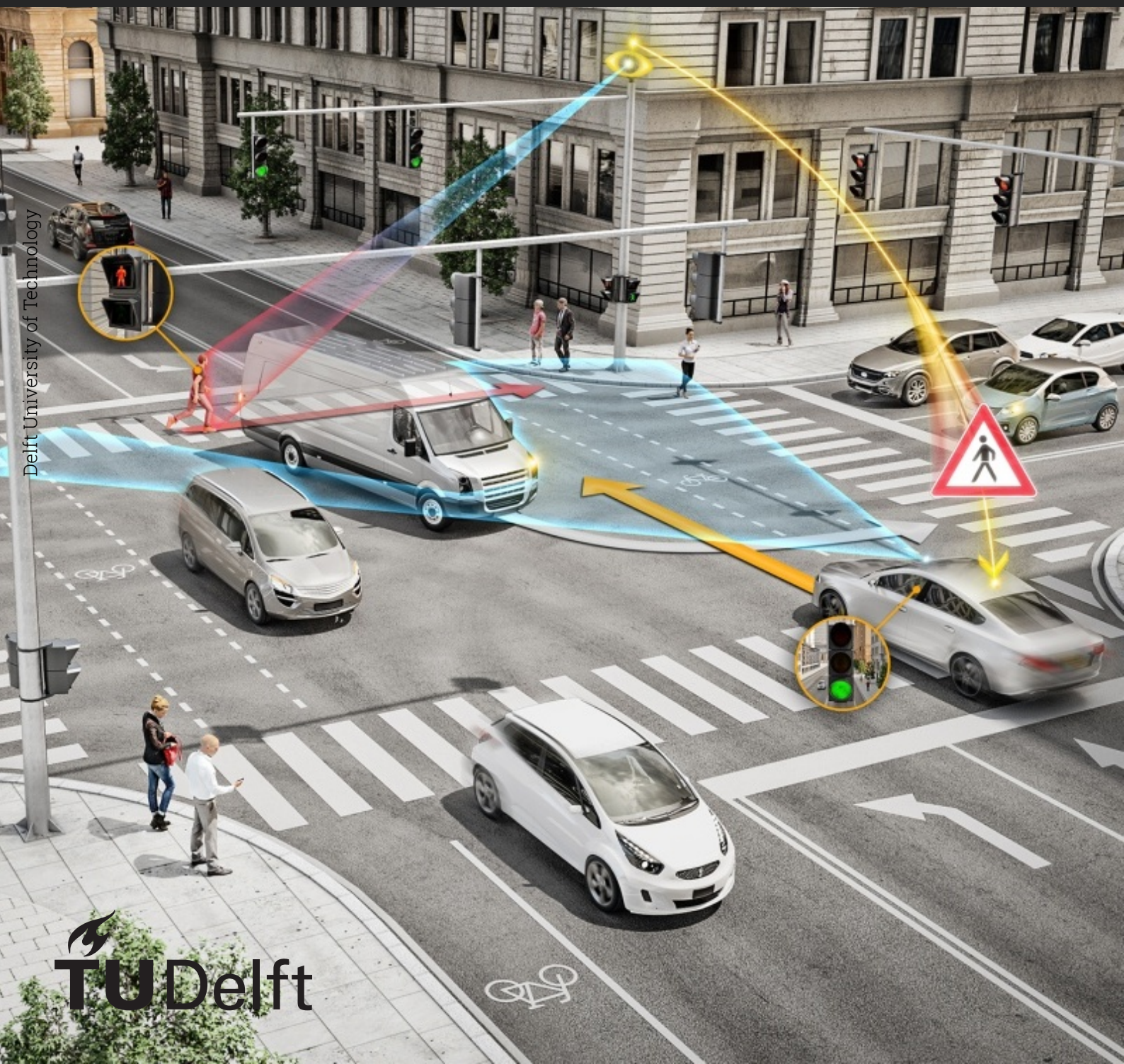
Realistic Adversarial Attacks for Robustness Evaluation of Trajectory Prediction Models

MSc Thesis Cognitive Robotics

Jeroen Hagenus



Delft University of Technology

TU Delft

Realistic Adversarial Attacks for Robustness Evaluation of Trajectory Prediction Models

by

Jeroen Hagenus

Student Name	Student Number
J. Hagenus	5099528

Primary supervisor:	Arkady Zgonnikov
Daily supervisor:	Julian Schumann, Frederik Baymler Mathiesen
Insitution:	Delft University of Technology
Project Duration:	February, 2024 - September, 2024
Faculty:	Faculty of Cognitive Robotics, Delft

Cover: Continental's V2X comms and sensor technology at an intersection

TU Delft

Realistic Adversarial Attacks for Robustness Evaluation of Trajectory Prediction Models

Jeroen Hagenus

***Abstract*—Trajectory prediction is a key element of autonomous vehicle systems, enabling them to anticipate and react to the movements of other road users. Robustness testing through adversarial methods is essential for evaluating the reliability of these prediction models. However, current approaches tend to focus solely on manipulating model inputs, which can generate unrealistic scenarios and overlook critical vulnerabilities. This limitation may result in incomplete assessments of model performance in real-world conditions. The specific effects of more comprehensive adversarial attacks on trajectory prediction models have not been thoroughly investigated. In this work, we demonstrate that by perturbing both model inputs and anticipated future states, we can uncover previously undetected weaknesses and provide a more realistic evaluation of model robustness. Our novel approach incorporates dynamical constraints and preserves tactical behaviors, enabling more effective and realistic adversarial attacks. We introduce new performance measures to assess the realism and impact of these adversarial trajectories. Testing our method on a state-of-the-art prediction model reveals significant increases in prediction errors and collision rates under adversarial conditions. Qualitative analysis further shows that our attacks can expose critical weaknesses, such as the model's inability to detect potential collisions in what appear to be safe predictions. These results underscore the need for more comprehensive adversarial testing to better evaluate and improve the reliability of trajectory prediction models for autonomous vehicles. To support further research in this area, we provide an open-source framework for studying adversarial robustness in trajectory prediction. This work advances adversarial testing techniques, contributing to the safety and reliability of autonomous driving systems.**

***Index Terms*—Trajectory prediction, adversarial attack, robustness**

I. Introduction

Trajectory prediction—the task of forecasting the future positions of traffic participants over time—is a critical component of autonomous vehicle (AV) systems, allowing them to adapt to the actions of nearby road users (Figure 1A). As AVs become more prevalent, ensuring their robustness and reliability is essential for public safety [1]. Adversarial testing has emerged as an essential tool for evaluating the robustness of trajectory prediction models, helping to uncover vulnerabilities that could lead to dangerous situations on the road [2]–[7]. The ability of a model to withstand such attacks is known as adversarial robustness [8].

Recent advances in adversarial attack generation have shown promise in exposing system vulnerabilities [2], [5], [7], [9]. Traditional methods focus on perturbing the observed states of target vehicles, creating scenarios that lead to prediction errors or simulated collisions. These approaches demonstrate two key capabilities: first, they generate dynamically feasible trajectories, allowing the attacks to transfer to real-world scenarios [5]; second, they adhere to realistic driving behaviors [2], [5]. This adherence is crucial, as significant modifications to trajectories make it difficult to determine whether prediction errors arise from model vulnerabilities or from fundamental changes in the underlying driving behavior [2]. Tactical behavior, which refers to an agent's decision-making processes like lane changes or speed adjustments, must be preserved to ensure the realism of the adversarial attack.

A key limitation of existing methods is their exclusive focus on perturbing observed states, neglecting the impact on future states. This oversight can result in unrealistic scenarios where the adversarial trajectory deviates significantly from intended behavior after the observation period (Figure 1B). For instance, an attack might cause a vehicle to swerve unnaturally or take an impossible path through an intersection. These unrealistic outcomes limit the effectiveness of adversarial testing, as they may not represent the real-world challenges AVs could encounter. The challenge lies in generating attacks that maintain tactical consistency throughout the entire trajectory, including both observed and future states.

In this paper, we present a novel method for generating adversarial attacks that accounts for both observed and future states. Our approach enhances existing techniques by introducing constraints on future states, ensuring that the entire adversarial trajectory maintains realistic driving behavior (Figure 1C). This allows us to create more subtle and potentially dangerous attack scenarios, such as slight modifications to a vehicle's path that lead to unexpected collisions while appearing safe to the prediction model (Figure 1D). By considering future states during attack generation, we ensure that adversarial trajectories remain plausible and reflective of real-world driving scenarios, leading to a more comprehensive and realistic evaluation of trajectory prediction model robustness.

The contributions of this paper can be summarized as follows:

- **Advanced adversarial attacks for trajectory prediction models**: We introduce a method that enhances the generation of dynamically feasible and tactically relevant adversarial attacks. This incorporates an attack generation strategy with improved dynamical constraints, building upon Cao et al. [2], but deriving dynamics solely from stored positions and facilitates both forward and backward agent movement. We refine tactical behavior preservation by implementing constraints that penalize deviations from the original trajectory and restrict the search space for adversarial examples. Additionally, we extend

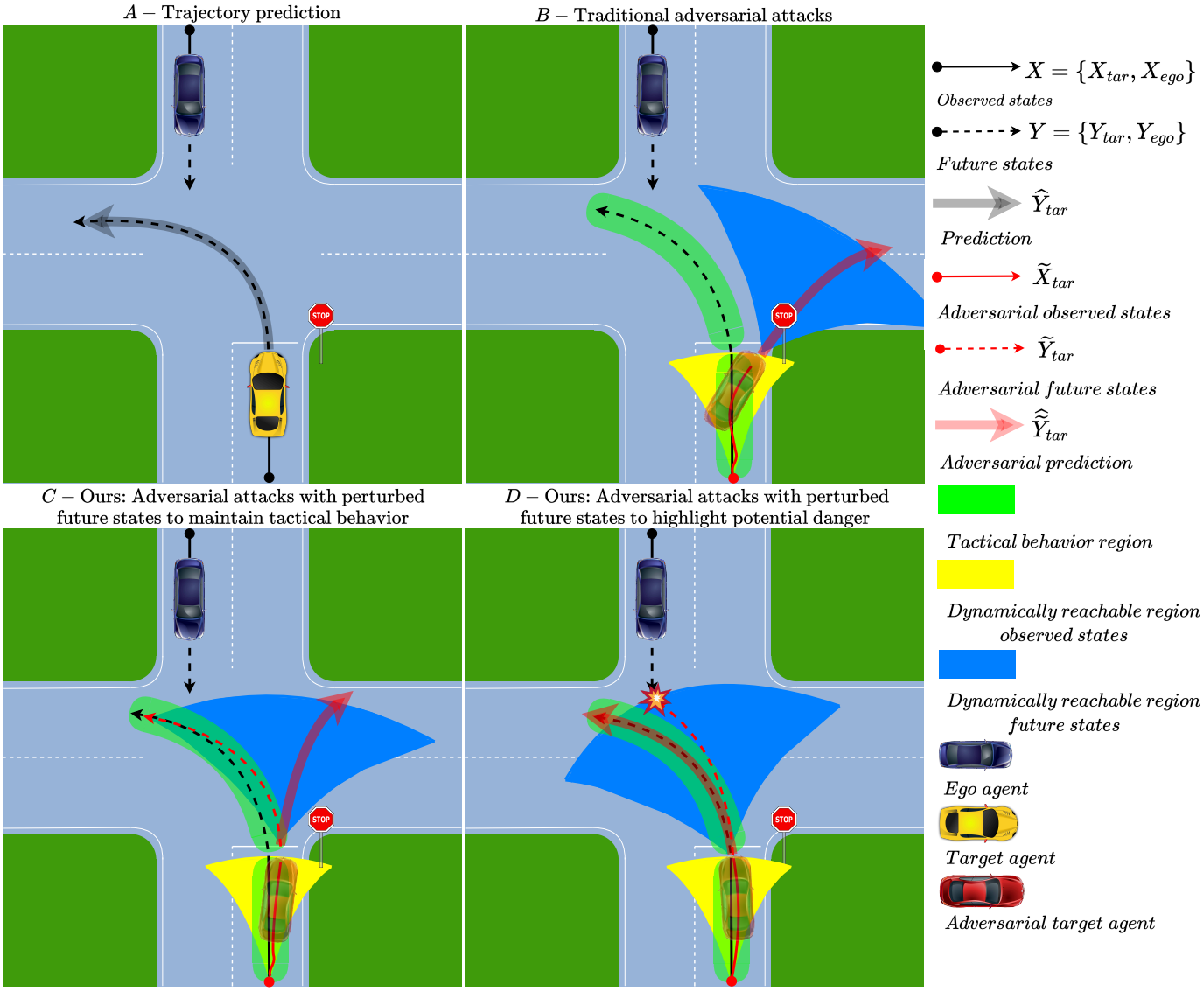


Fig. 1. Adversarial attacks on trajectory prediction: A) The ego agent (AV) correctly predicts the target agent's driving behavior as it makes a left turn at an intersection. The ground truth trajectories are split into two sections: observed states and future states, each containing the agent's positions at different time points. The prediction model's task is to accurately forecast the future states of road users based on the observed states. B) Current approaches focus on perturbing only the observed states, constraining them within dynamically feasible and tactical behavior regions. However, this method does not guarantee that future states will maintain the same tactical behavior as the ground truth. C) Our approach extends perturbations to both observed and future states. This ensures that the entire trajectory, including future states, remains within dynamically feasible regions and preserves tactical behavior. D) This approach creates scenarios where the model predicts a safe trajectory, but the perturbed future states actually lead to a collision.
Figure information: The overlapping area of the yellow/green and blue/green regions constitutes the feasible set for observed states C_{X} and future states C_{Y} of the adversarial trajectory.

adversarial attack objectives beyond existing ADE/FDE and collision attack objectives, including an enhanced collision attack concept integrated with the targeted attack method proposed by Tan et al. [7].

- **Comprehensive evaluation methodology for adversarial attacks**: We develop an approach to assess the effectiveness and realism of adversarial attacks on trajectory prediction models. This includes four novel performance measures: two quantifying maximum and average deviations from the ground truth, and two evaluating realism through required control actions (acceleration and curvature). We also establish a framework[1] for generating adversarial attacks using Projected Gradient Descent, facilitating the evaluation of diverse attack objectives and perturbation constraints.

[1] https://github.com/DAI-Lab-HERALD/General-Framework/tree/main/Framework/Perturbation_methods/Adversarial_classes

II. Related work

A. Adversarial Attack Generation for Vehicles with Dynamical Constraints

Adversarial attack generation for trajectory prediction models falls into two categories: using generative models to sample adversarial scenarios [10] and generating adversarial scenarios by perturbing real traffic scenarios [2], [5], [7], [9], [11]. The latter attack generation method can be divided into black-box attacks, which only use the model's output and do not rely on gradient information, and white-box attacks, which use the model's gradients to find perturbations.

Black-box attacks, like Particle Swarm Optimization (PSO) [5], can be slow and unreliable [2]. In contrast, white-box attacks, like Projected Gradient Descent (PGD) [2], [5], [7], [9], are generally effective but struggle with trajectory prediction models containing non-differentiable layers, as these layers prevent the calculation of the gradients necessary for the attack [5], [12]. This limitation is becoming less relevant as state-of-the-art models adopt differentiable architectures [13], [14]. In cases where access to gradients is restricted, for example due to intellectual property concerns, this challenge can be addressed by using surrogate models and generating transferable attacks [15]. Our work focuses on utilizing a white-box approach, which can be implemented by perturbing either spatial positions [5] or control actions [2]. Both approaches utilize different methods to limit the perturbation using dynamical constraints.

- **Dynamical Constraints Based on Spatial Positions**. Zhang et al. [5] perturb traffic scenarios by adjusting spatial positions while ensuring dynamic feasibility through data statistics, a method referred to as a "Search attack". Their method guarantees that perturbed trajectories stay within $\mu \pm 3\sigma$ of the mean (μ) and standard deviation (σ) of velocity, longitudinal/lateral acceleration, and their derivatives. By first obtaining adversarial perturbations to trajectory prediction model inputs, then scaling these perturbations to satisfy the data statistics conditions. Yin et al. [9] enhanced this approach by substituting the data statistics method with a continuous curvature model called an "SA-attack". Instead of using data statistics and scaling the perturbation accordingly, they employ a continuous curvature model based on a pure pursuit method. The pure pursuit method is a path-tracking algorithm that generates a trajectory by continuously pursuing a point on the path at a fixed lookahead distance from the current position [16]. Similar to the 'SA-attack' strategy, this approach first obtains adversarial perturbations and then fits the dynamical model between these perturbations to generate trajectories that ensure dynamic feasibility at specific speeds. The 'SA-attack' offers more realistic dynamical constraints for the generated trajectories compared to the data statistics method.
- **Dynamical Constraints Based on Control Actions**. Cao et al. [2] propose generating adversarial trajectories by perturbing control actions derived from the agent's spatial position, heading angle, and velocity using a kinematic bicycle model. Perturbing control actions, specifically acceleration and curvature, offer greater control in generating adversarial trajectories due to two key benefits. First, it ensures that the trajectories adhere to the physical constraints associated with a specific vehicle. Second, it ensures that the perturbed control actions remain consistent with those of the original trajectory by limiting the perturbation relative to the original control actions. These advantages together help create more realistic trajectories. The adversarial trajectories are then recovered using a similar kinematic bicycle model.

B. Constraint Adversarial Perturbations to Maintain Tactical Behaviour

Generating adversarial attacks that avoid detection as anomalies requires that they are dynamically feasible and do not deviate excessively from their ground truth, ensuring that they stay close to the intended trajectory. To address this latter aspect, two key strategies are currently used: fixed bound and constraint-encoding.

- **Fixed Bound**. This strategy restricts deviations from the ground truth positions to within a specified threshold by scaling the perturbation until this condition is met, thereby creating a fixed bound around the ground truth positions for each timestep in the trajectory [5]. While maintaining hard guarantees on constraint satisfaction, this approach transforms the problem into an unconstrained optimization task.
- **Constraint-encoding**. Cao et al. [2] introduced constraint-encoding based on a soft clipping function. Their method employs a non-linear penalty for perturbed input values. It measures the distance between the adversarial and ground truth trajectories and then applies a penalty term that becomes more severe as the changes in distances grow larger. Minimizing this distance during adversarial attack generation encourages the method to generate samples that more closely resemble the intended trajectory. However, while this approach offers improved control over perturbations, it does not impose an absolute limit on how far the perturbed trajectories can deviate from the ground truth, potentially allowing for significant deviations in some cases.

