# COLLIGEN: PHYSICS-GUIDED SELF-LEARNING FOR GENERATING COLLISION TRAJECTORIES OF VEHICLES

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Collision data are critical for improving the safety and robustness of autonomous driving systems. However, real collision trajectories are scarce and expensive to collect. We address this limitation by generating physically plausible collision trajectories using pretrained normal-driving predictors from abundant normal-driving data. We propose COLLIGEN, a physics-guided self-training framework that progressively adapts a normal-driving trajectory model into a collision trajectory generator. COLLIGEN consists of two training phases. In the Pre-collision phase, the model learns trajectories that lead to impact while remaining behaviorally plausible; in the Post-collision phase, it learns trajectories that satisfy post-impact physical constraints. To guide the adaptation, we adopt Markov Chain Monte Carlo sampling with physics-based scoring to enhance collision feasibility, dynamic consistency, and motion regularity of self-generation. Experiments show that COLLIGEN substantially outperforms strong text-to-trajectory, trajectory-to-trajectory, and trajectory-prediction baselines in reconstructing collision trajectories synthesized from NHTSA accident reports. Qualitative results show that COLLIGEN produces more realistic and physically plausible pre- and post-collision motion.

## 1 INTRODUCTION

Scenario generation has emerged as a key for training and evaluating autonomous driving systems, enabling models to encounter diverse and safety-critical situations beyond those frequently observed in real-world driving (Ding et al., 2023; Xu et al., 2022; Hanselmann et al., 2022). Recent studies suggest that incorporating collision and post-collision data into autonomous driving models can improve both collision avoidance and overall driving performance (Wang et al., 2025; Zhang et al., 2025; Patrikar et al., 2025). Such data expose perception, prediction, and planning modules to safety-critical situations that are rarely observed in ordinary driving logs, providing valuable supervision for understanding failure modes, impact dynamics, and recovery behaviors.

Large-scale normal-driving data are readily available from real-world datasets such as Waymo (Ettinger et al., 2021) and Argoverse (Chang et al., 2019). Recent trajectory models trained on such nominal traffic behaviors (Liang et al., 2020; Zhou et al., 2022; Tang et al., 2024; Feng et al., 2022) have shown substantial progress in motion forecasting and traffic simulation. However, they are poorly suited for synthesizing rare collision events, especially when the desired scenario involves vehicle interactions or post-impact motion. Generating collision data is particularly challenging because the resulting trajectories must be physically plausible. A collision synthesis must account for contact, impact response, and post-collision motion. Therefore, physics-aware generation (Raissi et al., 2019; Karniadakis et al., 2021) is essential for producing collision data that can serve as a meaningful supervision. However, the problem of jointly generating physically consistent pre- and post-collision trajectories remains largely unexplored, even for the case of two interacting vehicles.

We present COLLIGEN, a physics-guided self-learning framework that progressively adapts a normal-driving trajectory model into a collision trajectory generator for two vehicles. Starting from a base predictor trained on nominal driving data, COLLIGEN performs two phases of iterative self-training: in the Pre-collision Phase, COLLIGEN learns pre-collision trajectories that lead to eventual impact while remaining behaviorally plausible; in the Post-collision Phase, it learns post-collision trajectories that satisfy dynamic consistency and motion regularity constraints. This decomposition provides a scalable way to synthesize physically plausible pre- and post-collision trajectories. To facilitate



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

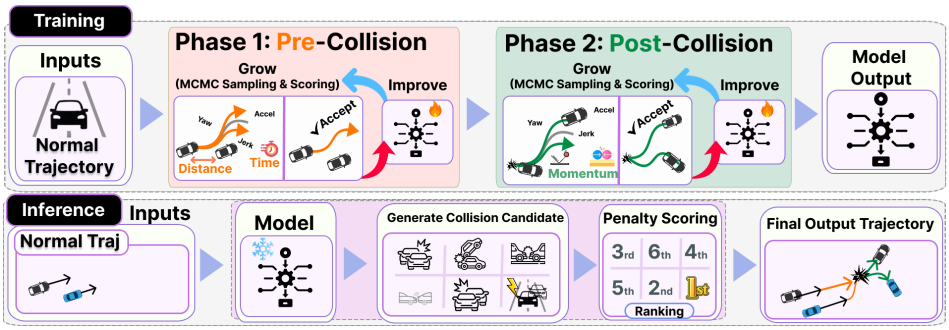



Fig 1: **Overview.** COLLIGEN is a self-training framework that progressively adapts a base trajectory model into a collision generator through pre-collision and post-collision phases. The model iteratively self-generates trajectory data (Grow) and is fine-tuned (Improve) through physics-guided sampling, scoring, and filtering. At inference, the model is input with an initial trajectory, generates candidate collision trajectories following the initial one, and outputs one with the highest physics-based score.

