Enduring, Efficient and Robust Trajectory Prediction Attack in Autonomous Driving via Optimization-Driven Multi-Frame Perturbation Framework

Yi Yu, Weizhen Han, Libing Wu, Bingyi Liu, Enshu Wang, Zhuangzhuang Zhang
{yui1212, wu, wanges17}@whu.edu.cn
{hanweizhen, byliu}@whut.edu.cn {zhuangzhuang.zhang}@cityu.edu.hk

A. Location Optimization Loss Functions

We design three specific loss functions to guide the optimization process. These functions focus on per-frame positional distance, directional alignment, and overall trajectory shape similarity respectively. In this section, we will provide a detailed explanation of each of these three loss functions.

A.1. Pose Loss

The pose loss $\mathcal{L}_{pose}$ quantifies the spatial difference between the detected perturbations P_t^{det} and the target perturbations P_t^{tar}. It is calculated using the following formula:

$$\mathcal{L}_{pose} = \frac{\frac{1}{n}\sum_{i=t-n+1}^{t} \left\| p_i^{det} - p_i^{tar} \right\|^2}{D_{max}}, \tag{1}$$

Where $D_{max} = \max\left(\left\| p_i^{det} - p_i^{tar} \right\|^2\right)$ represents the maximum distance between perturbations and is employed for normalization.

A.2. Heading Loss

The angular deviation θ between the directional vectors of the two types of state perturbations is measured by the heading loss $\mathcal{L}_{heading}$. $\mathcal{L}_{heading}$ ensures that the direction remains the same for each time step, as computed by the following formula:

$$\mathcal{L}_{heading} = -\left(\frac{1}{n}\sum_{i=t-n+1}^{t} \cos\theta_i\right), \tag{2}$$

where θ_i is the angular difference between headings in time step i.

A.3. Shape Loss

Dynamic Time Warping (DTW) [6, 9] is used in our method to measure the similarity between the shapes of two trajectories —— one from the detected perturbations P_t^{det} and the other from the target perturbations P_t^{tar}. DTW is especially suitable for comparing time series or sequences that may differ in length or are not perfectly aligned in time. Given $P_t^{det} = \{p_1^{det}, \ldots, p_n^{det}\}$ and $P_t^{tar} = \{p_1^{tar}, \ldots, p_n^{tar}\}$ are the sequences of positions from the detected and target perturbations, where p_i^{det} and p_j^{tar} represent the spatial coordinates (x, y) at frame i and j, respectively. The goal of DTW is to find the optimal alignment path π between these two trajectories to minimize the total alignment distance.

Local distance matrix. We first compute the local distance matrix D, where $D(i,j)$ represents the Euclidean distance between p_i^{det} and p_j^{tar}:

$$D(i,j) = \left\| p_i^{det} - p_j^{tar} \right\|. \tag{3}$$

Cumulative distance matrix. Next, we construct the cumulative distance matrix C, where $C(i,j)$ is the minimum cumulative distance to reach point (i,j) from the start: $C(i,j) = D(i,j) + \min(C(i-1,j), C(i,j-1), C(i-1,j-1))$. This recursive formula ensures that each point (i,j) in the matrix is aligned with the minimal cost from its neighbors.

Optimal warping path. Finally, we find the optimal warping path $\pi = \{(i_1,j_1), (i_2,j_2), \ldots, (i_k,j_k)\}$ that minimizes the cumulative distance. The total DTW distance along the optimal path is:

$$D_{DTW} = \sum_{(i,j)\in\pi} D(i,j), \tag{4}$$

where the path must satisfy boundary, monotonicity, and continuity conditions, meaning it starts at $(1,1)$ and ends at (n,n), moving either right, up, or diagonally.

Shape loss. The DTW distance D_{DTW} reflects the dissimilarity between the detected perturbations and the target perturbations. To express the shape loss $\mathcal{L}_{shape}$ with D_{DTW}, we normalize it as:

$$\mathcal{L}_{shape} = \frac{D_{DTW}}{D_{\max}}, \tag{5}$$

where $D_{\max}$ is a normalization factor, typically the largest possible distance between points in the trajectories. The closer the shape loss is to 0, the more similar the two perturbations are in shape.

