial attacks against deep reinforcement learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[19] T.-W. Weng, K. D. Dvijotham, J. Uesato, K. Xiao, S. Gowal, R. Stanforth, and P. Kohli, "Toward evaluating robustness of deep reinforcement learning with continuous control," in *International Conference on Learning Representations (ICLR)*, 2020.

[20] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.

[21] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, Jul 2018.

[22] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning visual classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[23] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[24] Z. Kong, J. Guo, A. Li, and C. Liu, "Physgan: Generating physical-world-resilient adversarial examples for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[25] J. Yang, A. Boloor, A. Chakrabarti, X. Zhang, and Y. Vorobeychik, "Finding physical adversarial examples for autonomous driving with fast and differentiable image compositing," *arXiv preprint arXiv:2010.08844*, 2020.

[26] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP)*, 2017.

[27] J. Kos and D. Song, "Delving into adversarial attacks on deep policies," *arXiv preprint arXiv:1705.06452*, 2017.

[28] V. Behzadan and A. Munir, "Vulnerability of deep reinforcement learning to policy induction attacks," in *International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM)*, 2017.

[29] A. Pattanaik, Z. Tang, S. Liu, G. Bommannan, and G. Chowdhary, "Robust deep reinforcement learning with adversarial attacks," *arXiv preprint arXiv:1712.03632*, 2017.

[30] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[31] H. Zhang, H. Chen, C. Xiao, B. Li, M. Liu, D. Boning, and C.-J. Hsieh, "Robust deep reinforcement learning against adversarial perturbations on state observations," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 21 024–21 037.

[32] A. Gleave, M. Dennis, C. Wild, N. Kant, S. Levine, and S. Russell, "Adversarial policies: Attacking deep reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2020.

[33] A. Boloor, K. Garimella, X. He, C. Gill, Y. Vorobeychik, and X. Zhang, "Attacking vision-based perception in end-to-end autonomous driving models," *Journal of Systems Architecture*, vol. 110, p. 101766, 2020.

[34] E. Riba, D. Mishkin, D. Ponsa, E. Rublee, and G. Bradski, "Kornia: an open source differentiable computer vision library for pytorch," in *Winter Conference on Applications of Computer Vision*, 2020.

[35] D. Ha and J. Schmidhuber, "Recurrent world models facilitate policy evolution," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018, pp. 2450–2462.

[36] O. Klimov, "Carracing-v0," in *http://gym.openai.com/*, 2016.

[37] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, 2016.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.