the self-generation of realistic trajectory rollouts, we adopt Markov Chain Monte Carlo (MCMC) sampling: our proposed sampling not only uses a novel parameterization of trajectories for controllable generation, but also effectively "tilts" the model distribution towards generating rare collision data based on energy-based model for physical scoring. For performance evaluation, we construct an evaluation set by synthesizing collision trajectories from NHTSA accident reports (National Highway Traffic Safety Administration, 2026). Experiments show that COLLIGEN substantially outperforms strong text-to-trajectory, trajectory-to-trajectory, and trajectory-prediction baselines in the reconstruction of collision trajectories. Qualitative studies also show that COLLIGEN is capable of generating realistic and physically plausible pre- and post-collision motion.

## 2 METHOD

**Overview.** The key challenge in the generation of collision trajectories is the limited availability of plausible collision samples. To address this issue, we present COLLIGEN, a two-phase self-training framework that progressively adapts a normal-driving trajectory model into a generator of collision trajectories. Our framework is model-agnostic and can be applied to any trajectory predictor that represents a distribution over future trajectories. An overview of COLLIGEN is illustrated in Fig. 1.

### 2.1 SELF-TRAINING FRAMEWORK

We assume that there are two driving vehicles: Vehicle 1 and Vehicle 2. A trajectory $\boldsymbol{\tau}$ is defined as a sequence of vehicle states over time steps, where we set the step $\Delta t$ to 0.1s. Initially, we have a base model that takes an initial (normal) trajectory $\boldsymbol{\tau}_0$ of $T_0$ seconds duration as input and generates a normal trajectory of duration $T$ seconds ($10T$ steps) following $\boldsymbol{\tau}_0$. The goal is to adapt the model to generate a collision trajectory that continues from $\boldsymbol{\tau}_0$ up to and after collision. We partition the trajectory generation around the collision time $t_c$ into the following two phases:

1. **Pre-Collision Phase ($\tau^{\text{pre}}$):** The trajectory during $[t_c - \Delta_{\text{pre}}, t_c)$, in which vehicles still exhibit near-normal driving behavior but transition towards a critical collision state.
2. **Post-Collision Phase ($\tau^{\text{post}}$):** The trajectory during $[t_c, t_c + \Delta_{\text{post}}]$, in which the motion is governed by impulsive contact forces and post-impact dynamics.

The duration $\Delta_{\text{pre}}$ of $\boldsymbol{\tau}^{\text{pre}}$ varies with the generated collision event but is at most $T$ seconds. The duration $\Delta_{\text{post}}$ of $\boldsymbol{\tau}^{\text{post}}$ is $T - \Delta_{\text{pre}}$, yielding the total duration $T$. Our phase decomposition is because the vehicle motion before impact is largely governed by normal driving dynamics, whereas that after impact is strongly influenced by collision physics. The final output is the trajectory $\boldsymbol{\tau}^{\text{pre}} \oplus \boldsymbol{\tau}^{\text{post}}$.

We adopt a self-learning framework, ReST Gulcehre et al. (2023), for both phases. Each phase consists of (i) *Grow* step constructing a dataset by self-generating trajectories for collision, and (ii)

**Algorithm 1:** COLLIGEN: Two-phase Self-Training with Physics-Guidance

**Input:** Base trajectory model parameter $\theta_0$ trained on normal driving data $\mathcal{D}_0$; initial trajectories $\mathcal{X}$; curriculum length $K = 3$; physical strictness scale (Appendix A.3) $\{\gamma_{\text{pre}}^{(k)}, \gamma_{\text{post}}^{(k)}\}_{k=1}^{K}$