The ultimate total loss $\mathcal{L}_{total}$ is calculated as $\mathcal{L}_{total} = \alpha \cdot \mathcal{L}_{pose} + \beta \cdot \mathcal{L}_{heading} + \gamma \cdot \mathcal{L}_{shape}$, which assesses the similarity between P_t^{det} and P_t^{tar} through various methodologies. The total loss helps guide the adversarial locations to generate state perturbations that closely resemble the target in terms of spatial alignment and shape, thereby maximizing the attack's effectiveness. In this paper, we set the weights of the three losses to $\alpha = 0.4$, $\beta = 0.4$, and $\gamma = 0.2$.

(a) The attack scenario in scene-0103 of nuScenes dataset.

(b) The attack scenario in scene-0001 of nuScenes dataset.

Figure 1. Examples of attack scenario, viewed from the perspective of the cam-front on the victim AV.

B. Implementation Details

we further elaborate on the details of our experiment implementation, including the selection criteria of driving scenes, the introduction of autonomous driving (AD) models used, the search space bounds for adversarial objects, and the hyper-parameters settings.

B.1. Selection Criteria for Driving Scenes

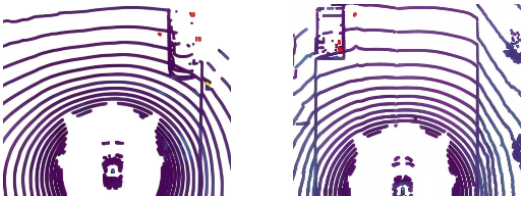
We manually select 50 driving scenes from the nuScenes dataset where the ego vehicle (victim AV) is driving on the road and the adversarial vehicle is parked roadside, as shown in Fig. 1. In each scene, we ensure that the victim AV remains in motion, and at the current frame, the lateral distance between the victim AV and adversarial vehicle is approximately 3-5 meters, with a longitudinal distance of 10-15 meters. During the attack, we identify adversarial locations using annotated key frames captured at 2Hz in the nuScenes dataset. We then maintain the victim AV's speed and evaluate these locations on the unannotated nuScenes dataset sampled at 20Hz.

B.2. Autonomous Driving Models

To ensure a fair comparison with *SinglePoint Attack* [3], we employ the same AD models as it. We use PIXOR [7] for LiDAR detection, CenterPoint Tracker [8] for tracking, and Trajectory++ [5] for trajectory prediction, along with a Model Predictive Control (MPC) planner [1] for motion planning. Following official guidelines, we train the PIXOR model using the nuScenes dataset and directly utilize the official pre-trained Trajectron++ model on the nuScenes dataset. In addition, we also adopt the default configuration provided by the official source.

B.3. Search Bounds of Adversarial Objects

To ensure that the adversarial locations can be executed in the real world, we constrain them to a reasonable, realistic 3D coordinate space. During the attack phase, the search space is limited to a 4x4x1 m^3 area above the adversarial vehicle. In the location optimization phase, we restrict the PSO bounds to a 4x4x0.8 m^3 space above the adversarial vehicle, with the initial location set at (2, 2, 0.4). Additionally, we define the adversarial objects as three circular cardboard pieces. After determining three adversarial locations, we simulate the point cloud detection results for each cardboard piece using four-point clouds. From these point clouds, we can derive the cardboard's center coordinates, radius, and orientation. The orientation is set to face the victim AV, and the radius is set to 0.1 m. For clarity, we visualize the calculated adversarial locations in Fig. 2.



(a) The scenario in scene-0103. (b) The scenario in scene-0001.

Figure 2. Visualization of adversarial locations marked in red.

B.4. Hyper-parameters Settings

In the attack phase, we set n to 5, meaning adversarial attacks are executed over five consecutive time steps. We use the Adam optimizer with a learning rate of 0.1 for the coordinate features and 0.05 for the orientation feature, iterating for 10 epochs with a maximum of 20 iterations per epoch. Additionally, for estimating distributions in comparative experiments, we set the number of randomly generated locations Q ($query$) to 1000. In the location optimization phase, we evaluate both Particle Swarm Optimization (PSO) [2] and the Adaptive Grey Wolf Optimizer (AGWO) [4]. Although both exhibit similar performance, we select PSO due to its simpler implementation and faster search speed. We configure PSO with 10 particles, each with 3x3 dimensions representing the 3 adversarial locations, where each position has x, y, and z coordinates as its features. The inertia