C. Attack types

Creating adversarial attacks involves specifying the attack objective for robustness evaluation. Current literature explores attacks along various measures, primarily focusing on six key aspects: Average Displacement Error (ADE)/Final Displacement Error (FDE) attack, lateral/longitudinal attack, targeted attack, and collision attack.

- **ADE/FDE Attack**. The ADE attack aims to identify perturbations causing the prediction model to produce the worst possible predictions, those deviating furthest from the ground truth [2], [5], [6], [17]. This is achieved by maximizing the difference between all predicted and

ground truth trajectory states. The FDE attack has a similar goal but focuses solely on the final state of the predicted and ground truth trajectories [5].

- **Lateral/Longitudinal Attack**. While the ADE/FDE attack does not specify a direction, deviations should be directed to the left/right for lateral attacks or forward/backward for longitudinal attacks to measure behaviors like spoofed lane changes or fake accelerations [5], [6], [17].
- **Targeted Attack**. This method forces the trajectory prediction model to make predictions as close as possible to user-specified desired trajectories, such as predictions pointing to the left or right side of the lane [7]. Unlike ADE/FDE and lateral/longitudinal attacks, this approach typically imposes stricter constraints on the perturbations, as the intended direction of the predicted trajectory is predetermined. While this method still allows for manipulation of the model's output, the range of possible outcomes is often more constrained compared to other attack methods.
- **Collision Attack**. Unlike the targeted attack, which manipulates the prediction model to output specific trajectories chosen by the attacker, the collision attack focuses on creating collisions using existing trajectories of other agents. It achieves this by minimizing the distance between the predicted trajectory of the adversarial agent and the ground truth trajectories of surrounding agents [3], [11].

III. Problem Definition

A. Trajectory Prediction

The goal of trajectory prediction is to forecast the future movements of road users, known as agents, by analyzing their observed trajectory. Each agent i has a state s_i^t at time t, which is the spatial positions $p = (x, y)$. The states of all agents at time t are defined as $S^t = (s_1^t, \ldots, s_N^t)$. This task involves capturing a series of states for each agent at fixed time intervals Δt and predicting multiple possible future trajectories.

The formal description of the trajectory prediction problem can be summarized as follows [2], [4]: Define $X^t = (x_1^t, \ldots, x_N^t)$ as the joint state of all agents at time $t \leq 0$, where N is the total number of agents. Let H be the length of the observation window, with $X = (X^{-H+1}, \ldots, X^0)$ representing the observed trajectory. Future states are represented by $Y^t = (y_1^t, \ldots, y_N^t)$, where y_i^t denotes the state of agent i at time $t > 0$. These future states are what the model aims to predict. The future trajectories span a finite horizon T, denoted as $Y = (Y^1, \ldots, Y^T)$.

Using a probabilistic model $\mathbb{P}_\theta(Y|X)$ parameterized by θ, the model generates K potential trajectories from the same initial state X. Each predicted trajectory is denoted as $\hat{Y}_k$ where $k \in \{1, \ldots, K\}$. The goal is to estimate the future trajectories as accurately as possible, where each $\hat{Y}_k \approx Y = (Y^1, \ldots, Y^T)$ (Figure 1A). The set of all predicted trajectories is represented as $\hat{Y} = \{\hat{Y}_1, \hat{Y}_2, \ldots, \hat{Y}_K\}$.

B. Adversarial Attack on Trajectory Prediction Models

This work focuses on creating adversarial scenarios for an autonomous vehicle (AV). In these scenarios, an adversarial agent, originally the target agent, attempts to deceive the prediction model employed by the ego agent (AV) into making incorrect predictions (Figure 1B). We chose this approach of defining ego and target vehicles because it closely mirrors real-world autonomous driving scenarios, where an AV (ego agent) must predict and respond to the behavior of other vehicles (target agents) in its environment. Their trajectories are defined as $X = \{X_{\text{tar}}, X_{\text{ego}}\}$ for the collection of observed states and $Y = \{Y_{\text{tar}}, Y_{\text{ego}}\}$ for the collection of future states. The attack on the prediction model is executed by modifying the observed states of the target agent, X_{tar}, using perturbations δ_{X}. In our work, we generate such perturbations δ using Projected Gradient Descent (PGD) to create adversarial observed states $\tilde{X}_{\text{tar}}$. The probabilistic model, originally trained on authentic samples, is then employed as $\mathbb{P}_\theta(\cdot|\tilde{X}_{\text{tar}})$, resulting in predictions on the adversarial trajectories denoted as $\hat{\tilde{Y}}_{\text{tar}}$. This allows us to evaluate the performance of the model against the adversarial trajectories $\tilde{X}_{\text{tar}}$. Additionally, this paper introduces perturbed future states for the target agent, defined as $\tilde{Y}_{\text{tar}}$, which are created using perturbation δ_{Y}. These perturbed future states will be utilized to create more realistic attacks.

C. Projected Gradient Descent

To generate adversarial attacks using a white-box attack strategy, we calculate perturbations δ using Projected Gradient Descent (PGD). PGD is an extension of traditional gradient descent methods for constrained optimization problems. While standard gradient descent methods minimize a function without constraints, PGD minimizes a function subject to specific constraints, making it particularly suitable for generating adversarial examples within defined limits. The PGD algorithm iteratively refines the perturbations using alternating gradient and projection steps [18]. The process begins with an initialized perturbation tensor δ^0 set to zeros. For each iteration m, where $m \in \{1, 2, ..., M_{max}\}$ and M_{max} denotes the maximum number of iterations, two key steps are performed. First, a gradient step utilizes a step size α to update the perturbation δ^{m-1}, producing an intermediate perturbation $\hat{\delta}^{m-1}$ (Equation 1). Second, a projection step finds an updated perturbation δ^m by projecting $\hat{\delta}^{m-1}$ on to the feasible set C (Equation 2).

$$\hat{\delta}^{m-1} := \delta^{m-1} - \alpha \nabla_\delta \mathcal{L}(\theta, X, Y, \delta^{m-1}) \tag{1}$$

$$\delta^m := \underset{v \in C}{\operatorname{argmin}} \left\| v - \hat{\delta}^{m-1} \right\| \tag{2}$$

The gradient used to update the perturbations δ is computed with respect to a loss function $\mathcal{L}$. This loss function incorporates several key elements: the model parameters θ, the spatial positions of the observed trajectory X, the future trajectory Y, and the perturbations δ^{m-1}.

D. Attack Function in PGD

This paper focuses on the ADE/FDE, and collision attack objectives, with loss functions determined according to the PGD format.

- **ADE/FDE Attack**. These attacks mislead the trajectory prediction model $\mathbb{P}_\theta$ by finding perturbations δ_{X} that result in the worst predictions $\hat{\tilde{Y}}_{\text{tar}}$ possible. This is achieved by increasing the mean Euclidean distance (l_2-squared) between all predicted future positions $\hat{\tilde{Y}}_{\text{tar}}$ and the ground truth future positions Y_{tar}, for the ADE attack (Equation 3), or only the final position for the FDE attack (Equation 4).

$$l_{\text{ADE}}(Y_{\text{tar}}, \hat{\tilde{Y}}_{\text{tar}}) = -\frac{1}{T}\sum_{t=1}^{T}\left\|\hat{\tilde{Y}}_{\text{tar}}^{t} - Y_{\text{tar}}^{t}\right\|_2 \tag{3}$$

$$l_{\text{FDE}}(Y_{\text{tar}}, \hat{\tilde{Y}}_{\text{tar}}) = -\left\|\hat{\tilde{Y}}_{\text{tar}}^{T} - Y_{\text{tar}}^{T}\right\|_2 \tag{4}$$

- **Collision Attack**. This attack aims to find perturbations δ_{X} that mislead the trajectory prediction model $\mathbb{P}_\theta$ into believing a potential collision will occur. This objective is achieved by minimizing the smallest Euclidean distance (l_2-squared) between the predicted future positions of the target agent $\hat{\tilde{Y}}_{\text{tar}}$ and the ground truth future positions of the ego agent Y_{ego} (Equation 5).

$$l_{\text{Col}}(Y_{\text{ego}}, \hat{\tilde{Y}}_{\text{tar}}) = \min_{t\in\{1,\ldots,T\}}\left\|\hat{\tilde{Y}}_{\text{tar}}^{t} - Y_{\text{ego}}^{t}\right\|_2 \tag{5}$$

E. Feasible Set for Perturbations

In the context of adversarial attacks on trajectory prediction models, we define additional constraints on the perturbations δ_{X} applied on the adversarial observed states $\tilde{X}_{\text{tar}}$. These constraints ensure that the adversarial observed states $\tilde{X}_{\text{tar}}$ remain dynamically feasible and within a realistic range relative to the ground truth trajectory X_{tar}. Formally, we define a constraint set C such that $\tilde{X}_{\text{tar}} = X_{\text{tar}} + \delta_{\mathrm{X}}, \quad \tilde{X}_{\text{tar}} \in C_{\mathrm{X}}$, where C_{X} represents the feasible region of the adversarial observed trajectory $\tilde{X}_{\text{tar}}$. This set is designed to constrain the perturbations, ensuring that the perturbed trajectory remains close to the intended trajectory and, crucially, that the resulting perturbed states are dynamically feasible. Dynamic feasibility in this context refers to trajectories that adhere to the physical limitations of the agent, such as maximum acceleration and turning radius constraints. However, focusing solely on a feasible region C_{X} for observed states does not guarantee that a feasible region C_{Y} exists for future states (Figure 1B). This limitation results in a loss of similarity between the predicted tactical behavior and the tactical behavior observed in the ground truth future states Y_{tar}. To address this issue, we propose a novel approach that extends the concept of feasible regions C_{Y} to future states Y_{tar} (Figure 1C), as discussed in detail in Section IV.

IV. Generating Constrained Adversarial Trajectories

A. Constraining Perturbations using Dynamical Constraints

The foundation for constraining the adversarial trajectory of the target agent using dynamic constraints in this paper is built upon the back-and-forth interplay between spatial positions and control actions. Perturbing control actions ensures the generation of dynamically feasible trajectories. Control actions, defined by acceleration a and curvature κ, are provided by user inputs through the gas/brake pedals and the steering wheel. Acceleration influences changes in the longitudinal direction of the trajectory, while curvature introduces variations in the lateral direction. Curvature is preferred over steering wheel angle because it is independent of the vehicle's internal dynamics (e.g., different wheelbases), making it easier to implement across a broader range of vehicles. At any given time t, the control action state for an agent i is represented as $u_i^t = (a_i^t, \kappa_i^t)$. Collectively, for all agents at time t, this is denoted as $U^t = (u_1^t, \ldots, u_N^t)$. For observed states, the control sequence is $U_{\mathrm{X}} = (U^{-H+1}, \ldots, U^{-1})$, and for future states, it extends to $U_{\mathrm{Y}} = (U^0, \ldots, U^{T-1})$.

1) Dynamical Model: The process of using control actions, used in the paper by Cao et al. [2], involves converting states s^t (including spatial postions p), heading angle θ^t, velocity v^t and control actions u^t to subsequent states s^{t+1}, heading angle θ^{t+1} and velocity v^{t+1} through a dynamical model Φ (Equation 6).

$$s^{t+1}, \theta^{t+1}, v^{t+1} = \Phi(s^t, u^t, \theta^t, v^t) \tag{6}$$

To derive the control actions for a given trajectory, an inverse dynamical model Φ^{-1} is utilized, which converts states s^t, s^{t+1} and s^{t+2} to control actions u^t (Equation 7).

$$u^t = \Phi^{-1}(s^{t+2}, s^{t+1}, s^t) \tag{7}$$

This bi-directional conversion is crucial for creating adversarial attacks against the trajectory prediction model, where the control actions U_{X} and U_{Y} are perturbed using perturbations $\delta_{U_{\mathrm{X}}}$ and $\delta_{U_{\mathrm{Y}}}$, resulting in the perturbed control actions $\tilde{U}_{\mathrm{X}}$ and $\tilde{U}_{\mathrm{Y}}$. Using the dynamical model Φ, these perturbed control actions are converted into adversarial trajectories $\tilde{X}$ and $\tilde{Y}$, with the important constraint that $\tilde{X}^{-H+1} = X^{-H+1}$, meaning the initial position remains unchanged. . Here, the positional changes compared to the Ground truth trajectory X and Y, are defined by the perturbations in spatial direction δ_X and δ_Y.

a) Model modifications: Our approach modifies the model proposed by Cao et al. [2] to use only spatial positions p, without assuming access to heading angle θ and velocity v information in the initial state. This modification necessitates changes to both the inverse dynamical model Φ^{-1} and the dynamical model Φ. The modified inverse dynamical model f^{-1} derives control actions u using states s^{t+1}, s^t, and s^{t-1} different compared to inverse dynamical model Φ^{-1}

and allows for both forward and backward movement of the adversarial agent (Equation 8).

$$
\begin{aligned}
u^t &= f^{-1}(s^{t+1}, s^t, s^{t-1}) \\
\begin{pmatrix} a^t \\ \kappa^t \end{pmatrix} &= \begin{pmatrix} \frac{v^{t+1}-v^t}{\Delta t} \\ \frac{\theta^{t+1}-\theta^t}{v^t \cdot \Delta t} \end{pmatrix}
\end{aligned}
$$

Where :

$$
\begin{aligned}
\theta^t &= \text{atan2}\left(c^t \cdot (p_y^t - p_y^{t-1}), c^t \cdot (p_x^t - p_x^{t-1})\right) \\
v^t &= c^t \cdot \frac{\left\| p^t - p^{t-1} \right\|_2}{\Delta t}
\end{aligned}
$$

Direction :

$$
\begin{aligned}
\hat{\theta}^t &= \text{atan2}\left(p_y^t - p_y^{t-1}, p_x^t - p_x^{t-1}\right) \\
d^t &= \begin{cases} +1 & \text{if } t \leq -H+2 \\ +1 & \text{if } |\hat{\theta}^t - \hat{\theta}^{t-1}| \leq \frac{\pi}{2} \\ -1 & \text{if } |\hat{\theta}^t - \hat{\theta}^{t-1}| > \frac{\pi}{2} \end{cases}
\end{aligned}
$$

Sign :

$$
c^t = \prod_{i=-H+1}^{t} d^i \tag{8}
$$

To calculate the control action u^t, heading angle θ^t, and velocity v^t at time t, we require the previous state s^{t-1}. This presents a challenge at the beginning of our trajectory, specifically at $t = -H+1$. At this point, we lack information about the state at $t = -H$, which is needed to compute the initial values. To address this data gap issue, we implement a solution that begins the control actions calculations at $t = -H+2$ instead of $t = -H+1$. For the initial state, we use the same heading angle θ and velocity v from time $t = -H+2$ to derive the control actions u^{-H+1} (Equation 9).

$$
\begin{pmatrix} \theta^{-H+2} \\ v^{-H+2} \end{pmatrix} = \begin{pmatrix} \theta^{-H+1} \\ v^{-H+1} \end{pmatrix} \tag{9}
$$

For the forward pass, we employ a modified dynamical model f that generates the next state s^{t+1} given the current state s^t, control actions u^t, heading angle θ^t, and velocity v^t, similar to the dynamical model Φ (Equation 10). A key distinction between our model f and Cao et al.'s model Φ lies in the indexing of the heading angle and velocity. We utilize θ^{t+1} and v^{t+1} to calculate s^{t+1}, whereas their model use θ^t and v^t. For the initial state at $t = -H+1$, we require a heading angle θ and velocity v. As these values are not directly available from our dataset, we initialize θ^{-H+1} and v^{-H+1} using the same values employed for the inverse dynamical model f^{-1} (Equation 9).

$$
\begin{aligned}
s^{t+1}, \theta^{t+1}, v^{t+1} &= f(s^t, u^t, \theta^t, v^t) \\
\begin{pmatrix} p_x^{t+1} \\ p_y^{t+1} \end{pmatrix} &= \begin{pmatrix} v^{t+1} \cdot \cos(\theta^{t+1}) \cdot \Delta t + p_x^t \\ v^{t+1} \cdot \sin(\theta^{t+1}) \cdot \Delta t + p_y^t \end{pmatrix}
\end{aligned} \tag{10}
$$

Where :

$$
\begin{aligned}
\theta^{t+1} &= v^t \cdot \kappa^t \cdot \Delta t + \theta^t \\
v^{t+1} &= a^t \cdot \Delta t + v^t
\end{aligned}
$$

2) Adversarial Trajectory Generation: Utilizing both the dynamical model f and the inverse dynamical model f^{-1}, trajectories and control actions are interchangeable, denoted by $(X, Y) \Leftrightarrow (U_\text{X}, U_\text{Y})$. Using this dynamical model, adversarial trajectories can be generated through a process involving five steps, divided into three phases: initialization, PGD, and finalization (Figure 2).

- **Initialize (1)**. The first step in creating adversarial attacks using this framework is to use the ground truth spatial positions for the target agent, X_tar and Y_tar, to initialize the first state $s_\text{tar}^{-H+1} = (\theta_\text{tar}^{-H+1}, v_\text{tar}^{-H+1}, p_\text{tar}^{-H+1})$. Then, using the inverse dynamical model f^{-1} (Equation 8), determine the control actions for all states, U_X^0 and U_Y^0. The next crucial step is to initialize the zero perturbation tensor δ_U^0, as this enables the gradient computations needed for the subsequent steps.
- **PGD (2, 3, 4)**. As described, we will use white-box attacks employing PGD to update the perturbations δ_U^{m-1}. In the second step of the perturbation framework, the perturbation tensor δ_U^{m-1} is split into two components: $\delta_{U_\text{X}}^{m-1}$ for observed states X_tar^{m-1} and $\delta_{U_\text{Y}}^{m-1}$ for future states Y_tar. These components are then used to update the control actions (Equations 11 and 12). Given the updated control actions, the updated trajectory can be recovered using the dynamical model f (Equation 10). The updated observed states for iteration m are denoted as $X_{U_\text{X}}^m$ and $Y_{U_\text{Y}}^m$, respectively. Since the perturbations are initially set to zero, this trajectory replicates the ground truth trajectory in the first iteration.

$$
U_\text{X}^m := U_\text{X}^0 + \delta_{U_\text{X}}^{m-1} \tag{11}
$$

$$
U_\text{Y}^m := U_\text{Y}^0 + \delta_{U_\text{Y}}^{m-1} \tag{12}
$$

In the third step, the updated observed states $X_{U_\text{X}}^m$ are utilized as input for the prediction model to generate predictions on these updated states, denoted as $\hat{Y}_{U_\text{X}}^m$. Given the prediction, the loss can be calculated based on the objective of the adversarial attack. Using the calculated loss, the gradients for the initialized perturbation tensor δ_U^0 can be computed. Given the gradients and the predefined step size α, the perturbations δ_U^{m-1} can be updated (Equations 1 and 2).