**Output:** Collision-trajectory generation model $\theta_{\text{final}}$.

**Pre-Collision Phase: self-training on $\tau^{\text{pre}}$.**

$\theta_{\text{pre}}^{(0)} \leftarrow \theta_0$ `#initialize model`
**for** $k \leftarrow 1$ **to** $K$ **do**
  $\mathcal{A}_{\text{pre}}^{(k)} \leftarrow \emptyset$
  **foreach** *init. traj.* $\boldsymbol{\tau}_0 \in \mathcal{X}$ **do**
    $\mathcal{A} \leftarrow \text{MH_SAMPLE}(\boldsymbol{\tau}_0, \theta_{\text{pre}}^{(k-1)}, \gamma_{\text{pre}}^{(k)})$
    $\mathcal{A}_{\text{pre}}^{(k)} \leftarrow \mathcal{A}_{\text{pre}}^{(k)} \cup \mathcal{A}$
  $\mathcal{D}_{\text{pre}}^{(k)} \leftarrow \mathcal{D}_0 \cup \mathcal{A}_{\text{pre}}^{(k)}$ `#grow`
  $\theta_{\text{pre}}^{(k)} \leftarrow \text{argmin}_\theta \mathbb{E}_{\boldsymbol{\tau} \sim \mathcal{D}_{\text{pre}}^{(k)}}[\mathcal{L}(\boldsymbol{\tau}; \theta)]$ `#improve`

MH_SAMPLE $(\boldsymbol{\tau}_0, \theta, \gamma)$
**for** $k = 0$ **to** $N - 1$ **do**
  Generate candidate trajectory $\boldsymbol{\tau} \sim q_\theta(\cdot|\boldsymbol{\tau}_0)$
  $\alpha = \min\left(1, \frac{\pi_\gamma(\boldsymbol{\tau}) q_\theta(\boldsymbol{\tau}_k|\boldsymbol{\tau}_0)}{\pi_\gamma(\boldsymbol{\tau}_k) q_\theta(\boldsymbol{\tau}|\boldsymbol{\tau}_0)}\right)$
  Sample $u \sim \text{Uniform}(0, 1)$
  $\boldsymbol{\tau}_{k+1} \leftarrow \boldsymbol{\tau}$ **if** $(u \leq \alpha)$**; else** $\boldsymbol{\tau}_k$

**Post-Collision Phase: self-training on $\tau^{\text{post}}$.**

$\theta_{\text{post}}^{(0)} \leftarrow \theta_{\text{pre}}^{(K)}$ `#init model from Phase 1`
$\mathcal{X} \leftarrow \{\boldsymbol{\tau} \in \mathcal{A}_{\text{pre}}^{(K)} : \boldsymbol{\tau}$ has collision at $t^*\}$
**for** $k \leftarrow 1$ **to** $K$ **do**
  $\mathcal{A}_{\text{post}}^{(k)} \leftarrow \emptyset$
  **foreach** $\boldsymbol{\tau} \in \mathcal{X}$ **do**
    $\boldsymbol{\tau}_0 \leftarrow$ 20-step trajectory up to $t^*$ from $\boldsymbol{\tau}$
    $\mathcal{A} \leftarrow \text{MH_SAMPLE}(\boldsymbol{\tau}_0, \theta_{\text{post}}^{(k-1)}, \gamma_{\text{post}}^{(k)})$
    $\mathcal{A}_{\text{post}}^{(k)} \leftarrow \mathcal{A}_{\text{post}}^{(k)} \cup \mathcal{A}$
  $\mathcal{D}_{\text{post}}^{(k)} \leftarrow \mathcal{D}_0 \cup \mathcal{A}_{\text{post}}^{(k)}$ `#grow`
  $\theta_{\text{post}}^{(k)} \leftarrow \text{argmin}_\theta \mathbb{E}_{\boldsymbol{\tau} \sim \mathcal{D}_{\text{post}}^{(k)}}[\mathcal{L}(\boldsymbol{\tau}; \theta)]$ `#improve`
**return** $\theta_{\text{final}} \leftarrow \theta_{\text{post}}^{(K)}$

*Improve* step fine-tuning the model on the dataset. Post-collision phase begins after completing Pre-collision phase.

**Grow.** The model is initially (pre-)trained on the base dataset denoted by $\mathcal{D}_0$ which contains normal trajectories from, e.g., Argoverse Chang et al. (2019). At stage $k$ of self-training, the current model generates a candidate trajectory pool $\mathcal{A}^{(k)}$ of size $\alpha^{(k)}|\mathcal{D}_0|$. The newly generated data is merged with $\mathcal{D}_0$, i.e., $\mathcal{D}^{(k)} = \mathcal{D}_0 \cup \mathcal{A}^{(k)}$. The size parameter $\alpha^{(k)}$ of the generated dataset is scheduled over steps: see Appendix A.2 for details. Importantly, sampling in Grow step is regulated by physics-guided scoring and filtering, which will be explained in detail.