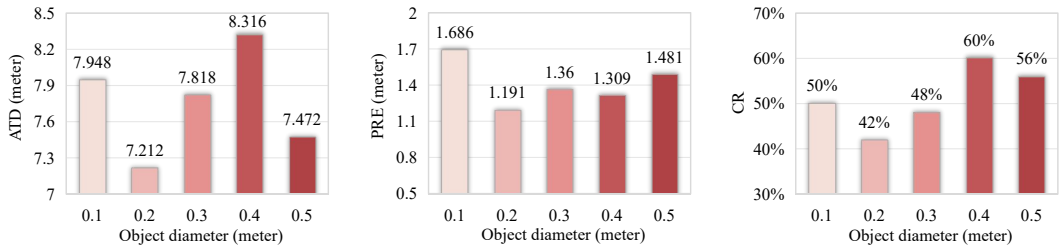


Figure 3. The evaluation results of varying object diameter on the SP-Attack [3]. In the adversarial location search phase of SP-Attack, we define the diameter of the cardboard as 0.2 meters. We then assess the impact of varying diameters at the generated adversarial locations.

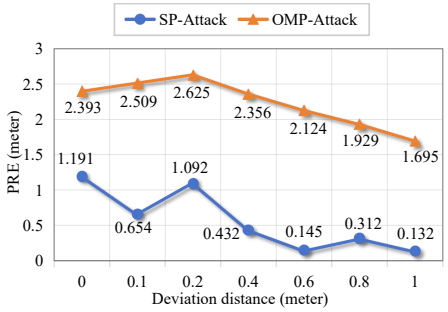


Figure 4. The comparison PRE results of the robustness experiments for the deviation distance between the victim AV and attack point.

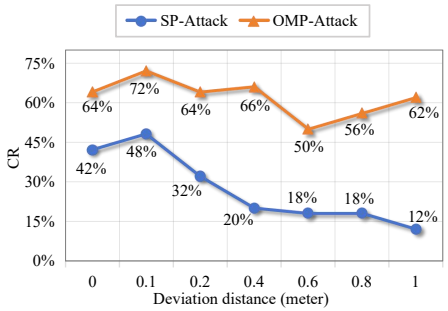


Figure 5. The comparison CR results of the robustness experiments for the deviation distance between the victim AV and attack point.

weight is set to 1.0, and the acceleration coefficients are set to (2.0, 2.0). The number of adversarial objects is set to 3 and the radius of the object is set to 0.1 m. We set the object's placement orientation to 0° relative to the x-axis. For baseline settings, SP-Attack is configured using the official default configuration, with *query* set to 1000.

C. Results of Attack Robustness

Figure 4 and Figure 5 respectively show the comparison results of the planning-response error (PRE) and collision rate (CR) in the robustness experiment regarding the impact of the victim AV's deviation from the attack point. The results

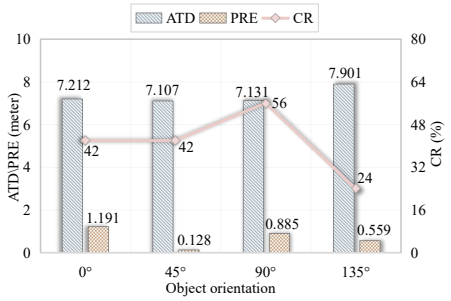


Figure 6. The robustness experiment of different object placement orientations on the SP-Attack [3]. The orientation represents the angle between the object and the positive x-axis, observed from an overhead perspective along the z-axis.

reveal a clear downward trend in both PRE and CR as the offset distance increases. Notably, however, OMP-Attack demonstrates a more gradual rate of decline in performance metrics, with a minimum PRE of 1.695 meters compared to the significantly lower 0.132 meters observed for SP-Attack. Furthermore, the CR for OMP-Attack remains relatively stable, showing no significant reduction across increasing offset distances, whereas SP-Attack experiences a considerable drop. These results underscore the robustness of OMP-Attack, which maintains higher performance levels even as offset distances vary, in contrast to the more sensitive performance of SP-Attack.

Robustness results on object properties of SP-Attack. To comprehensively compare the robustness of object properties, we evaluate the effect of object size and orientation on SP-Attack. We set the cardboard diameter to 0.2 meters and the placement orientation to 0° during attack planning and then adjusted the properties to test the attack's sensitivity. First, we assess the impact of variations in cardboard diameter on SP-Attack. The results in Fig. 3 indicate that the average trajectory distance (ATD) is minimized at the predefined diameter of 0.2 meters in SP-Attack. Adjusting the diameter either upwards or downwards results in a significant rise in ATD, highlighting the sensitivity of SP-Attack to variations in object size. Second, we perform a robustness experiment to explore the influence of object

Table 1. Robustness of adversarial locations against small shifts.