As the car has physical limits on deceleration/acceleration and turning, the updated control actions U^m are limited to the absolute physical limits U_min^abs and U_max^abs (Equation 13). Additionally, the control actions perturbations δ_U^m are also limited relative $\delta_{U_\text{min}}^\text{rel}$ and $\delta_{U_\text{max}}^\text{rel}$, helping to maintain the adversarial trajectory consistent with the expected behavior of the vehicle (Equation 14). The absolute physical limits for acceleration a are determined using the limits within the dataset, while the other limits are specified by the constraints provided by Wang et al. [11] (Table I). Note that the limits for both acceleration a and curvature κ differ in magnitude, which affects the convergence speed for generating adversarial attacks for both control actions. To compensate for this difference,

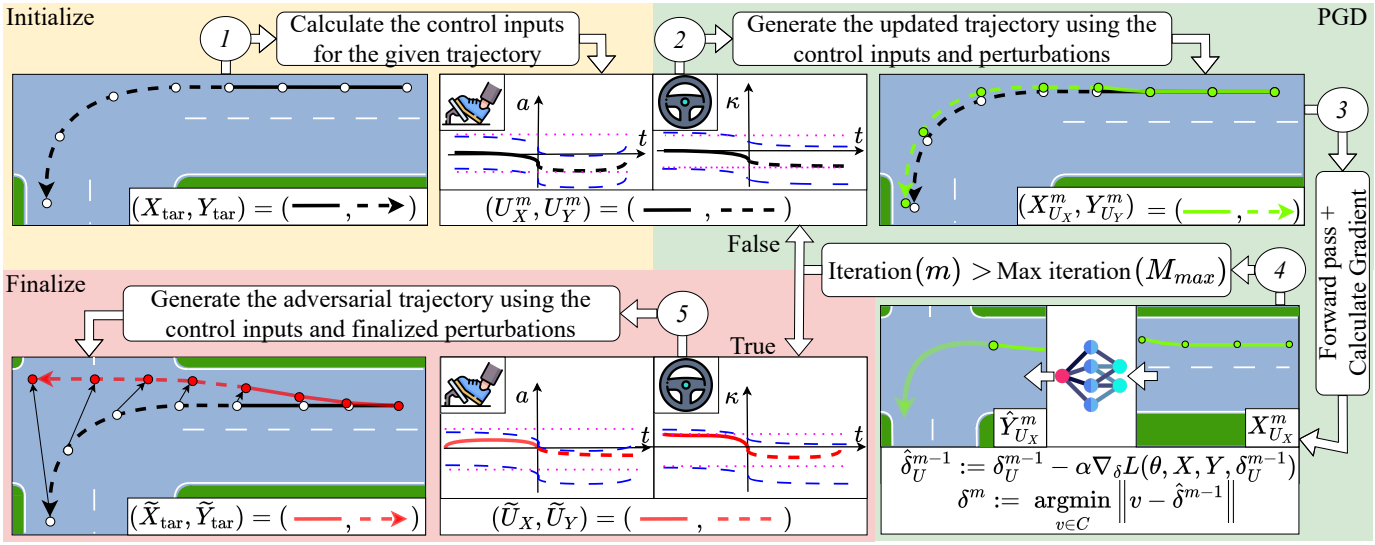


Fig. 2. Generating adversarial trajectories using control actions involves three phases: initialization (yellow), Projected Gradient Descent (green), and finalization (red). Step 1 initializes the first state s_{tar}^{-H+1} using ground truth positions and determines initial control actions (U_X^0, U_Y^0). Steps 2-4 involve PGD, iteratively refining the trajectory by updating control actions U_X^m and U_Y^m with perturbations $\delta_{U_X}^{m-1}$ and $\delta_{U_Y}^{m-1}$, subject to physical constraints (absolute limits: purple dotted line, relative limits: blue dashed line). A prediction model generates $\hat{Y}_{U_X}^m$ from updated states $X_{U_X}^m$, guiding further perturbation changes by retrieving the gradients. Steps 4-5 finalize the process, converting adversarial control actions $(\tilde{U}_X, \tilde{U}_Y)$ to adversarial trajectories $(\tilde{X}_{\text{tar}}, \tilde{Y}_{\text{tar}})$.

TABLE I
PHYSICAL CONSTRAINTS

	Relative		Absolute	
Control action	Min	Max	Min	Max
Acceleration (m/s^2)	-2	2	Data	Data
Curvature (m^{-1})	-0.05	0.05	-0.2	0.2

the step size α for acceleration a will be scaled according to the same ratio as the scaling difference of the relative limits (Table I). In the fourth step, it is checked if the maximum number of iterations is reached; if not, steps 2, 3, and 4 are repeated.

$$U_{\min}^{\text{abs}} < U^0 + \delta_U^m < U_{\max}^{\text{abs}} \tag{13}$$

$$\delta_{U\min}^{\text{rel}} < \delta_U^m < \delta_{U\max}^{\text{rel}} \tag{14}$$

- **Finalize (4, 5)**. Once the desired iteration number M_{max} is reached in the fourth step, the adversarial control actions $\tilde{U}_X$ and $\tilde{U}_Y$ are determined. The final step finishes the perturbation process by using the dynamical model f (Equation 10), to convert these adversarial control actions back into adversarial trajectories, defined as $\tilde{X}_{\text{tar}}$ and $\tilde{Y}_{\text{tar}}$.

B. Constraining Perturbations to Preserve Tactical Behavior

In the context of constraining deviations from the ground truth trajectories, we identified three conceptual approaches for applying constraints and developed four implementation strategies for generating attacks. These methods aim to balance the effectiveness and realism of the adversarial trajectories.

1) Conceptual Approaches to Constraining Adversarial Perturbations: Constraining perturbations in adversarial trajectories is essential for developing effective adversarial attacks. This technique minimizes the size of the perturbation δ_X and δ_Y, making the attack less detectable by ensuring that adversarial trajectories remain close to the ground truth trajectory, which represents human driving behavior. By preserving this proximity, constraint-encoding ensures that adversarial trajectories maintain the tactical characteristics of the original trajectory. Beyond enhancing concealment, constraint-encoding plays a critical role in identifying potential hazards in future states Y_{tar} that may be missed by the prediction model. This aspect is particularly significant because relying solely on ground truth future states Y_{tar} to assess model performance can be insufficient. A prediction model may fail to predict dynamically feasible trajectories that could lead to dangerous situations, such as collisions with the AV.

- **Perturbed Observed States**. To minimize significant deviations from the ground truth observed states, constraints can be encoded into the perturbations δ_X (Figure 1B). This function ensures that adversarial attacks remain subtle and are not flagged as anomalies by the AV systems. The perturbed inputs $\tilde{X}_{\text{tar}}$ must maintain realistic behavior. Suppose the perturbed inputs $\tilde{X}_{\text{tar}}$ cause the vehicle to exhibit erratic behaviors, such as crossing road lines or swerving off the road. In that case, it undermines the quality and credibility of the adversarial attacks. These deviations could easily be detected and dismissed as unrealistic, defeating the purpose of the attack.
- **Perturbed Future States**. To ensure that the ground truth future states Y_{tar} are reachable from the perturbed observed states $\tilde{X}_{\text{tar}}$, we use perturbed future states $\tilde{Y}_{\text{tar}}$. Using the perturbed future states $\tilde{Y}_{\text{tar}}$, we can create new

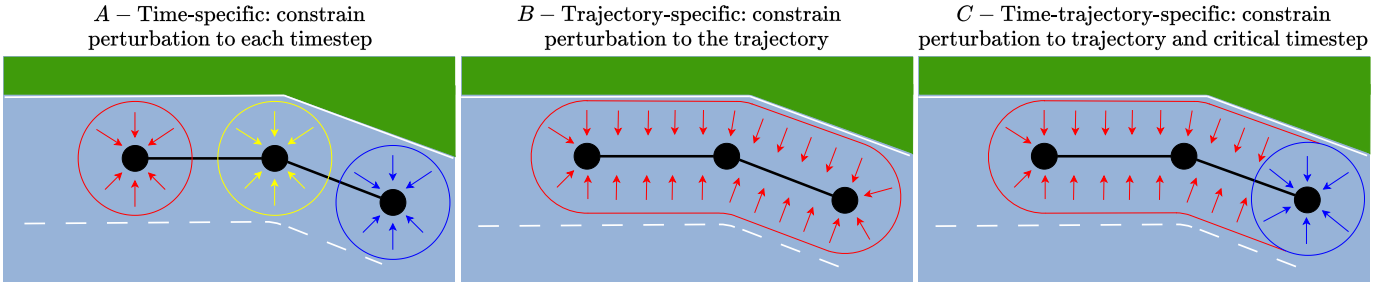


Fig. 3. Constraint to intended trajectory: A) Maintain the adversarial trajectory within an D_{max} distance (L2 norm) from the intended trajectory at each time step. B) Maintain the entire adversarial trajectory within an D_{max} distance (L2 norm) from the intended trajectory. C) Maintain the adversarial trajectory within an D_{max} distance (L2 norm) from the intended trajectory, with a specific L2 norm distance applied at a critical time step t_{crit}.
Figure information: The black line represents the ground truth trajectory. Each color corresponds to a different timestep or a combination of multiple timesteps, with the colored line indicating the distance threshold and the colored arrows showing the direction in which the penalties are applied.

trajectories that maintain the same tactical behavior as the ground truth trajectory. This approach allows for perturbations δ_X in observed states that mislead the model into making incorrect predictions, while still ensuring that maneuvers such as a left turn remain possible (Figure 1C). Unlike the method proposed by Yin et al. [9], which attempts to connect the perturbed observed states $\tilde{X}_{\text{tar}}$ directly to the ground truth future states Y_{tar}, our approach allows the model to search for adversarial trajectories within a predefined area around the ground truth future states Y_{tar} (green area in Figure 1C). These perturbed future states $\tilde{Y}_{\text{tar}}$ are initialized using the ground truth future states Y_{tar} and are enforced to stay close to them during the perturbation process. This flexibility avoids strictly adhering to the exact positions of the ground truth future states Y_{tar} and ensures that adversarial trajectories replicate realistic tactical human behavior, thereby avoiding the creation of unrealistic scenarios.

- **Predicted Future States**. One can also leverage targeted attacks introduced by Tan et al. [7], in combination with perturbed future states $\tilde{Y}_{\text{tar}}$. This method unfolds in several steps. The process begins with the prediction model generating an initial prediction $\hat{Y}_{\text{tar}}$ based on the ground truth observed states X_{tar}. This initial prediction $\hat{Y}_{\text{tar}}$ then serves as a user-specified trajectory Y_{ref}, acting as a reference point for the subsequent step. Next, the model predicts $\hat{\tilde{Y}}_{\text{tar}}$ on adversarial states $\tilde{X}_{\text{tar}}$. These predictions are constraint to remain in proximity to the reference trajectory Y_{ref} (Equation 15). Simultaneously, the future perturbed states $\tilde{Y}_{\text{tar}}$ are manipulated adversarially (Figure 1D). The goal of this manipulation is to induce collisions between the adversarial future states $\tilde{Y}_{\text{tar}}$ and the ego agent's ground truth future states Y_{ego} (Equation 16).

$$Y_{\text{ref}} = \hat{Y}_{\text{tar}}, \quad \hat{\tilde{Y}}_{\text{tar}} \approx Y_{\text{ref}} \tag{15}$$

$$l_{\text{Col}}(Y_{\text{ego}}, \tilde{Y}_{\text{tar}}) = \min_{t \in \{1,\dots,T\}} \left\| \tilde{Y}^t_{\text{tar}} - Y^t_{\text{ego}} \right\|_2 \tag{16}$$

While prediction models inherently exhibit errors even under normal conditions, constraining must account for these inaccuracies (Figure 6). To address this issue, instead of using the ground truth future states Y_{tar}, predictions based on observed states in a nominal setting $\hat{Y}_{\text{tar}}$ are used as the reference trajectory (Equation 15). This ensures that the adversarial strategy does not need to overcompensate for the model's prediction errors.

2) Implementation Approaches for Constraining Adversarial Perturbations: Penalizing perturbations for both observed δ_X and future states δ_Y is realized using four strategies in this paper: time-specific, trajectory-specific, time-trajectory-specific, and a ADE-specific approach. These strategies are designed to ensure that adversarial perturbations are controlled and that the resulting behaviors remain realistic and safe. The functions are determined according to the PGD format.

- **Time-specific**. To penalize perturbations δ_X and δ_Y, a logarithmic function is employed as a constraint-encoding, inspired by interior point methods [19] (Equation 17). This term penalizes deviations from the ground truth non-linearly and limits perturbations, δ_X and δ_Y, to a predefined threshold, D_{max} (Figure 3A). The perturbation region is circular and time-dependent, starting at time step t_0 and ending at t_1. This constraint term calculates the mean Euclidean distance (l_2-squared) between the perturbed states $\tilde{S}_{\text{tar}}$, and the ground truth states S_{tar}, where larger deviations are penalized more heavily due to the logarithmic nature of the equation.

$$l_{\text{Time}}(S_{\text{tar}}, \tilde{S}_{\text{tar}}, t_0, t_1) = -\frac{1}{t_1 - t_0 + 1} \sum_{t=t_0}^{t_1} \ln \left(D_{\text{max}} - \left\| \tilde{S}^t_{\text{tar}} - S^t_{\text{tar}} \right\|_2 \right) \tag{17}$$

- **Trajectory-specific**. A limitation of the time-specific approach is its potential to excessively penalize deviations in the longitudinal direction. This occurs because the approach penalizes deviations δ_X and δ_Y, in both lateral and longitudinal directions. Penalization in the lateral direction is essential to stay close to maintain tactical behavior, such as making a left turn (Figure 1A). However, penalization in the longitudinal direction is less critical because these deviations are primarily influenced by speed and acceleration and can naturally

vary without significantly impacting the overall trajectory. Over-penalizing in this direction can restrict the model's flexibility, preventing it from exploring a broader range of realistic adversarial scenarios.

To address this issue, the "trajectory-specific" approach defines a feasible region around the entire trajectory (Figure 3B). This method computes a distance $d(z,t)$ for each perturbed state $\tilde{S}^t_{\text{tar}}$ relative to the ground truth trajectory S_{tar}, where z represents a line segment between two consecutive trajectory points. The distance is calculated by evaluating two measures: the perpendicular Euclidean distance $d_\perp$ from the perturbed state $\tilde{S}^t_{\text{tar}}$ to the line segment formed by the consecutive ground truth states Z^z_{tar} and Z^{z+1}_{tar}, and the minimum Euclidean distance between the perturbed state $\tilde{S}^t_{\text{tar}}$ and the individual ground truth states S^z_{tar} and S^{z+1}_{tar}. The choice between these distances is determined by the relative position of the perturbed state along the line segment, as indicated by $r(z,t)$ (Figure 4). The minimum distance across all segments of the trajectory is then incorporated into a logarithmic barrier function (Equation 18). Similar to the previous approach, the logarithmic function imposes greater penalties on states that deviate further from the original trajectory.

$$l_{\text{Traj}}(S_{\text{tar}}, \tilde{S}_{\text{tar}}, t_0, t_1) = -\frac{1}{t_1 - t_0 + 1} \sum_{t=t_0}^{t_1} \ln \left(D_{\text{Max}} - \min_{z \in \{-H+1, \dots, T-1\}} d(z,t) \right) \tag{18}$$

$$d(z,t) = \begin{cases} d_\perp(\tilde{S}^t_{\text{tar}}, Z^z_{\text{tar}}, Z^{z+1}_{\text{tar}}) & \text{if } 0 < r(z,t) < 1, \\ \min(\|\tilde{S}^t_{\text{tar}} - Z^z_{\text{tar}}\|_2, & \\ \quad \|\tilde{S}^t_{\text{tar}} - Z^{z+1}_{\text{tar}}\|_2) & \text{otherwise.} \end{cases}$$

$$d_\perp(\tilde{S}^t_{\text{tar}}, Z^z_{\text{tar}}, Z^{z+1}_{\text{tar}}) = \left| \frac{D^1_x D^2_y - D^1_y D^2_x}{\|D^1\|_2} \right|$$

$$r(z,t) = \frac{D^1_x D^2_x + D^1_y D^2_y}{\|D^1\|_2^2}$$

$$\text{with} \quad D^1 = Z^{z+1}_{\text{tar}} - Z^z_{\text{tar}}, \quad D^2 = \tilde{S}^t_{\text{tar}} - Z^z_{\text{tar}}.$$

- **Time-trajectory-specific**. The trajectory-specific approach allows for more flexibility in longitudinal deviations, thereby enabling the exploration of a broader range of realistic adversarial scenarios. While this flexibility can be advantageous, it also poses a risk: perturbations may lead to substantial deviations from the ground truth states at critical moments. This can result in unrealistic or unsafe behaviors that do not align with the natural flow of the trajectory. For example, when making a left turn (Figure 1A), it is crucial that, at prediction time $t = 0$, the target agent can still feasibly complete the turn without risking a collision with the ego agent.

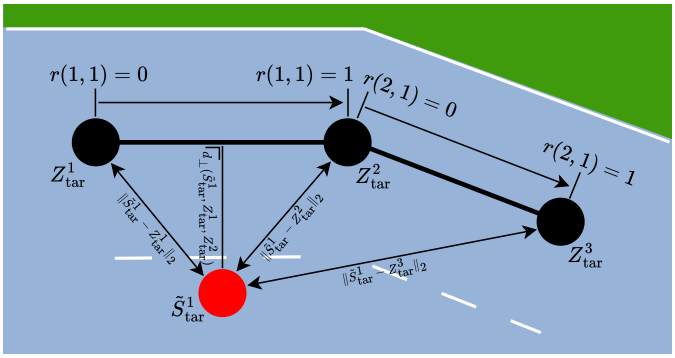


Fig. 4. The visualization of the algorithm for constraining adversarial perturbations using the trajectory-specific approach (Equation 18).