**Improve.** At stage $k$ of self-training, the Improve step is a fine-tuning of the model from stage $k-1$ on the dataset $\mathcal{D}^{(k)}$. Let $\theta$ denote the model parameter, then

$$\theta^{(k)} = \underset{\theta}{\text{argmin}}\, \mathbb{E}_{\boldsymbol{\tau} \sim \mathcal{D}^{(k)}}[\mathcal{L}(\boldsymbol{\tau}; \theta)], \quad \theta \text{ initialized from } \theta^{(k-1)} \tag{1}$$

where $\mathcal{L}$ is the loss of the base model, e.g., negative log-likelihood. The exact form of $\mathcal{L}$ varies with the base model. Some examples are provided in Appendix A.1. In Pre-collision phase, $\theta_0$ is the parameter (pre-)trained on $\mathcal{D}_0$. In Post-collision phase, $\theta_0$ is set to the model that completed Pre-collision phase.

The Grow–Improve steps are iteratively performed to fit the model to the desired distribution. In this paper, the total iteration denoted by $K$ is set to 3 for both Pre- and Post-collision training.

**Sampling Difficulty in Self-Training.** ReST is mainly developed for LLM self-improvement with a reward function. However, our problem domain is quite different: collisions never exist in the base normal-driving dataset. A model must generate and learn from extremely rare events in terms of the physical laws of observed trajectories. Sampling such rare events may take a long time or even fail. Our goal is to make a model *self-deviate* by gradually "tilting" its distribution towards rare events, such as collisions. To that end, we propose using Markov-Chain Monte Carlo (MCMC) Tierney (1994); specifically adopt Independent Metropolis-Hastings (IMH) Liu (1996) to gradually adapt the model to a rare-event distribution. Our framework is efficient not only in sampling of deviated trajectories but also in imposing physical laws on trajectory distribution. Experiments show that our method significantly speeds up self-training by more than 16× compared to randomized sampling. Algorithm 1 summarizes COLLIGEN, a two-phase self-training procedure with physics-guidance.

### 2.2 SAMPLING DEVIATED TRAJECTORIES VIA MCMC

The IMH algorithm uses a proposal distribution that is independent of the current sample Tierney (1994). We choose IMH over regular MH, because there exist several issues in sampling plausible trajectories, as explained later. We first explain the target and proposal densities for IMH.

**Target density.** For a generated trajectory $\boldsymbol{\tau}$, we define the *unnormalized* target density $\pi(\boldsymbol{\tau})$ in a typical form in MCMC:

$$\pi_\gamma(\boldsymbol{\tau}) = \exp\left(-\frac{E_\gamma(\boldsymbol{\tau})}{\mathcal{T}}\right), \tag{2}$$

where $E_\gamma(\boldsymbol{\tau})$ is the energy function based on physical constraints, and $\mathcal{T}$ is a temperature parameter. $E_\gamma(\boldsymbol{\tau})$ penalizes the trajectories violating physical laws. Eq. (2) encourages the sampling of low-energy trajectories. The threshold parameter $\gamma$ controls the strictness of the penalty for violation. We adopt various pre- and post-collision physics for realistic generation, as discussed in the next section.

**Proposal density.** The proposal density should generate diverse candidate trajectories for exploration. However, there exist challenges in trajectory sampling: (i) how to perform a randomized and controllable sampling of trajectories?; (ii) the sampled trajectories should be realistic; (iii) the proposal should admit a closed-form likelihood. We address these challenges by introducing a novel trajectory proposal based on *parametric curves* in vector graphics.

We first sample a trajectory $\boldsymbol{\tau}$ (possibly deterministic) from the base model. We fit $\boldsymbol{\tau}$ using a *quartic Bézier curve* Ebert et al. (1998) . The fitted curve is parameterized with 5 control points denoted by $P_0, ..., P_4$ where $P_0$ and $P_4$ are the start and end points of $\boldsymbol{\tau}$. Next, we randomly perturb some control points using a simple distribution $f$, e.g., Gaussian or Laplace. Let $\Delta \sim f$ denote the sampled 2D perturbation. $P_2$, $P_3$ and $P_4$ are gradually perturbed for smoothness as follows.

$$P_2^* = P_2 + g_2 \cdot \Delta, \quad P_3^* = P_3 + g_3 \cdot \Delta, \quad P_4^* = P_4 + \Delta$$

where $g_2 = 0.5$, $g_3 \approx 0.84$ are derived from the cubic Hermite interpolation Ebert et al. (1998) for smooth perturbation. We obtain new control points $(P_0, P_1, P_2^*, P_3^*, P_4^*)$ and construct a quartic Bézier curve with these points to produce the final trajectory. This procedure is shown in Fig. 2. See Appendix A.4 for a detailed construction.