Shift(cm)	0	2	4	6	8
ATD(m) ↓	6.439	6.280	5.815	6.412	6.042
PRE(m) ↑	2.393	2.406	2.227	2.868	2.649
CR↑	64%	63%	56%	69%	56%

Table 2. Comparison of SI algorithms. *: Time per iteration.

Methods	**TPI***(s) ↓	**ATD**(m) ↓	**PRE**(m) ↑	**CR**↑
GWO	0.141	6.687	2.404	64%
AGWO	0.257	6.432	2.557	66%
PSO	0.141	6.439	2.393	64%

Table 3. Results of OMP-Attack combined with randomly placed objects.

Methods	**ATD**(m) ↓	**PRE**(m) ↑	**CR**↑
w PSO (1k queries)	6.439	2.393	64%
w RPO (1k locations)	7.632	0.964	32%
w RPO (3k locations)	6.698	1.929	54%
w RPO (5k locations)	6.278	2.522	70%

orientation on SP-Attack. The results presented in Fig. 6 demonstrate that SP-Attack exhibits a degree of robustness to cardboard orientation. However, the CR is heavily influenced by the direction, with the CR peaking at 12.8% in the worst case. This suggests that the threat posed by the attack is dependent on the placement orientation of objects. These sensitivities indicate that the effectiveness of SP-Attack is closely tied to the specific adversarial object's properties, which can limit its robustness across varying conditions.

Robustness results on adversarial location shifts. To validate the stability of our *Vague Optimization* strategy, we evaluate the robustness of adversarial location offsets ranging from 0 to 8 cm. As shown in Tab. 1, our framework exhibits high robustness, achieving an ATD of below 6.5 m and a collision rate of above 55%. This is because auxiliary points near the center point also retain a strong attack effect. Notably, in real-world scenarios, effective attacks can be achieved by ensuring the cardboard covers a sufficient area of the adversarial locations, rather than requiring precise alignment. This observation further corroborates the robustness of adversarial locations against minor spatial deviations.

D. Additional Ablation Experiments

This section presents additional ablation experiments, including a comparison between PSO and the latest swarm intelligence (SI) algorithm, as well as an analysis of the impact of randomly placed objects on the location search stage.

The comparison of SI algorithms. For the location optimization phase, PSO surpasses the latest SI algorithm (*e.g.*, AGWO [4], GWO) in search speed while achieving comparable performance. We test PSO, AGWO, and GWO under the same setups. As shown in Tab. 2, while PSO performs slightly worse than AGWO in ATD and CR, it reduces the *time per iteration* (TPI) by nearly half compared to AGWO, making it practical for real-world attacks.

The impact of randomly placed objects. We conduct additional ablation studies where PSO is replaced by the **r**andomly **p**laced **o**bjects (RPO) method. RPO selects locations where the perturbations are closest to the computed ones. As shown in Tab. 3, RPO requires 5k random locations to achieve performance compared to PSO with only 1k queries (involving 10 particles and 100 iterations), demonstrating the superior efficiency of our framework.

References

[1] Yuxiao Chen, Ugo Rosolia, Wyatt Ubellacker, Noel Csomay-Shanklin, and Aaron D Ames. Interactive multi-modal motion planning with branch model predictive control. *IEEE Robotics and Automation Letters*, 7(2):5365–5372, 2022. 2

[2] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-International Conference on Neural Networks*, pages 1942–1948. ieee, 1995. 2

[3] Yang Lou, Yi Zhu, Qun Song, Rui Tan, Chunming Qiao, Wei-Bin Lee, and Jianping Wang. A first Physical-World trajectory prediction attack via LiDAR-induced deceptions in autonomous driving. In *33rd USENIX Security Symposium*, pages 6291–6308, 2024. 2, 3

[4] Kazem Meidani, AmirPouya Hemmasian, Seyedali Mirjalili, and Amir Barati Farimani. Adaptive grey wolf optimizer. *Neural Computing and Applications*, 34(10):7711–7731, 2022. 2, 4

[5] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, pages 683–700. Springer, 2020. 2

[6] Aleksandar Vakanski, Iraj Mantegh, Andrew Irish, and Farrokh Janabi-Sharifi. Trajectory learning for robot programming by demonstration using hidden markov model and dynamic time warping. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(4):1039–1052, 2012. 1

[7] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018. 2

[8] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11784–11793, 2021. 2

[9] Jiaping Zhao and Laurent Itti. shapedtw: Shape dynamic time warping. *Pattern Recognition*, 74:171–184, 2018. 1