To address this challenge, we combined the time-specific with the trajectory-specific constraint approach (Equation 19). By integrating trajectory-specific constraints with time-specific constraints at critical moments t_{crit}, we maintain the coherence of the trajectory while allowing for the exploration of diverse adversarial scenarios (Figure 3C).

$$l_{\text{Time-traj}}(S_{\text{tar}}, \tilde{S}_{\text{tar}}, t_0, t_1, t_{crit}) = l_{\text{Time}}(S_{\text{tar}}, \tilde{S}_{\text{tar}}, t_{crit}, t_{crit}) + l_{\text{Traj}}(S_{\text{tar}}, \tilde{S}_{\text{tar}}, t_0, t_1) \tag{19}$$

- **ADE-specific**. Since the predicted future trajectory $\hat{Y}_{\text{tar}}$ from the prediction model is probabilistic, it becomes more challenging to use a predefined search area for constraining the predicted future states $\hat{\tilde{Y}}_{\text{tar}}$. Without control over the directions of the predictions, the adversarial model can become unstable if the predictions fall outside this predefined area.

  To address this, instead of using a logarithmic function with a fixed bound for predictions, the ADE measure will be used to constrain the perturbations δ_{X} and δ_{Y} (Equation 20). This approach minimizes the distance between the predicted future states in the nominal setting, $\hat{Y}_{\text{tar}}$, and the predictions in the adversarial setting, $\hat{\tilde{Y}}_{\text{tar}}$. One disadvantage of this approach is that there is no hard bound to constrain the predictions $\hat{\tilde{Y}}_{\text{tar}}$, which might lead to larger deviations than desired.

$$l_{\text{ADE}_{\text{tac}}}(\hat{Y}_{\text{tar}}, \hat{\tilde{Y}}_{\text{tar}}) = \frac{1}{T} \sum_{t=1}^{T} \left\| \hat{\tilde{Y}}^t_{\text{tar}} - \hat{Y}^t_{\text{tar}} \right\|_2 \tag{20}$$

C. Adversarial attack objectives

As we focused on the ADE/FDE and collision attack we will use the PGD formatted objective functions as baseline (Equations 3, 4 and 5). Building on these functions, along with the dynamical constraints l_{dyn} and tactical behavior constraints l_{tac} (Section IV), we formulate both traditional and newly designed attack functions, including feasible ADE/FDE attacks and false positive/negative collision attacks.

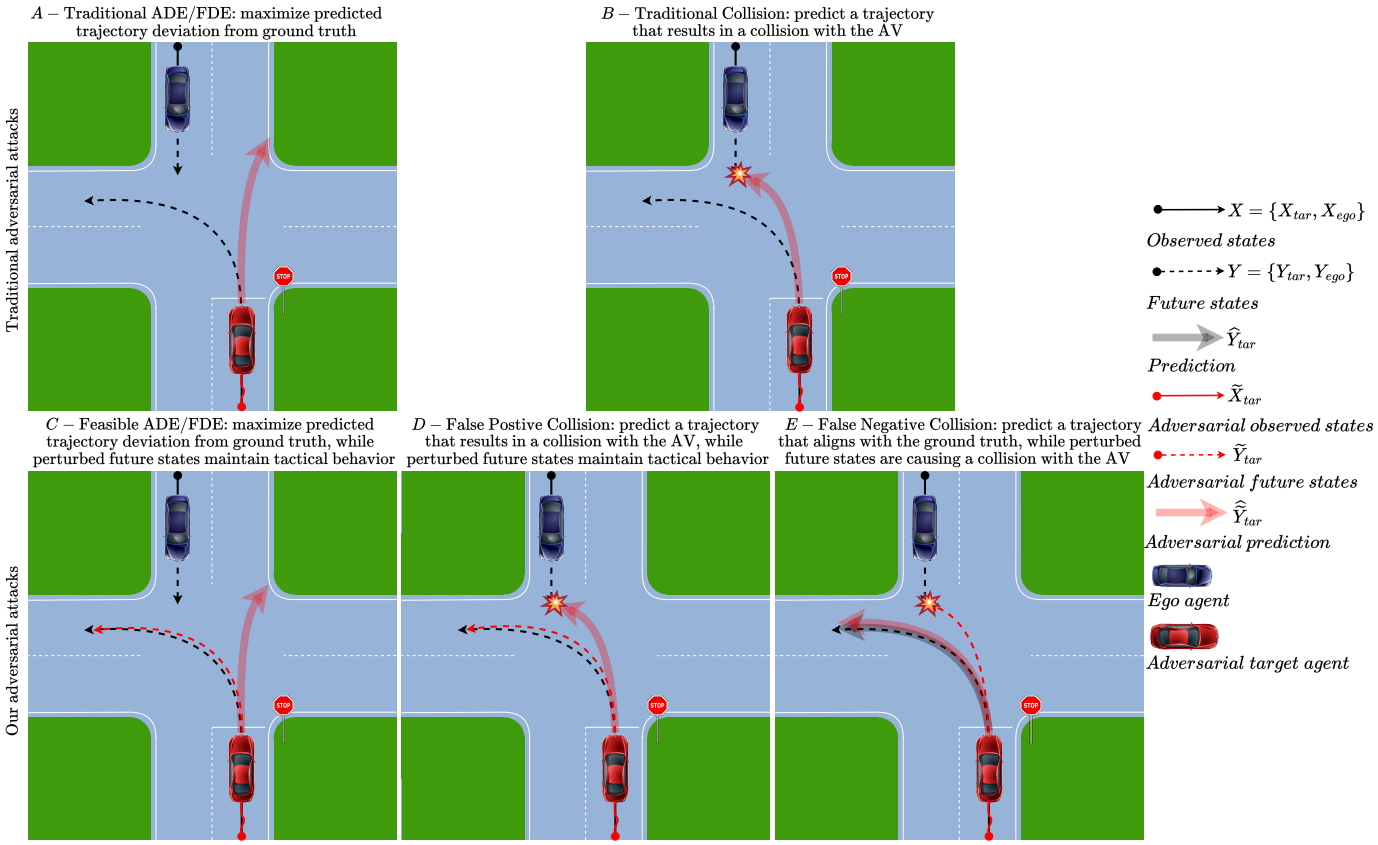


Fig. 5. Visual comparison between the traditional adversarial attacks and our adversarial attacks.

1) ADE/FDE: The foundation for ADE/FDE attacks is their respective loss functions (Equations 3 and 4). Using these, both traditional and newly designed feasible ADE/FDE attacks are developed.

- **Traditional ADE/FDE**. The tradictional approach combines ADE/FDE loss functions with dynamical and tactical behavior constraints applied to the perturbed observed states $\tilde{X}_{\text{tar}}$ (Equations 21 and 22). This method seeks to find perturbations δ_X that cause the ego agent (AV) to predict a future trajectory $\hat{\tilde{Y}}_{\text{tar}}$ for the adversarial agent that deviates as far as possible from the intended trajectory Y_{tar} (Figure 5A). These perturbations are constrained within the feasible set C_{X} for the observed states $\tilde{X}_{\text{tar}}$ (Figure 1B).

$$\mathcal{L}_{\text{ADE}} = l_{\text{ADE}}(Y_{\text{tar}}, \hat{\tilde{Y}}_{\text{tar}}) + l_{\text{dyn}}(\tilde{X}_{\text{tar}}) + l_{\text{tac}}(\tilde{X}_{\text{tar}}) \quad (21)$$

$$\mathcal{L}_{\text{FDE}} = l_{\text{FDE}}(Y_{\text{tar}}, \hat{\tilde{Y}}_{\text{tar}}) + l_{\text{dyn}}(\tilde{X}_{\text{tar}}) + l_{\text{tac}}(\tilde{X}_{\text{tar}}) \quad (22)$$

- **Feasible ADE/FDE**. Unlike the traditional approach, the "feasible ADE/FDE" attack objective imposes dynamical and tactical behavior constraints on both the perturbed observed states $\tilde{X}_{\text{tar}}$ and future states $\tilde{Y}_{\text{tar}}$ of the trajectory (Equations 23 and 24). This approach aims to find perturbations δ_X and δ_Y that cause the ego agent (AV) to predict a future trajectory $\hat{\tilde{Y}}_{\text{tar}}$ for the adversarial agent that deviates significantly from the intended trajectory Y_{tar}, while perturbed future states $\tilde{Y}_{\text{tar}}$ remain close to the ground truth future states Y_{tar} (Figure 5C). The perturbations must satisfy the constraints defined by the feasible sets C_{X} for both observed states $\tilde{X}_{\text{tar}}$ and C_{Y} for future states $\tilde{Y}_{\text{tar}}$ (Figure 1C).

$$\mathcal{L}_{\text{F-ADE}} = l_{\text{ADE}}(Y_{\text{tar}}, \hat{\tilde{Y}}_{\text{tar}}) + l_{\text{dyn}}(\tilde{X}_{\text{tar}}, \tilde{Y}_{\text{tar}}) + l_{\text{tac}}(\tilde{X}_{\text{tar}}, \tilde{Y}_{\text{tar}}) \quad (23)$$

$$\mathcal{L}_{\text{F-FDE}} = l_{\text{FDE}}(Y_{\text{tar}}, \hat{\tilde{Y}}_{\text{tar}}) + l_{\text{dyn}}(\tilde{X}_{\text{tar}}, \tilde{Y}_{\text{tar}}) + l_{\text{tac}}(\tilde{X}_{\text{tar}}, \tilde{Y}_{\text{tar}}) \quad (24)$$

2) Collision: The collision attack objective is based on two attack functions (Equations 5 and 16). Both aim to induce collisions between the ego agent's future states Y_{ego} and the target agent's states: the first function targets predicted future states $\hat{\tilde{Y}}_{\text{tar}}$, while the second targets perturbed future states $\tilde{Y}_{\text{tar}}$.

- **Tradictional Collision**. The traditional approach uses the first function and constrains the perturbed observed states $\tilde{X}_{\text{tar}}$ with dynamical and tactical behavior constraints. It seeks perturbations δ_X that cause the ego agent (AV) to predict a trajectory $\hat{\tilde{Y}}_{\text{tar}}$ resulting in a collision with itself (Figure 5B). While ensuring that the perturbed observed states $\tilde{X}_{\text{tar}}$ remain within the feasible set C_{X} (Figure 1B).

$$\mathcal{L}_{\text{col}} = l_{\text{Col}}(Y_{\text{ego}}, \hat{\tilde{Y}}_{\text{tar}}) + l_{\text{dyn}}(\tilde{X}_{\text{tar}}) + l_{\text{tac}}(\tilde{X}_{\text{tar}}) \quad (25)$$

- **False Positive Collision**. The "False Positive Collision" attack, similar to the Feasible ADE/FDE objective, adds dynamical and tactical behavior constraints to the perturbed future states $\tilde{Y}_{\text{tar}}$. It aims to deceive the ego agent by finding perturbations δ_X that make the ego agent (AV) predict a collision course with itself. However, the perturbations δ_Y keep the perturbed future states $\tilde{Y}_{\text{tar}}$ close to the ground truth Y_{tar}, preventing the actual collision from occurring (Figure 5D). This approach adheres to the constraints defined by two feasible sets: C_{X} for the perturbed observed states $\tilde{X}_{\text{tar}}$, and C_{Y} for the future states $\tilde{Y}_{\text{tar}}$ (Figure 1C).

$$\mathcal{L}_{\text{col}} = l_{\text{Col}}(Y_{\text{ego}}, \hat{\tilde{Y}}_{\text{tar}}) + l_{\text{dyn}}(\tilde{X}_{\text{tar}}, \tilde{Y}_{\text{tar}}) + l_{\text{tac}}(\tilde{X}_{\text{tar}}, \tilde{Y}_{\text{tar}}) \tag{26}$$

- **False Negative Collision**. The "False Negative collision" attack differs from both the traditional and false positive collision approaches. It dynamically constrains the perturbed observed $\tilde{X}_{\text{tar}}$ and future states $\tilde{Y}_{\text{tar}}$, while uniquely constraining the predicted future states $\hat{\tilde{Y}}_{\text{tar}}$ for tactical behavior. This attack aims to deceive the ego agent (AV) by finding perturbations δ_X that cause the ego agent to predict $\hat{\tilde{Y}}_{\text{tar}}$ the correct trajectory Y_{ref}, while simultaneously causing an actual collision with the ego agent in the perturbed future states $\tilde{Y}_{\text{tar}}$ (Figure 5E). This approach adheres to the feasible set C_{X} for perturbed observed states $\tilde{X}_{\text{tar}}$ while ensuring a collision occurs within the dynamically feasible set C_{Y} for the perturbed future states $\tilde{Y}_{\text{tar}}$, with the predicted future states $\hat{\tilde{Y}}_{\text{tar}}$ remaining close to reference trajectory Y_{ref} (Figure 1D).

$$\mathcal{L}_{\text{col}} = l_{\text{Col}}(Y_{\text{ego}}, \tilde{Y}_{\text{tar}}) + l_{\text{dyn}}(\tilde{X}_{\text{tar}}, \tilde{Y}_{\text{tar}}) + l_{\text{tac}}(\tilde{X}_{\text{tar}}, \hat{\tilde{Y}}_{\text{tar}}) \tag{27}$$

V. Experimental Setting

A. Dataset

The data for evaluation was sourced from the L-GAP dataset [20] (Figure 1A), collected from participants who manually controlled a target agent (yellow) to stop in front of the stop board and then execute left turns at an intersection. The autonomous vehicle, referred to as the ego agent (blue), maintains a constant velocity. In these specific left-turn scenarios, variables such as the distance of the ego agent from the intersection (90, 120, or 150 meters) and the time-to-arrival intervals (4, 5, or 6 seconds) were randomly assigned [20]. The target agent (yellow) was specifically selected to generate adversarial attacks. This choice is strategic because the ego agent (blue) follows a predetermined, computer-generated path at a fixed velocity and lacks a "true" ground truth. This characteristic makes the blue car an ineffective target for perturbation. Targeting the human-driven vehicle is logical for testing the trajectory prediction model, as the autonomous vehicle (blue) must accurately anticipate human actions in traffic scenarios.

The L-GAP dataset introduces a significant bias because the target agent either makes a left turn or waits at the crossing. To address this issue, the prediction model was trained using the L-GAP and NuScenes dataset, which offers a greater variety of scenarios [21]. For this training, the duration of the observation period was set to $H = 12$, and the prediction horizon to $T = 12$, with a time interval of $\Delta t = 0.1s$. These parameters were constrained by the sampling length of the dataset. To evaluate the L-GAP dataset, originally captured within the Carla simulation environment, modifications were made to improve data quality. Specifically, a Savitzky-Golay filter was employed to smooth the trajectories and mitigate inherent noise in tracking the Cartesian coordinates inside the simulator. Lastly, since this dataset captures the specific behavior of gap acceptance, we utilize this information to select samples with a higher likelihood of potential vehicle interactions. Our method for determining prediction times t_{crit} allows us to focus on moments when vehicles are more likely to be in critical decision-making situations regarding gap acceptance.

B. Prediction Model Settings

For training, generating attacks, and evaluation in both nominal and adversarial settings, we will utilize the state-of-the-art trajectory prediction model, Trajectron++ [13]. During training, this model generates $K = 100$ predictions based on the observed states X_{tar} and optional map information. For the generation of attacks, the model similarly uses $K = 100$ predictions. To determine the loss regarding the attacks, we average the K predicted trajectories, resulting in the predicted future trajectory $\hat{\tilde{Y}}_{\text{tar}}$ given the adversarial observed states $\tilde{X}_{\text{tar}}$.

C. Adversarial Attack Settings

For this research, we will compare three different dynamical models. The first is the position-based method, which uses no dynamics and perturbs the spatial positions directly. The second is the data statistics approach from Zhang et al. [5], which also perturbs spatial positions but limits the perturbation using data statistics. The third is the dynamical model proposed in this paper, inspired by Cao et al. [2], which uses control actions to perturb the spatial positions. Each type of dynamical model in our study requires the adjustment of multiple hyperparameters to fine-tune the attack strategy. To ensure a fair and consistent comparison across different attack methodologies, we first establish a fixed distance threshold for maintaining tactical behavior. We set this threshold, D_{Max}, to 0.9 meter for both adversarial observed states $\tilde{X}_{\text{tar}}$ and future states $\tilde{Y}_{\text{tar}}$. This distance threshold is selected because the lane width in the L-GAP dataset is 3.5 meters, and the width of a standard car is 1.7 meters. Therefore, a car can deviate a maximum of 0.9 meters from the center to the left or right without driving in the opposite direction or going off-road. Given that our critical time step t_{crit} for prediction is at $t = 0$, we apply time-specific and time-and-trajectory-specific constraints to the perturbed observed states $\tilde{X}_{\text{tar}}$. To constrain the tactical behavior of perturbed future states $\tilde{Y}_{\text{tar}}$, we employ trajectory-specific and ADE_{tac}-specific constraints. We use trajectory-specific constraint because future positions lack a critical time step t_{crit}. The ADE_{tac}-specific constraint

is chosen due to the prediction model's probabilistic nature. Other crucial hyperparameters, such as the step size α are set to 0.01, and the exponential decay γ is set to 0.99. To ensure that the adversarial trajectories remain within the feasible region C_{X} and C_{Y}, we implement an adaptive step size α mechanism. If the updated observed and future states fall outside the tactical behavior constraint region (Figure 1), we repeatedly scale the step size α by 0.5 until the condition is met. The maximum number of iterations M_{max} for generating adversarial attacks is set to 100.

D. Measures

For a comprehensive evaluation, we employ seven performance measures: three widely used in trajectory prediction literature and four novel measures we introduce to assess the characteristics of adversarial trajectories. The three established measures are Average Displacement Error (ADE) [22], Final Displacement Error (FDE) [23], and Collision Rate [3]. Our four new measures specifically evaluate the adversarial observed trajectories $\tilde{X}_{\text{tar}}$: peak deviation D_{peak}, mean deviation D_{mean}, acceleration α, and curvature κ. D_{peak} and D_{mean} quantify the extent of perturbations from the original trajectory X_{tar}, while acceleration and curvature provide insights into the physical feasibility of the adversarial trajectories. All measures are computed across N samples, with each sample n representing a unique scenario in our dataset.

- **ADE**: Measures the average Euclidean distance between predicted and ground truth trajectories across all time steps, predictions, and samples:

$$\text{ADE} = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{T}\sum_{t=1}^{T}\|\hat{\tilde{Y}}^{t}_{\text{tar},n,k} - Y^{t}_{\text{tar},n}\|_2 \quad (28)$$

- **FDE**: Calculates the average Euclidean distance between predicted and ground truth trajectories at the final prediction time step across all predictions and samples:

$$\text{FDE} = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{K}\sum_{k=1}^{K}\|\hat{\tilde{Y}}^{T}_{\text{tar},n,k} - Y^{T}_{\text{tar},n}\|_2 \quad (29)$$

- **Collision Rate**: Assesses the frequency of collisions in two scenarios: a) For predicted trajectories, across all samples and predictions:

$$\text{CR}_{\text{pred}} = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{K}\sum_{k=1}^{K}1\{\text{BB}(\hat{\tilde{Y}}_{\text{tar},n,k})\cap\text{BB}(Y_{\text{ego},n}) \neq \emptyset\} \quad (30)$$

  b) For the "False Negative Collision" attack, across all samples, and perturbed future states:

$$\text{CR}_{\text{FNC}} = \frac{1}{N}\sum_{n=1}^{N}1\{\text{BB}(\tilde{Y}_{\text{tar},n})\cap\text{BB}(Y_{\text{ego},n}) \neq \emptyset\} \quad (31)$$

  where $\text{BB}(\cdot)$ represents the bounding box of a vehicle, and $1\{\cdot\}$ is the indicator function that equals 1 if a collision occurs and 0 otherwise.