Our candidate generation addresses the aforementioned issues: (i) a simple generation with well-known distributions; (ii) vehicle trajectories for short duration are likely to be well-fitted with quartic Bézier curves, leading to realistic outputs; (iii) the generation needs only one random element $\Delta$ whose likelihood is known in closed form. We also reveal a potential issue with using regular MH. If a candidate is sampled sequentially from the previous trajectory using regular MH, the trajectory initially generated from the base model must be perturbed iteratively. Such sequential fitting and error accumulation may lead to unrealistic trajectories. Thus, IMH is a more practically viable choice than regular MH.

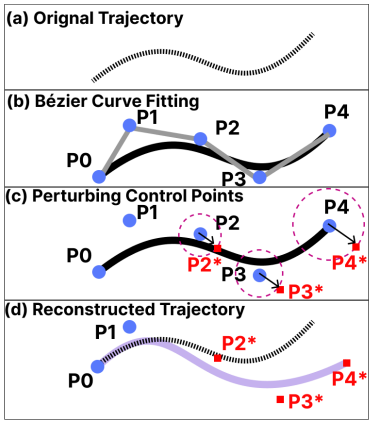



Fig 2: Candidate trajectory generation with quartic Bézier curves.

**Example: HiVT.** HiVT Zhou et al. (2022) is a trajectory generation model for normal driving. Given an initial trajectory $\boldsymbol{\tau}_0$ of 2 s duration, HiVT outputs six trajectories $\boldsymbol{\tau}_1, ..., \boldsymbol{\tau}_6$ of duration $T = 3$ s each and a categorical probability $(p_1, \ldots, p_6)$. We select a trajectory $\boldsymbol{\tau}_k$ with probability $p_k$, and let $\tilde{\boldsymbol{\tau}}$ be the candidate trajectory sampled from $\boldsymbol{\tau}_k$ using our method. The proposal density $q$ is a mixture:

$$q(\tilde{\boldsymbol{\tau}}|\boldsymbol{\tau}_0) = \sum_{k=1}^{6} p_k \cdot \text{Laplace}(\tilde{\boldsymbol{\tau}}(T); \boldsymbol{\tau}_k(T), b_k) \tag{3}$$

where Laplace denotes a 2D Laplace density, $\boldsymbol{\tau}(T)$ denotes the vehicle position at time $T$ or the end-point of the trajectory, and $b_k$ is the uncertainty parameter related to the variance. In HiVT, $b_k$ is a fitted parameter during training and is provided as a model output along with the trajectories. If such parameters are not available from the base model, the variance/uncertainty parameters can be set adaptively, as in Holden et al. (2009).

### 2.3 PHYSICS-GUIDED SCORING AND FILTERING

**Normalized Penalty Scoring.** The energy $E(\boldsymbol{\tau})$ is defined as the sum of penalty functions $\phi$, where each $\phi$ represents the penalty score for *residuals*. The residual is a nonnegative discrepancy between the expected physical motion and the generated motion. The residual also models physical plausibility, where zero (resp. positive) residual means plausible (resp. violation). For consistency and simplicity, we adopt a unified form of penalty function that maps residuals to penalty scores:

$$\phi_\gamma(r) = \text{clip}\left(\frac{r - \gamma\mu}{\sigma}, 0, 1\right), \quad \text{clip}(x, a, b) := \min[\max(x, a), b] \tag{4}$$

where $r$ denotes the residual of the motion of interest; $\mu$ and $\sigma^2$ represent the empirical mean and variance of $r$ measured on the dataset from the preceding self-training stage. $\gamma \in [0, 1]$ is a hyperparameter representing the stage-dependent physical strictness scale. The basic property of penalty $\phi_\gamma(r)$ is that it is increasing in $r$, i.e., a higher penalty for a higher residual because we would like the residual to be small. Additional properties of our penalty score are explained as follows.

*1) Normalized score.* Later, we define the residuals of various physical quantities for collisions, e.g., time, distance, momentum, etc. $\phi_\gamma(r)$ is dimensionless irrespective of the physical nature of $r$ because it is normalized with $\mu$ and $\sigma$. We apply $\phi_\gamma$ to all the physical residuals as a unified form of penalty scores, which significantly simplifies our design.