- **Peak deviation**: Identifies the largest deviation of the adversarial observed states from the ground truth:

$$D_{\text{peak}} = \frac{1}{N}\sum_{n=1}^{N}\max_{\{t\in-H+1,\ldots,0\}}\|\tilde{X}^{t}_{\text{tar},n} - X^{t}_{\text{tar},n}\|_2 \quad (32)$$

- **Mean deviation**: Calculates the average deviation of the adversarial observed states from the ground truth:

$$D_{\text{mean}} = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{H}\sum_{t=-H+1}^{0}\|\tilde{X}^{t}_{\text{tar},n} - X^{t}_{\text{tar},n}\|_2 \quad (33)$$

- **Acceleration**: Quantifies the acceleration input required to achieve the adversarial observed states:

$$\alpha = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{H-1}\sum_{t=-H+1}^{-1}|a^{t}_{n}| \quad (34)$$

  where a^{t}_{n} is the acceleration derived from the inverse dynamical model f^{-1} for sample n at time step t.

- **Curvature**: Determines the curvature input needed to achieve the adversarial observed states:

$$\kappa = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{H-1}\sum_{t=-H+1}^{-1}|\kappa^{t}_{n}| \quad (35)$$

  where κ^{t}_{n} is the curvature derived from the inverse dynamical model f^{-1} for sample n at time step t.

E. Framework

To create the adversarial attacks, we have developed an open-sourced framework, accessible at our GitHub repository[2]. This framework is designed to facilitate the validation of adversarial attack objectives and enables the exploration of various combinations of dynamical and tactical behavior constraints. The framework is also available for future research, allowing new attack objectives, tactical behavior and dynamical constraints to be added. Additionally, we have integrated this framework into an existing trajectory prediction framework, which supports multiple trajectory prediction models, splitting methods, datasets, and evaluation metrics, enabling versatile evaluation of trajectory prediction tasks.

VI. Results

To measure the performance of the attacks, we evaluated the three traditional attack strategies and the four new attack strategies (Section IV-C). For the evaluation, we used both quantitative and qualitative results.

A. Quantitative Results

To comprehensively evaluate these results, we applied the attacks across the entire dataset. Our analysis compared different dynamical and tactical behavior constraints, as well as their performance against the nominal (unperturbed) prediction scenario. Our analysis revealed several key findings (Table II). To highlight specific comparisons, we have used color coding

[2]https://github.com/DAI-Lab-HERALD/General-Framework/tree/main/Framework/Perturbation_methods/Adversarial_classes

TABLE II
ATTACK PERFORMANCE

Attack	Dynamical constraint		Tactical behavior constraint		Measure							
	Observed	Future	Observed	Future	ADE	FDE	CR_{pred}	CR_{FNC}	D_{peak}	D_{mean}	α	κ
Unperturbed	✗	✗	✗	✗	0.115	0.283	0.015	-	-	-	0.510	0.006
Traditional ADE	Positions	✗	Time	✗	4.045	3.447	0.007	-	0.770	0.195	53.407	20.860
	Positions	✗	Time-traj	✗	3.447	6.203	0.011	-	0.681	0.168	47.066	25.222
	Search	✗	Time	✗	3.073	5.521	0.007	-	0.714	0.177	49.392	20.100
	Search	✗	Time-traj	✗	2.028	3.668	0.010	-	0.539	0.131	38.738	27.571
	Control actions	✗	Time	✗	1.030	1.604	0.014	-	0.841	0.525	1.658	0.006
	Control actions	✗	Time-traj	✗	0.738	1.201	0.017	-	0.525	0.153	1.402	0.005
Traditional FDE	Positions	✗	Time	✗	3.983	7.158	0.008	-	0.766	0.186	51.868	23.368
	Positions	✗	Time-traj	✗	4.035	7.262	0.008	-	0.846	0.197	56.280	22.338
	Search	✗	Time	✗	2.283	4.124	0.007	-	0.594	0.148	43.665	23.848
	Search	✗	Time-traj	✗	2.306	4.165	0.007	-	0.623	0.152	45.145	23.805
	Control actions	✗	Time	✗	1.059	1.651	0.015	-	0.856	0.267	1.792	0.007
	Control actions	✗	Time-traj	✗	0.841	1.360	0.017	-	0.598	0.161	1.644	0.006
Traditional Collision	Positions	✗	Time	✗	1.371	2.495	0.020	-	0.467	0.119	36.695	27.427
	Positions	✗	Time-traj	✗	1.389	2.530	0.020	-	0.495	0.125	39.004	29.138
	Search	✗	Time	✗	1.792	3.241	0.014	-	0.514	0.123	36.494	27.938
	Search	✗	Time-traj	✗	1.819	3.290	0.015	-	0.548	0.130	38.991	25.836
	Control actions	✗	Time	✗	0.446	0.715	0.021	-	0.360	0.127	1.065	0.006
	Control actions	✗	Time-traj	✗	0.332	0.570	0.021	-	0.217	0.067	1.005	0.005
Feasible ADE	Positions	✓	Time	Trajectory	4.074	7.313	0.009	-	0.770	0.198	54.229	21.966
	Positions	✓	Time-traj	Trajectory	5.138	9.282	0.009	-	1.195	0.232	66.763	21.088
	Search	✓	Time	Trajectory	2.861	5.147	0.008	-	0.683	0.170	47.764	20.173
	Search	✓	Time-traj	Trajectory	2.866	5.154	0.009	-	0.722	0.179	51.107	17.658
	Control actions	✓	Time	Trajectory	0.250	0.470	0.021	-	0.150	0.054	0.901	0.005
	Control actions	✓	Time-traj	Trajectory	0.222	0.435	0.021	-	0.113	0.038	0.878	0.005
Feasible FDE	Positions	✓	Time	Trajectory	3.949	7.091	0.007	-	0.761	0.194	53.394	20.200
	Positions	✓	Time-traj	Trajectory	5.185	9.363	0.007	-	1.205	0.237	67.594	20.478
	Search	✓	Time	Trajectory	3.090	5.559	0.009	-	0.708	0.179	49.681	20.771
	Search	✓	Time-traj	Trajectory	3.152	5.673	0.008	-	0.758	0.190	53.582	19.544
	Control actions	✓	Time	Trajectory	0.581	0.943	0.021	-	0.459	0.159	1.112	0.006
	Control actions	✓	Time-traj	Trajectory	0.503	0.845	0.021	-	0.349	0.108	1.113	0.005
False Positive Collision	Positions	✓	Time	Trajectory	1.470	2.661	0.020	-	0.524	0.141	41.768	21.354
	Positions	✓	Time-traj	Trajectory	1.508	2.740	0.021	-	0.562	0.147	44.317	20.614
	Search	✓	Time	Trajectory	2.564	4.603	0.013	-	0.652	0.160	43.792	22.793
	Search	✓	Time-traj	Trajectory	2.604	4.678	0.012	-	0.692	0.171	47.615	21.905
	Control actions	✓	Time	Trajectory	0.295	0.504	0.020	-	0.214	0.077	0.915	0.006
	Control actions	✓	Time-traj	Trajectory	0.255	0.459	0.020	-	0.152	0.049	0.905	0.006
False Negative Collision	Positions	✓	Time	ADE_{tac}	0.562	1.020	0.019	0.736	0.423	0.116	36.427	25.177
	Positions	✓	Time-traj	ADE_{tac}	0.598	1.085	0.020	0.741	0.467	0.126	40.322	25.739
	Search	✓	Time	ADE_{tac}	0.878	1.586	0.020	0.709	0.452	0.122	38.110	24.988
	Search	✓	Time-traj	ADE_{tac}	0.933	1.685	0.019	0.709	0.481	0.130	40.906	24.469
	Control actions	✓	Time	ADE_{tac}	0.123	0.222	0.018	0.723	0.074	0.028	0.845	0.006
	Control actions	✓	Time-traj	ADE_{tac}	0.105	0.198	0.018	0.723	0.052	0.021	0.855	0.006

in the table. If two colors are separated by a slash (/), it indicates an overlapping cell box of a specific measure and attack type. If they are separated by the word 'and', it represents a comparison between measures. In the following section, we first focus on each pair of related attacks: Traditional ADE and Feasible ADE (pink), Traditional FDE and Feasible FDE (white), and Traditional Collision and False Positive Collision (lightblue).

1) Dynamical Constraints: When comparing the performance under adversarial attacks using control actions as dynamical constraints, we observed a significant drop in prediction performance. This is evident from notable decreases in accuracy and substantial increases in both ADE, FDE, and collision measures (lightgrey and yellow). Comparing traditional attack strategies with those using dynamical and tactical behavior constraints in future states $\tilde{Y}_{tar}$, we noted that these additional constraints led to a significant drop in ADE and FDE attack performance (yellow/pink, yellow/white, yellow/lightblue). The impact of these extra restrictions is also reflected in the newly introduced measures. Both D_{peak} and D_{mean}, which measure deviation from the ground truth, are significantly lower for the newly created attacks compared to the traditional methods (blue/pink, blue/white, blue/lightblue). A similar pattern emerges for the control action measures (brown/pink, brown/white, brown/lightblue). Interestingly, the attack model focuses more on perturbations in the longitudinal

direction, as evidenced by a significant change in acceleration α with only slight modifications to curvature κ (darkgrey and brown).

When evaluating performance under adversarial attacks using a position-based perturbation strategy without dynamical constraints, we observed a significant increase in attack performance for ADE and FDE measures (lightgrey and violet). Unlike the control action dynamical constraint approach, restricting future states in this method led to even better attack performance (violet/pink, violet/white, violet/lightblue). However, the control action measures for this approach showed unrealistically high values for both acceleration α and curvature κ, indicating that this method is unsuitable for creating realistic attacks (darkgray and cyan).

Comparing performance under adversarial attacks using dynamical constraints according to the "Search" attack, we again observed a significant improvement in ADE and FDE measures over the control action dynamical constraints (lightgrey and olive). However, this performance increase is generally worse compared to the position-based approach, as expected, since this method applies dynamical constraints to the position-based approach (olive and violet). Similar to the position-based method, when evaluating the control actions performance measure, these values are unrealistically high (darkgray and magenta), indicating that both methods are unable to generate dynamically feasible adversarial trajectories.

2) Tactical Behavior Constraints: When comparing the two different tactical behavior constraints for adversarial observed states $\tilde{X}_{\text{tar}}$ - time-specific and time-trajectory-specific constraints - we observe several key differences. Generally, the time-specific approach shows better performance than its counterpart. However, there are exceptions, particularly for position-based perturbation strategies, though the advantage in these cases is marginal (beige/pink, beige/white, beige/lightblue, orange/pink, orange/white, orange/lightblue). For control action-based dynamical constraints, we observe that values for D_{peak} and D_{mean} are generally closer to each other compared to position (blue and beige) and "Search"-based (blue and orange) dynamical constraints. This suggests that changes are more evenly distributed over the whole trajectory when using control action-based constraints, indicating a prioritization of consistent modifications throughout the path rather than localized, extreme deviations. An interesting observation emerges for the position-based strategy under time-trajectory-specific constraints. In this scenario, we noted that the $D_{[\text{peak}]}$ deviations exceed the threshold $D_{[\text{max}]}$ of 0.9 meters (beige/pink, beige/white). This suggests that this particular attack strategy exploits the greater allowance for change in the longitudinal direction, effectively utilizing the full range of permitted deviations.

3) False Negative Collision: One of the novel attacks designed in this paper, which has no direct counterpart, is the false negative collision attack. For this attack, we adopt a different evaluation approach. Instead, we compare the ADE and FDE with the prediction in the nominal setting $\hat{Y}_{\text{tar}}$, taking into account the inherent prediction errors present in the nominal setting. For the control action dynamical constraints, we observe that predictions on adversarial states $\hat{\tilde{Y}}_{\text{tar}}$ closely align with the predictions in the nominal setting $\hat{Y}_{\text{tar}}$ (yellow/lightorange). This alignment is not observed in the other two approaches (violet/lightorange, olive/lightorange). This difference is also reflected in the D_{peak} and D_{mean} measures, where the control action approach shows minimal deviation from the ground truth (blue/lightorange), in stark contrast to the other approaches (beige/lightorange, orange/lightorange). A significant finding across all three attacks is their ability to create perturbations in the perturbed future states $\tilde{Y}_{\text{tar}}$ that result in collisions. In approximately 70% of the samples, a collision occurs, regardless of the constraint combination used (red/lightorange).

B. Qualitative Results

To provide a more comprehensive understanding of the attack objectives performance, we conducted a detailed analysis of a specific left-turning sample from the L-GAP dataset. The actual prediction performance for all attack objectives using different dynamical and tactical behavior constraints are visualized in the appendix (Figures 6-26). For consistency, we applied the same settings used in the quantitative analysis, as noted in the figure captions. We selected time-specific constraints for the tactical behavior in the observed states, as these generally performed the best (Table II). This in-depth examination revealed several significant insights.

1) Nominal Setting: In the nominal setting, we observed that the model struggles with accurate predictions, failing to correctly estimate the curvature and acceleration of future states, which highlights inherent limitations in the prediction model even without adversarial interference.

2) Attack Performance: When analyzing the attacks and their behavior in relation to the loss function, we observe that the attacks generally behave as expected for all attack objectives, with the exception of the false negative collision attack (Figure 24 and 25). This attack exhibits some unique characteristics. For the position-based perturbation strategy without dynamical constraints, we observe that it forces one of the points in the future states to converge to the ego agent's future states Y_{ego}. This behavior is logical given that the objective is to minimize the distance at a specific time step, and subsequent states are not constrained relative to each other. This results in an unrealistic trajectory where only one point aligns with the collision scenario. In contrast, the Search attack shows no modification at all in future states for this particular objective. This suggests that the position and "Search" based method may not be effective for the false negative collision attack scenario, this observation aligns with the quantitative results (Table II). For the control action strategy, we see the generation of an actual trajectory causing a collision. This behavior is more realistic and aligned with the attack objective, as subsequent states are constrained relative to each other due to the inherent properties of the control action approach.

3) Observed States: Examining the observed states reveals a significant finding that verifies our quantitative results. We

previously noted that D_{peak} and D_{mean} are generally closer to each other for the control action-based dynamical constraints compared to position and "Search" based dynamical constraints. The position and "Search" based approaches primarily focus on perturbing the final points of the observed state X_{tar}. In contrast, the control action-based approach shows a more gradual increase in deviation from the ground truth X_{tar}. This gradual deviation explains why the values of both measures are closer to each other and results in the creation of more realistic trajectories. Specifically, the control action-based method produces a smoother transition from the initial to the final state, with perturbations distributed more evenly across the trajectory. This is in stark contrast to the abrupt, localized changes seen in the position and "Search" based methods, which tend to concentrate their modifications towards the end of the trajectory.

4) Future States: Analyzing the future states reveals distinct patterns across different approaches. For the control action-based approach, we observe that the trajectory smoothly transitions from the adversarial observed states $\tilde{X}_{\text{tar}}$ to the perturbed future states $\tilde{Y}_{\text{tar}}$. This transition occurs within the bounded area defined by the tactical behavior constraints (Figure 1C), successfully creating trajectories where both observed and future states comply with the tactical behavior of the intended trajectory. In contrast, the position and "Search" based attacks exhibit a markedly different behavior. For these approaches, we observe that the perturbed future states remain unmodified. Instead, the final states of the adversarial observed states $\tilde{X}_{\text{tar}}$, which already exhibit unrealistic behavior, are directly connected to the ground truth future states Y_{tar}. This creates a discontinuity in the trajectory, where the adversarial modifications abruptly end at the transition point between observed $\tilde{X}_{\text{tar}}$ and future states $\tilde{Y}_{\text{tar}}$. This difference highlights a key advantage of the control action-based approach, its ability to maintain consistency and realism across the entire trajectory, from adversarial observed $\tilde{X}_{\text{tar}}$ to perturbed future states $\tilde{Y}_{\text{tar}}$.

VII. Discussion

Our comprehensive analysis of various adversarial attack strategies on the state-of-the-art trajectory prediction model revealed several key findings. Control action-based dynamical constraints, while resulting in more realistic trajectories, showed lower attack performance compared to position-based and "Search" methods. However, these latter methods often produced unrealistic trajectories with implausibly high acceleration and curvature values. Time-specific tactical behavior constraints generally outperformed time-trajectory-specific constraints, with some exceptions in position-based strategies. The false negative collision attack demonstrated the ability to create perturbations leading to collisions in about 70% of samples, regardless of constraint combinations. Qualitative analysis further supported these findings, highlighting the control action-based approach's superiority in maintaining trajectory consistency and realism. While these findings provide valuable insights into the performance and characteristics of different adversarial attack strategies, they also highlight areas where our current approach could be improved. Through careful examination of both our quantitative and qualitative results, we have identified several limitations in our methodology and found areas for future research.

A. Limitation Dataset

For this project, we utilize the L-GAP dataset [20]. This dataset originally includes an equal distribution of samples: half involving left-turning vehicles and half with vehicles standing still in front of the intersection. The ego agent is initialized at varying distances from the intersection, while the target agent crosses the intersection with different time and distance margins between the two vehicles. These varying margins pose a challenge for the traditional collision, false positive collision, and false negative collision attacks, as they rely on the distance margins to create crashes in target agent predictions. The wide range of margins (from 5 meters to 80 meters) results in highly variable loss values, potentially leading to the problem of exploding gradients during attack generation. To address this issue, we utilized the gap acceptance behaviour information of the dataset by sampling instances with a high likelihood of critical decision-making situations. This selects samples where the ego agent's future states are near the intersection, and the target agent is still approaching the intersection. While this approach successfully mitigated the exploding gradients problem, it significantly impacted the dataset balance. Upon analyzing the refined dataset, we observed a substantial imbalance: only 53 samples involve making a left turn, while 973 samples represent instances where the subjects are standing still. This imbalance is crucial to note because the samples of stationary subjects are also affected by adversarial attacks, which can significantly influence the quantitative results. The disproportionate representation of stationary samples makes it challenging to accurately assess the performance of the attack objectives across the entire dataset.

Another significant concern related to the dataset is its inherent bias, which results in the vehicle either standing still or making a left turn. To address this limitation, we utilized both the L-GAP and NuScenes datasets during training. However, balancing the training data between L-GAP and the more dominant NuScenes introduced a new issue: compromised prediction performance in nominal settings, as evident in both quantitative results (Table II) and qualitative results (Figures 6-26).

This issue is further complicated by the fact that the L-GAP dataset lacks training images, whereas the NuScenes dataset provides figures for training. In an attempt to address this dataset imbalance and the standing still bias, we created a map for the L-GAP dataset similar to those used for training with NuScenes to improve prediction performance (Figure 27). Unfortunately, using this figure resulted in even poorer predictions (Figure 28).

Further limitation of the dataset is the inclusion of crash scenarios involving both the target and ego agent. These crash

scenarios pose a particular challenge for the data statistics dynamical approach "Search" proposed by Zhang et al. [5]. Crash events typically involve extreme vehicle dynamics, characterized by high accelerations and curvatures that fall outside normal driving patterns. The approach by Zhang et al. uses the entire dataset, including these crash data, to establish the bounds of normal vehicle behavior. As a result, their method struggles to differentiate between realistic adversarial trajectories and physically implausible ones, leading to the generation of dynamically infeasible adversarial trajectories.

B. Future Works

Following an in-depth analysis of our findings, we identified seven research opportunities that could be addressed using our perturbation framework:

- This paper primarily focuses on the utilization of ADE, FDE, and collision attacks. Future research could investigate the performance and implications of lateral/longitudinal attacks using perturbed future states.
- While the study effectively utilizes the L-GAP dataset, evaluating the generalizability of the attacks would be greatly enhanced by applying them to more widely recognized autonomous vehicle datasets, such as NuScenes [21]. NuScenes includes a broad range of real-world driving scenarios, making it a better benchmark for testing the attacks' performance in diverse conditions.
- The idea presented in this paper is to perturb both observed and future states of the vehicle's trajectory. This approach improves the realism of adversarial examples by maintaining consistent tactical behavior and enables the creation of new datasets. These datasets, which reflect adversarial conditions, can be used to fine-tune prediction models. Training on these datasets may help models better anticipate and mitigate adversarial attacks, thereby improving their overall robustness.
- Two approaches are utilized to generate attacks that align with the expected tactical behavior. First, we limited the control action relative to their original control actions. Second, we applied tactical behavior constraints that penalize trajectories deviating from their ground truth. Future research could investigate how these approaches influence each other and their impact on the effectiveness of the attack.
- In our analysis of various combinations of dynamical constraints and tactical behavior constraints, we maintained a consistent step size α and decay rate γ for the step size across all attacks. However, to further optimize the performance of these attacks, future research should focus on hyperparameter tuning.
- For the dataset settings, we utilized a step size Δt of 0.1 seconds, resulting in small positional changes. Future research could experiment with the impact of increasing the step size Δt, extending the observation period H, or expanding the prediction horizon T.
- In this research, we set the distance threshold $D_{\max}$ to 0.9 meters. Future studies could investigate how changes in this attack budget would impact the performance of the adversarial attacks.

VIII. Conclusion

This paper introduces novel strategies for generating adversarial attacks on trajectory prediction models, balancing the goal of misleading the prediction model while creating realistic driving scenarios. Our approach, incorporating new dynamical and tactical behavior constraints, demonstrates effectiveness in impacting prediction performance while maintaining plausible vehicle behavior. Experiments reveal significant increases in ADE, FDE, and collision measures under adversarial conditions and expose model vulnerabilities. Additionally, we develop four new performance measures for evaluating the realism and impact of adversarial trajectories. However, limitations in dataset balance highlight areas for future research. This work contributes to improving the robustness of trajectory prediction models in autonomous driving systems, providing a framework for future research in AV safety and reliability.

Acronyms

ADE Average Displacement Error. 3–5, 8, 9, 12–14, 16
AV Autonomous vehicle. 1, 4, 7, 10, 11, 16
FDE Final Displacement Error. 3–5, 9, 12–14, 16
PGD Projected Gradient Descent. 3–6, 8, 9
PSO Particle Swarm Optimization. 3

References

[1] J. Hagenus, F. B. Mathiesen, J. F. Schumann, and A. Zgonnikov, "A survey on robustness in trajectory prediction for autonomous vehicles," 2024. arXiv:2402.01397.

[2] Y. Cao, C. Xiao, A. Anandkumar, D. Xu, and M. Pavone, "AdvDO: Realistic adversarial attacks for trajectory prediction," in *Eur. Conf. on Comput. Vis.*, 2022.

[3] S. Saadatnejad, M. Bahari, P. Khorsandi, M. Saneian, S.-M. Moosavi-Dezfooli, and A. Alahi, "Are socially-aware trajectory prediction models really socially-aware?," in *Transp. Research Part C: Emerg. Technol.*, 2022.

[4] Y. Cao, D. Xu, X. Weng, Z. Mao, A. Anandkumar, C. Xiao, and M. Pavone, "Robust trajectory prediction against adversarial attacks," in *PMLR The 6th Conf. on Robot Learn.*, 2023.

[5] Q. Zhang, S. Hu, J. Sun, Q. A. Chen, and Z. M. Mao, "On adversarial robustness of trajectory prediction for autonomous vehicles," 2022. arXiv:2201.05057.

[6] R. Jiao, X. Liu, T. Sato, Q. A. Chen, and Q. Zhu, "Semi-supervised Semantics-guided Adversarial Training for Robust Trajectory Prediction," in *IEEE/CVF Int. Conf. on Comput. Vis.*, Oct. 2023.

[7] K. Tan, J. Wang, and Y. Kantaros, "Targeted adversarial attacks against neural network trajectory predictors," 2022. arXiv:2212.04138.

[8] A. Tocchetti, L. Corti, A. Balayn, M. Yurrita, P. Lippmann, M. Brambilla, and J. Yang, "A.i. robustness: a human-centered perspective on technological challenges and opportunities," 2022. arXiv:2210.08906.

[9] H. Yin, J. Li, P. Zhen, and J. Yan, "Sa-attack: Speed-adaptive stealthy adversarial attack on trajectory prediction," 2024. arXiv:2404.12612.

[10] D. Rempe, J. Philion, L. J. Guibas, S. Fidler, and O. Litany, "Generating useful accident-prone driving scenarios via a learned traffic prior," 2022. arXiv:2112.05077.

[11] J. Wang, A. Pun, J. Tu, S. Manivasagam, A. Sadat, S. Casas, M. Ren, and R. Urtasun, "AdvSim: Generating safety-critical scenarios for self-driving vehicles," in *IEEE/CVF Conf. on Comput. Vis. Pattern Recognit.*, IEEE, 2021.

[12] Z. Zheng, X. Ying, Z. Yao, and M. C. Chuah, "Robustness of trajectory prediction models under map-based attacks," in *Proc. IEEE/CVF Winter Conf. on Appl. Comput. Vis.*, 2023.

[13] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Eur. Conf. on Comput. Vis.*, 2020.
[14] Y. Yuan, X. Weng, Y. Ou, and K. Kitani, "AgentFormer: Agent-aware transformers for socio-temporal multi-agent forecasting," in *IEEE/CVF Int. Conf. on Comput. Vis.*, 2021.
[15] Y. Qin, Y. Xiong, J. Yi, and C.-J. Hsieh, "Training meta-surrogate model for transferable adversarial attack," in *Proc. AAAI Conf. on Artif. Intell.*, 2023.
[16] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," Tech. Rep. CMU-RI-TR-92-01, Carnegie-Mellon University Pittsburgh PA Robotics Institute, 1992.
[17] A. Duan, R. Wang, Y. Cui, P. He, and L. Chen, "Causal Robust Trajectory Prediction Against Adversarial Attacks for Autonomous Vehicles," *IEEE Internet Things J.*, 2023.
[18] M. Schmidt, "Projected newton first-order optimization algorithms for machine learning projected-gradient methods." https://www.cs.ubc.ca/~schmidtm/Courses/5XX-S20/S5.pdf, 2020. University of British Columbia, Summer 2020.
[19] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.
[20] A. Zgonnikov, D. Abbink, and G. Markkula, "Should i stay or should i go? cognitive modeling of left-turn gap acceptance decisions in human drivers," in *Hum. Factors: The J. Hum. Factors Ergon. Soc.*, 2024.
[21] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," in *IEEE/CVF Conf. on Comput. Vis. Pattern Recognit.*, 2020.
[22] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *IEEE 12th Int. Conf. on Comput. Vis.*, 2009.
[23] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *IEEE Conf. on Comput. Vis. Pattern Recognit.*, 2016.

APPENDIX

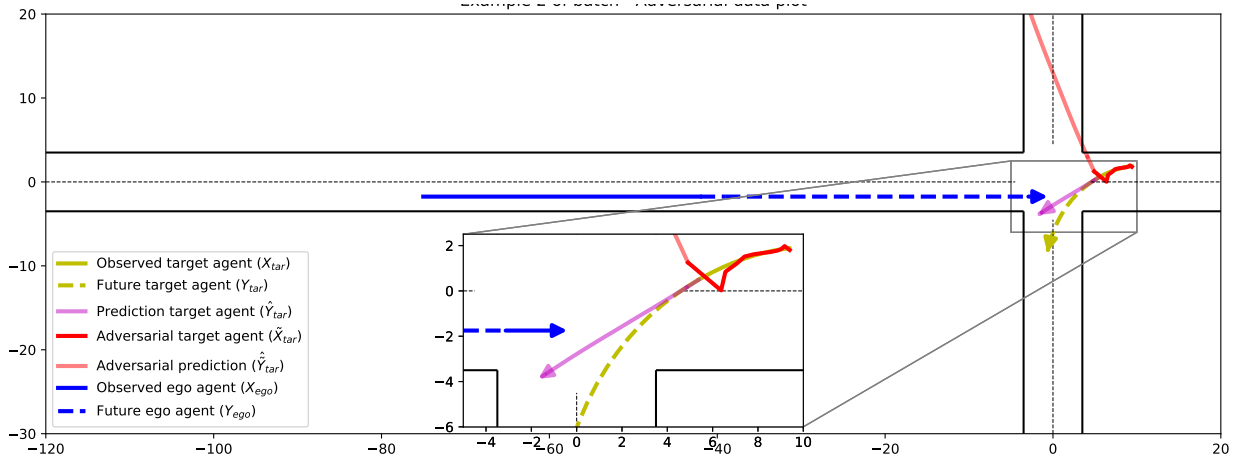


Fig. 6. Traditional ADE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Position - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - Extra: Train and prediction without map. Summary: Traditional ADE attack without dynamical constraints creates an unrealistic trajectory, with large deviations in the final states of the observed states.

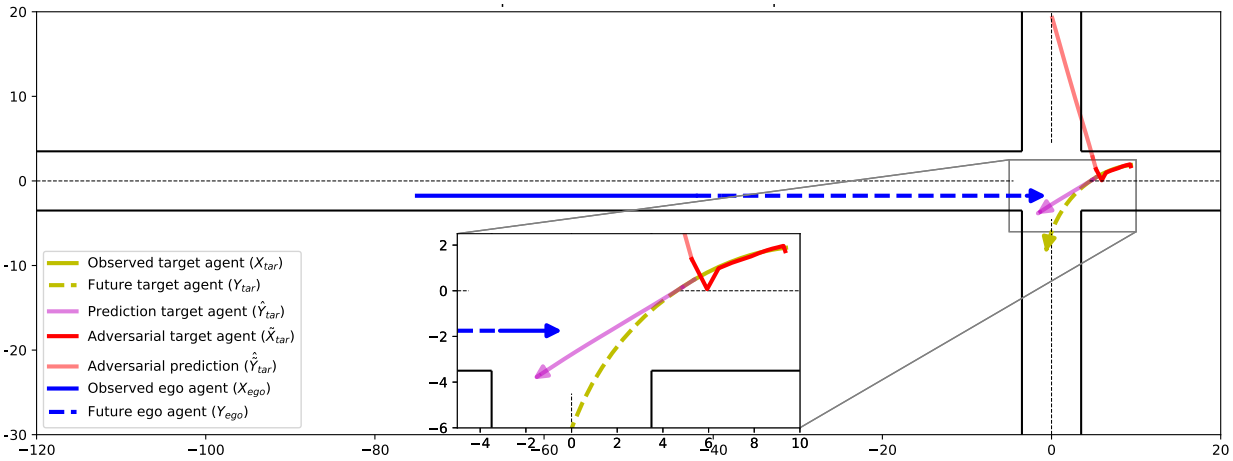


Fig. 7. Traditional ADE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Search - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - Extra: Train and prediction without map. Summary: Traditional ADE attack using "Search" dynamical constraints creates an unrealistic trajectory, with large deviations in the final states of the observed states.

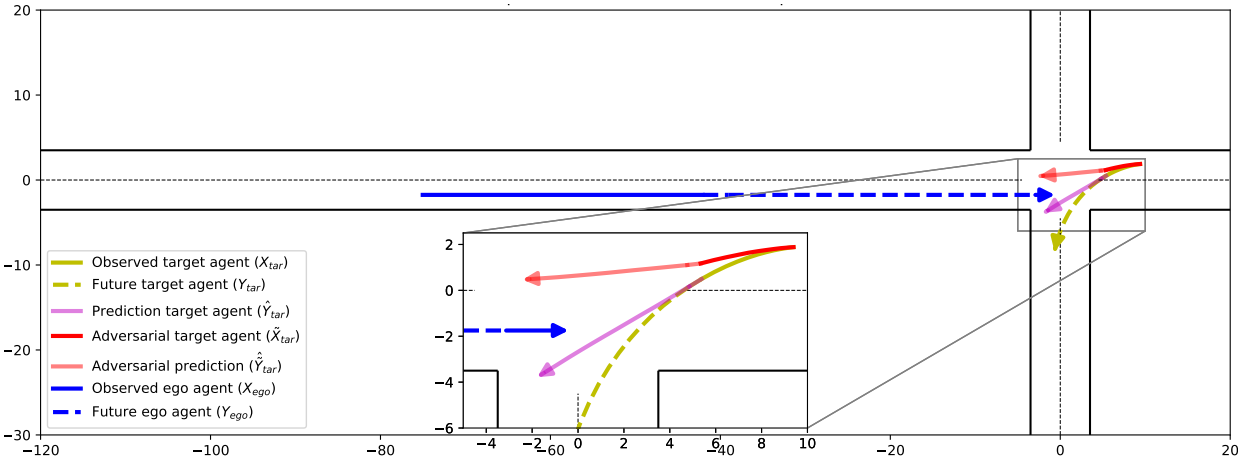


Fig. 8. Traditional ADE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Control action - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - Extra: Train and prediction without map. Summary: Traditional ADE attack with control action dynamical constraints creates a realistic trajectory, with a gradual increase in deviation from the ground truth.

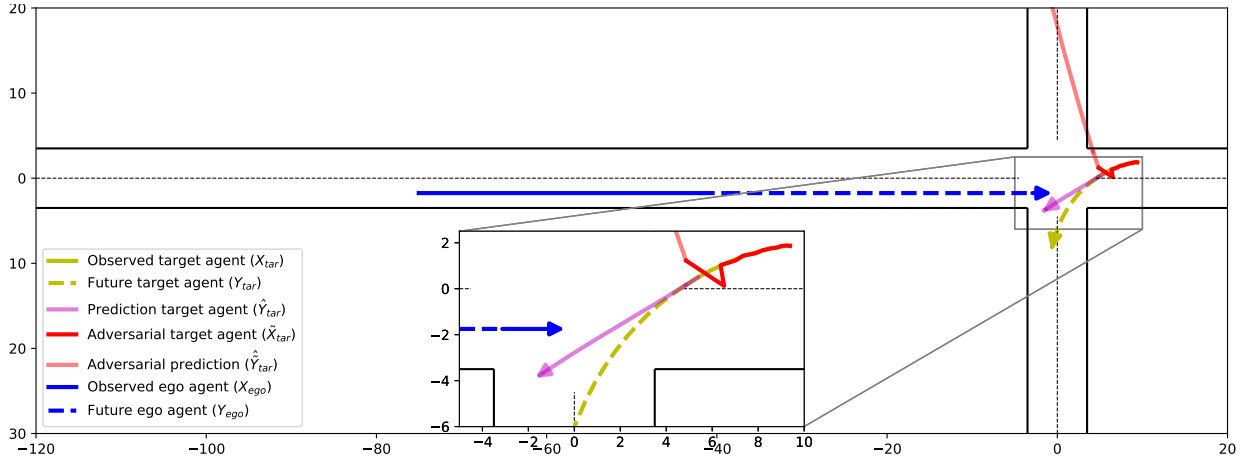


Fig. 9. Traditional FDE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Position - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - Extra: Train and prediction without map. Summary: Traditional FDE attack without dynamical constraints creates an unrealistic trajectory, with large deviations in the final states of the observed states.

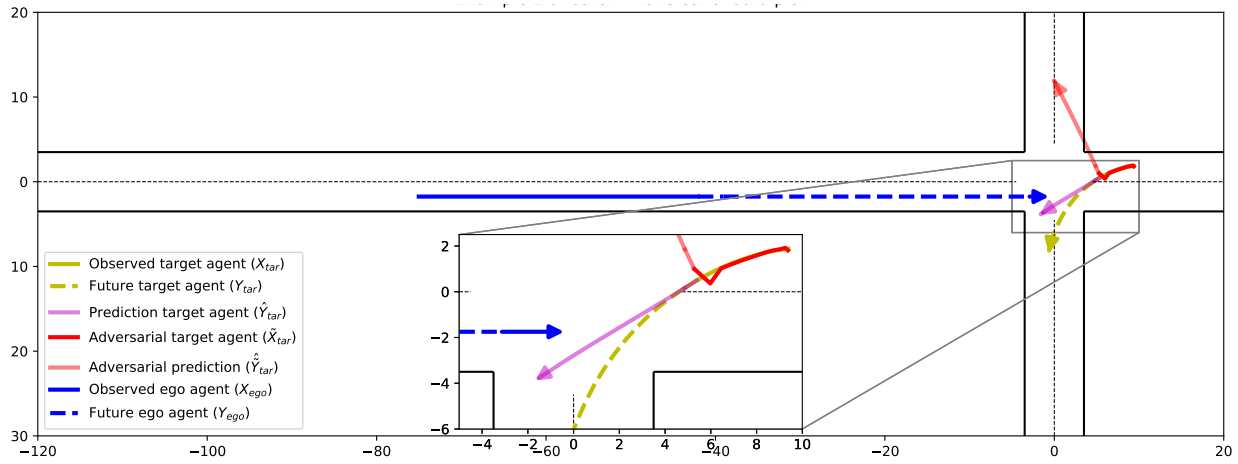


Fig. 10. Traditional FDE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Search - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - Extra: Train and prediction without map. Summary: Traditional FDE attack with "Search" dynamical constraints creates an unrealistic trajectory, with large deviations in the final states of the observed states.

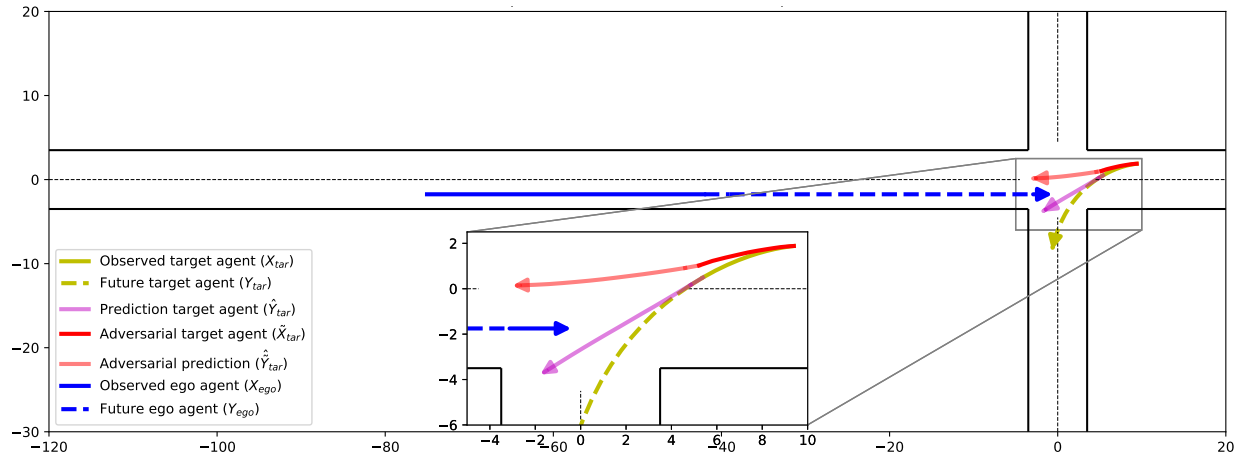


Fig. 11. Traditional FDE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Control action - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - Extra: Train and prediction without map. Summary: Traditional FDE attack with control action dynamical constraints creates a realistic trajectory, with a gradual increase in deviation from the ground truth.

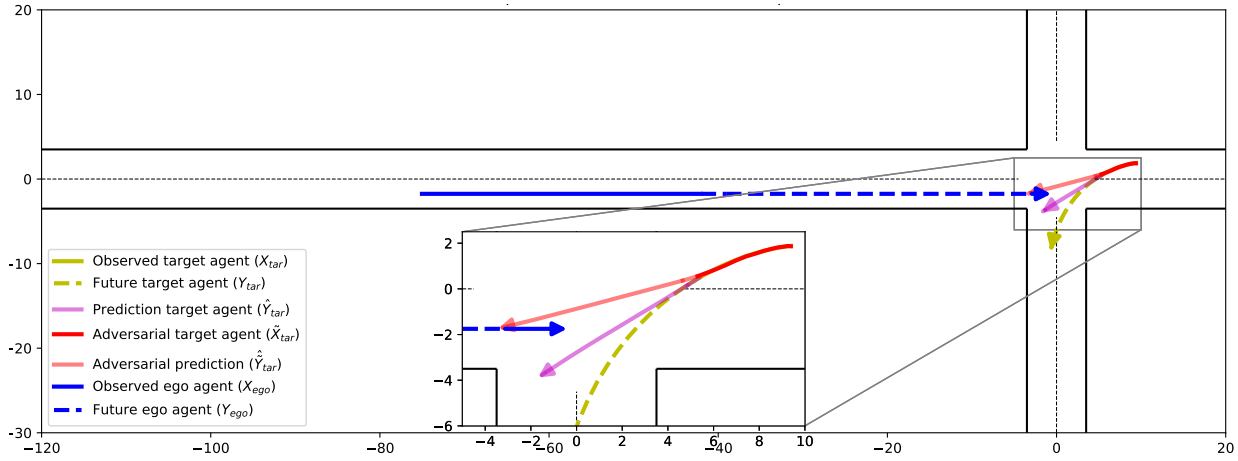


Fig. 12. Traditional collision attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Position - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - Extra: Train and prediction without map. Summary: Traditional collision attack without dynamical constraints creates a realistic trajectory, with a small deviation in the final state of the observed states.

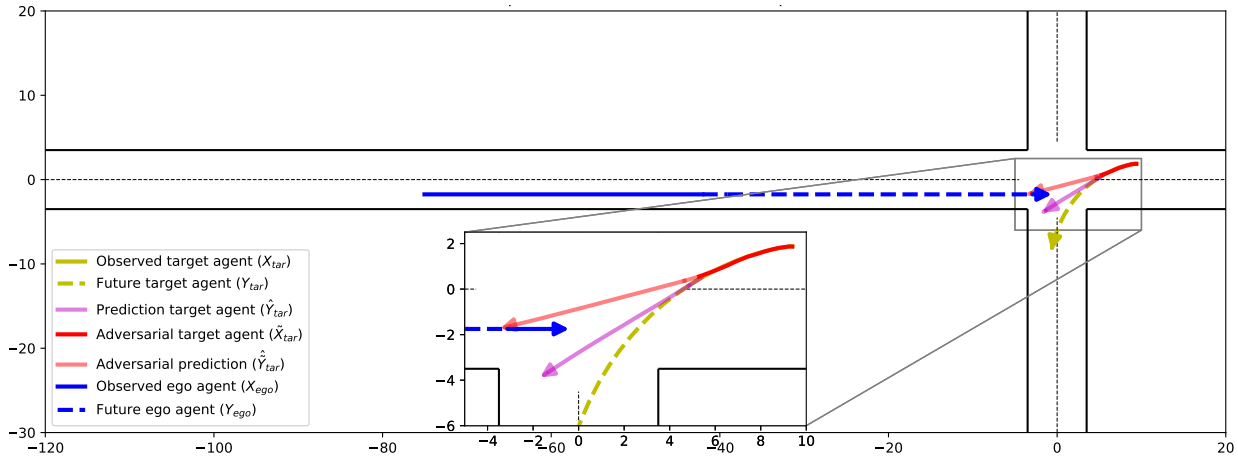


Fig. 13. Traditional collision attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Search - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - Extra: Train and prediction without map. Summary: Traditional collision attack with "Search" dynamical constraints creates a realistic trajectory, with a small deviation in the final state of the observed states.

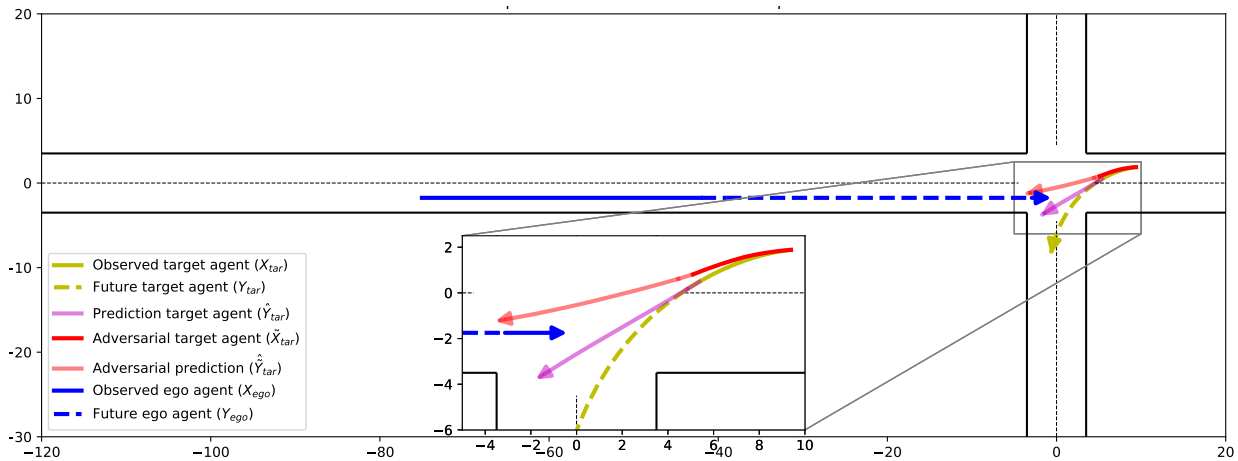


Fig. 14. Traditional collision attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Control action - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - Extra: Train and prediction without map. Summary: Traditional collision attack with control action dynamical constraints creates a realistic trajectory, with a gradual increase in deviation from the ground truth.

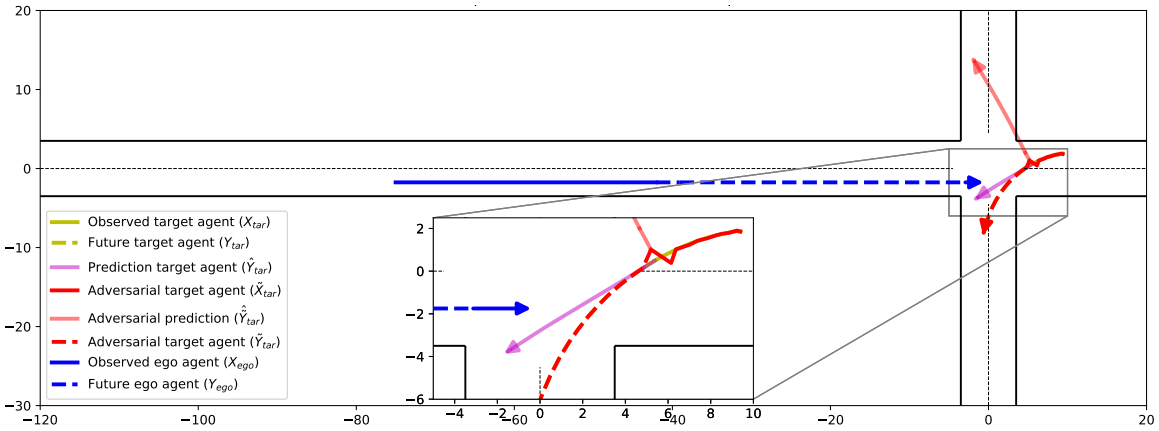


Fig. 15. Feasible ADE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Position - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Position - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: Feasible ADE attack without dynamical constraints creates an unrealistic trajectory, with large deviations in the final states of the observed sequence. The final observed state is directly connected to the first state of the ground truth future states without modifications to those future states.

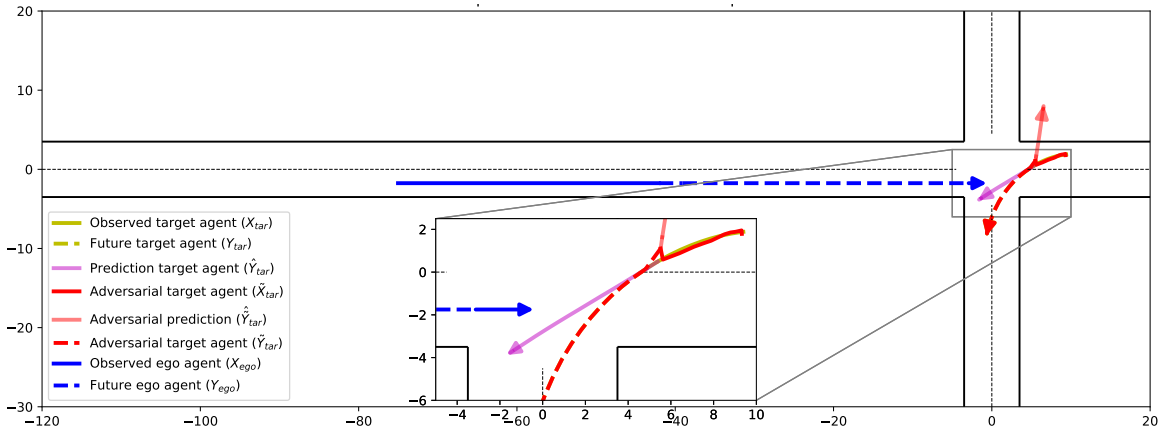


Fig. 16. Feasible ADE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Search - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Search - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: Feasible ADE attack with "Search" dynamical constraints creates an unrealistic trajectory, with large deviations in the final states of the observed sequence. The final adversarial observed state is directly connected to the first state of the ground truth future states without modifications to those future states.

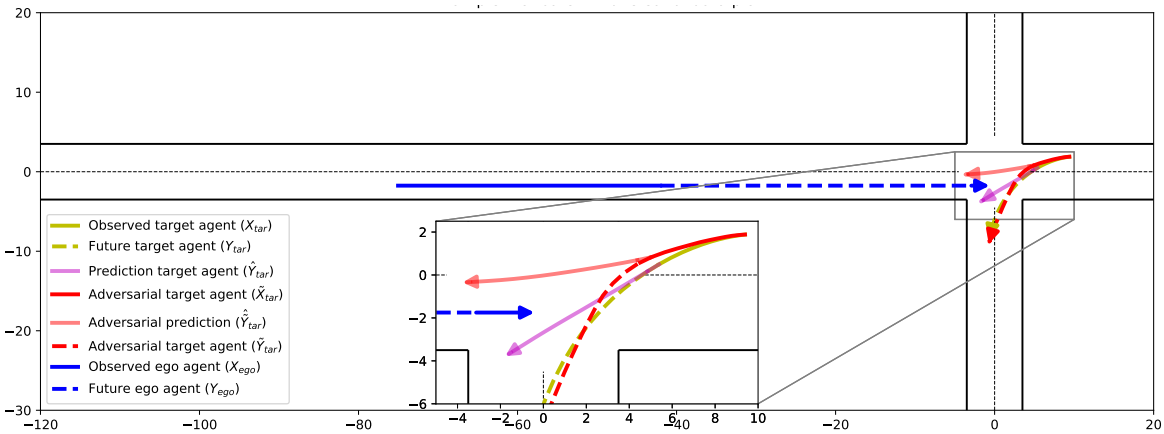


Fig. 17. Feasible ADE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Control action - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Control action - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: Feasible ADE attack with control action dynamical constraints creates a realistic trajectory, with a gradual increase in deviation from the ground truth. The adversarial observed states smoothly transition to the adversarial future states that closely resemble the future ground truth states.

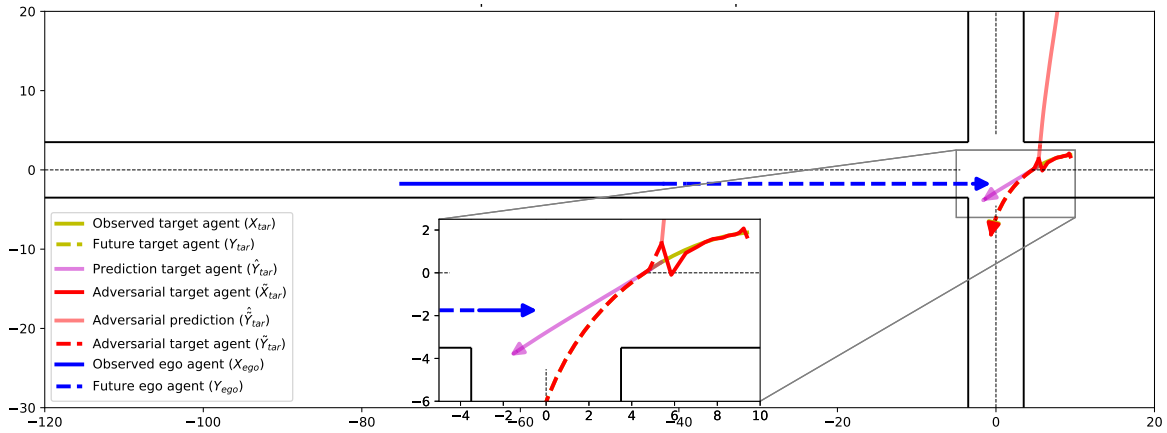


Fig. 18. Feasible FDE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Position - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Position - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: Feasible FDE attack without dynamical constraints creates an unrealistic trajectory, with large deviations in the final states of the observed sequence. The final adversarial observed state is directly connected to the first state of the ground truth future states without modifications to those future states.

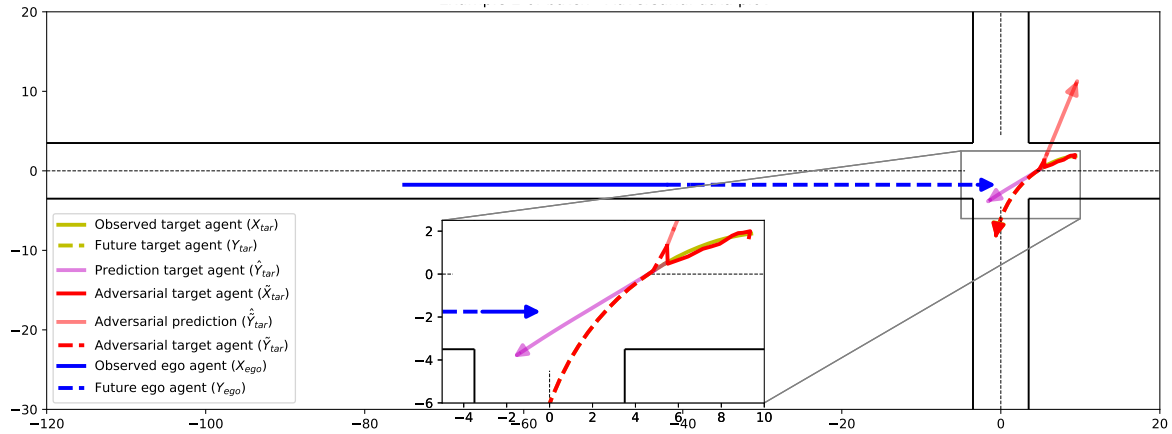


Fig. 19. Feasible FDE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Search - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Search - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: Feasible FDE attack with "Search" dynamical constraints creates an unrealistic trajectory, with large deviations in the final states of the observed sequence. The final adversarial observed state is directly connected to the first state of the ground truth future states without modifications to those future states.

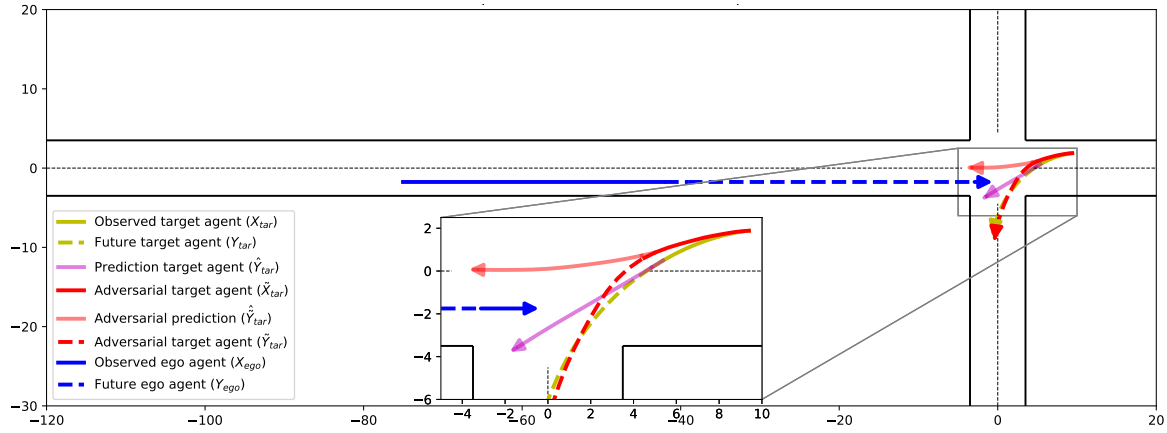


Fig. 20. Feasible FDE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Control action - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Control action - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: Feasible FDE attack with control action dynamical constraints creates a realistic trajectory, with a gradual increase in deviation from the ground truth. The adversarial observed states smoothly transition to the adversarial future states that closely resemble the future ground truth states.

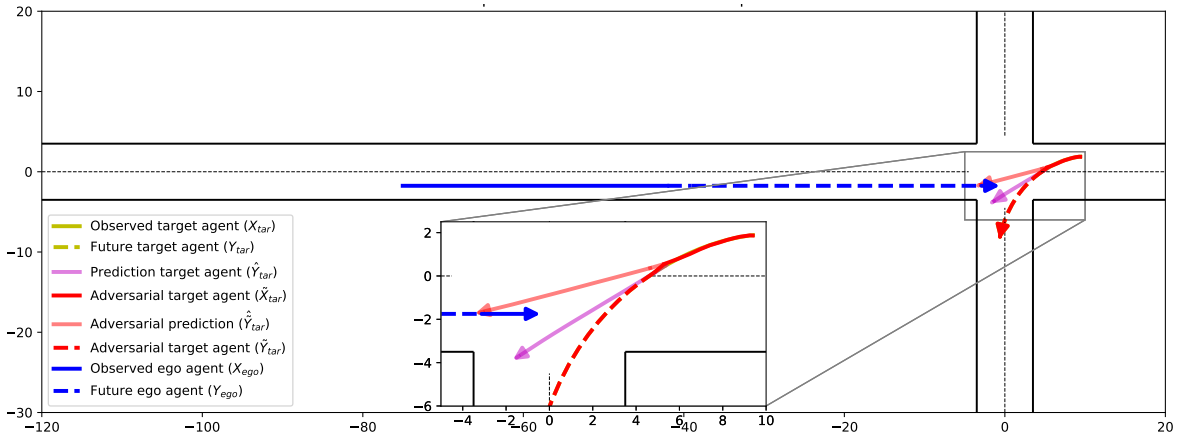


Fig. 21. False positive collision attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Position - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Position - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: False positive collision attack without dynamical constraints creates a realistic trajectory, with a small deviation in the final state of the observed states. The final adversarial observed state is directly connected to the first state of the ground truth future states without modifications to those future states.

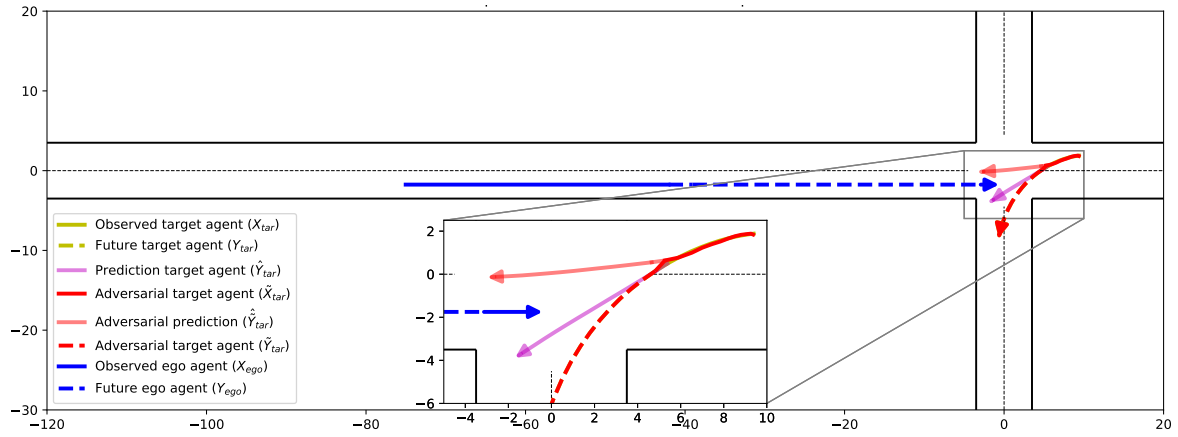


Fig. 22. False positive collision attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Search - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Search - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: False positive collision with "Search" dynamical constraints creates a realistic trajectory, with a small deviation in the final state of the observed states. The final adversarial observed state is directly connected to the first state of the ground truth future states without modifications to those future states.

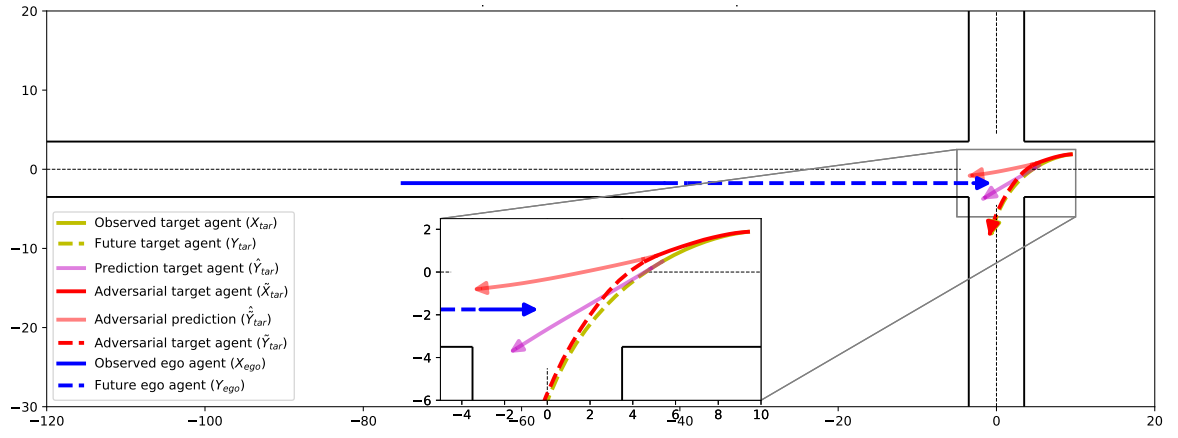


Fig. 23. False positive collision attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Control action - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Control action - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: False positive collision attack with control action dynamical constraints creates a realistic trajectory, with a gradual increase in deviation from the ground truth. The adversarial observed states smoothly transition to the adversarial future states that closely resemble the future ground truth states.

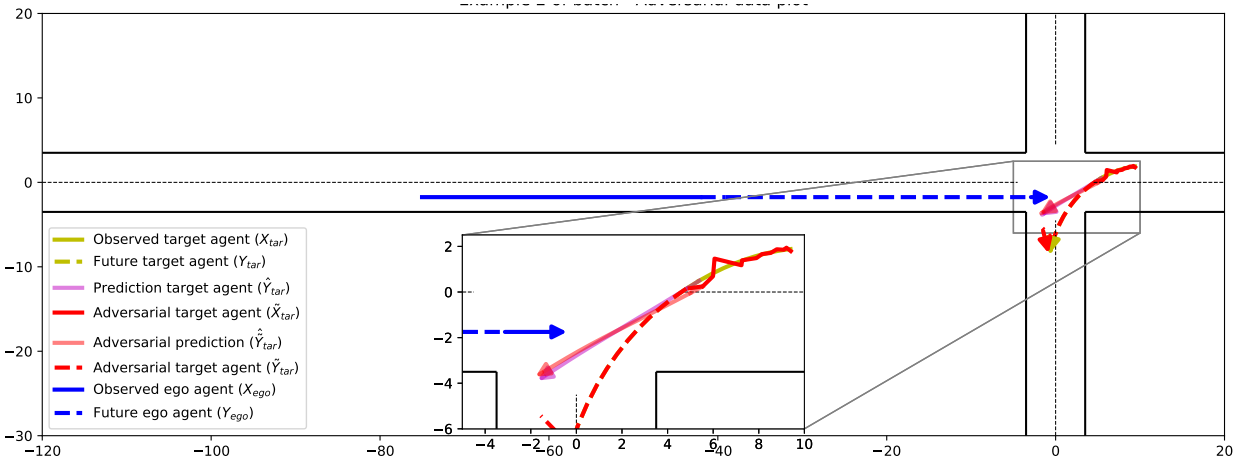


Fig. 24. False negative collision attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Position - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Position - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: The prediction on the adversarial observed states is directed toward the nominal setting prediction. The perturbed future states align with the ground truth, except for one specific timestep that converges to the future states of the ego agent to cause a collision. However, the trajectory does not look realistic for observed and future states.

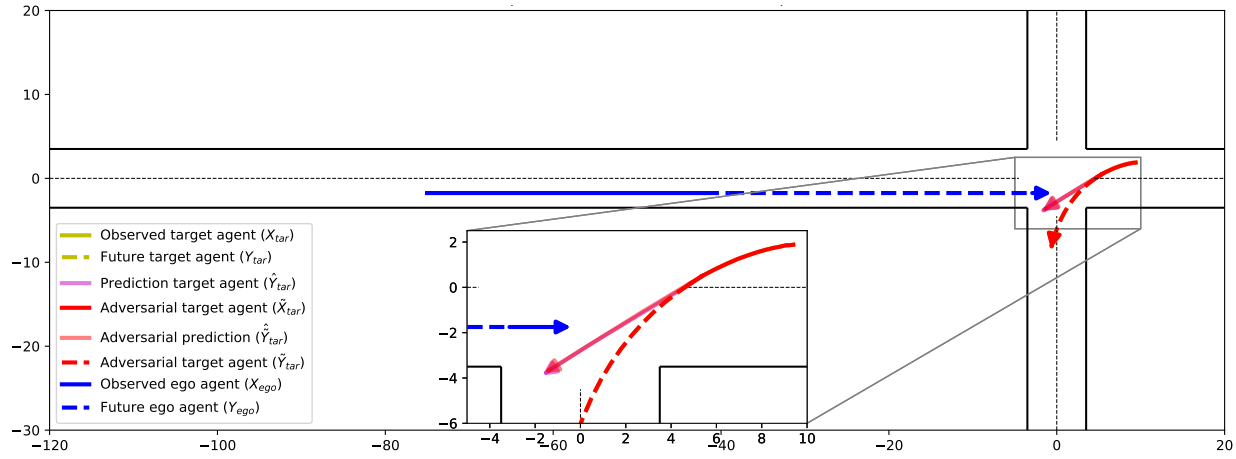


Fig. 25. False negative collision attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Search - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Search - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: The prediction on the adversarial observed states is directed toward the nominal setting prediction. The perturbed future states are not moving in the correct direction to cause a collision.

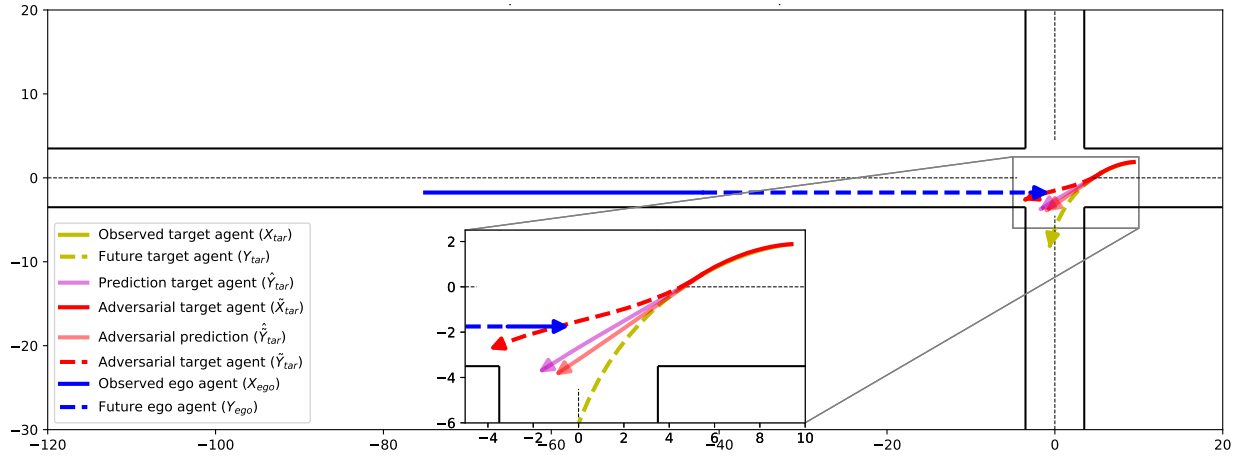


Fig. 26. False negative collision attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Control action - $l_{\text{dyn}}(\tilde{Y}_{\text{tar}})$: Control action - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - $l_{\text{tac}}(\tilde{Y}_{\text{tar}})$: Trajectory-specific - Extra: Train and prediction without map. Summary: The prediction on the adversarial observed states is directed toward the nominal setting prediction. The perturbed future states are directed toward the ego agent, causing a collision.

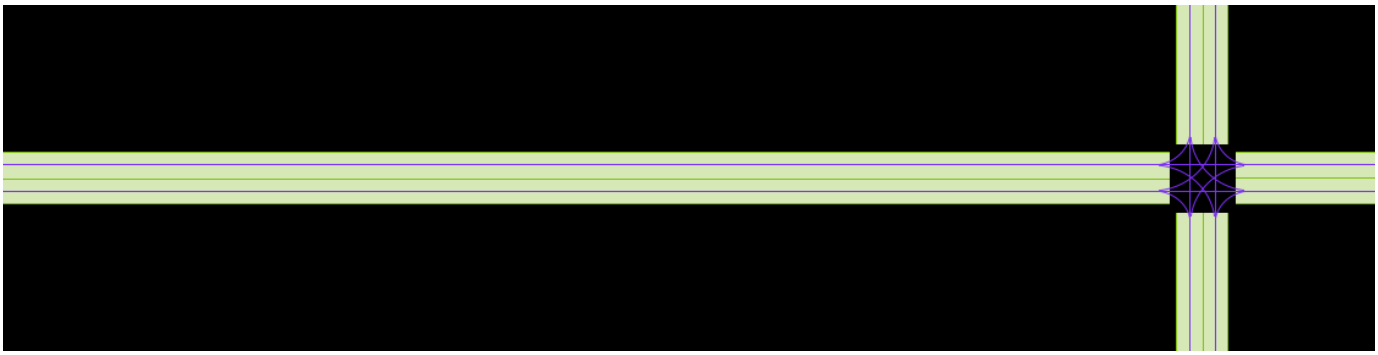


Fig. 27. L-GAP dataset map used during training

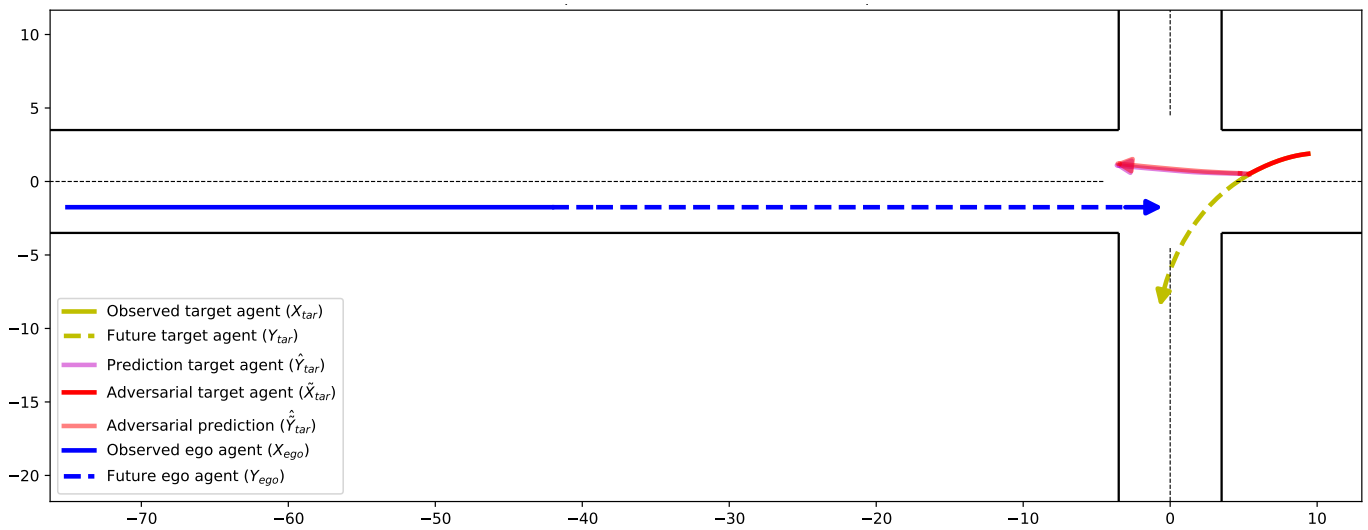


Fig. 28. ADE attack (Train: NuScene + L-GAP, Evaluate: L-GAP) - Max iteration: 100 - α: 0.01 - γ: 0.99 - $l_{\text{dyn}}(\tilde{X}_{\text{tar}})$: Control action - $l_{\text{tac}}(\tilde{X}_{\text{tar}})$: Time-specific - Extra: Train and prediction with map. Summary: A prediction model is tested using map information to improve prediction accuracy (Figure 27). However, it did not perform as expected and is therefore not utilized.