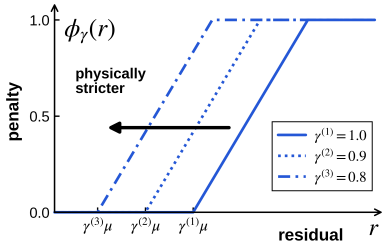



Fig 3: Plot of $\phi_\gamma(r)$ over $r$ with changing $\gamma$. $\gamma^{(i)}$ is the value of $\gamma$ scheduled at $i$-th self-training stage.

*2) Controllable physical strictness.* We adjust $\gamma \in [0, 1]$ to control how severely $\phi_\gamma$ penalizes residuals. For fixed $r$, $\phi_\gamma$ yields a higher penalty with a smaller $\gamma$ (see Fig. 3). $\gamma$ is scheduled to decrease over self-training stages, and the model evolves to penalize residuals/errors more severely. Thus, stricter physical criteria are applied to learning in later stages. Another view is that $\phi_\gamma$ imposes *zero* penalty if the residual is less than its current mean $\mu$ scaled by $\gamma$. Thus, $\gamma\mu$ is an effective *threshold* for zero penalty. A lower threshold for residuals means stricter criteria for zero penalty. The schedule of $\gamma$ is discussed in Appendix A.3.

$E(\boldsymbol{\tau})$ is defined separately for Pre- and Post-collision phases as follows.

**Phase 1: Pre-collision scoring.** Given a pre-collision trajectory $\boldsymbol{\tau}$, we evaluate five residuals and their penalty scores. The residuals are defined in two groups. *Group 1: basic motion (distance, time).* The residuals are: $r_{\text{dist}}$ penalizes an implausibly long distance between two vehicles for collision; $r_{\text{ttc}}$ penalizes excessive time-to-collision. *Group 2: motion irregularity.* The residuals are: $r_{\text{acc}}, r_{\text{jerk}}$, and $r_{\text{yaw}}$ penalize implausibly irregular motion in acceleration, jerk, and yaw rate, respectively. The energy is set to the mean of penalty scores (4) applied to the residuals:

$$E_\gamma(\boldsymbol{\tau}) = \frac{1}{5}\left[\phi_\gamma(r_{\text{dist}}(\boldsymbol{\tau})) + \phi_\gamma(r_{\text{ttc}}(\boldsymbol{\tau})) + \phi_\gamma(r_{\text{acc}}(\boldsymbol{\tau})) + \phi_\gamma(r_{\text{jerk}}(\boldsymbol{\tau})) + \phi_\gamma(r_{\text{yaw}}(\boldsymbol{\tau}))\right] \tag{5}$$

We explain $r_{\text{dist}}$ as an example. To compute $r_{\text{dist}}(\boldsymbol{\tau})$, we first find the time step $t^*$ in $\boldsymbol{\tau}$ where two vehicles are closest, and denote the position and velocity of vehicle $i$ at $t^*$ by $\mathbf{x}_i$ and $\mathbf{v}_i$ for $i = 1, 2$. We evaluate the feasibility of collision within time $\Delta t_{\text{pre}}$ which denotes the remaining time of Pre-collision phase:

$$r_{\text{dist}}(\boldsymbol{\tau}) = \max\left(0, \|\mathbf{x}_1 - \mathbf{x}_2\|_2 - (\|\mathbf{v}_1\|\Delta t_{\text{pre}} + \|\mathbf{v}_2\|\Delta t_{\text{pre}})\right). \tag{6}$$

A positive $r_{\text{dist}}$ implies that the vehicles are too far away to collide within Pre-collision phase. Thus, $r_{\text{dist}}$ penalizes the physical implausibility of the occurrence of a collision.

The other residuals for Pre-collision phase are defined in Appendix A.5.

**Phase 2: Post-collision scoring.** We consider six residuals for scoring the post-collision trajectories. They are divided into two groups. *Group 1: post-impact kinetics.* We define residuals $r_{\text{mnt}}$, $r_{\text{norm}}$, and $r_{\text{tan}}$, which are momentum consistency, normal-direction velocity error, and tangential-direction velocity error, respectively. *Group 2: motion irregularity.* We define residuals $r_{\text{acc}}$, $r_{\text{jerk}}$, and $r_{\text{yaw}}$, for acceleration, jerk, and yaw rate, respectively.

We explain $r_{\text{mnt}}$ as an example. $r_{\text{mnt}}$ evaluates the momentum consistency under ground friction. We first consider the magnitude of ground-friction impulse $\mathbf{J}_{\text{fric}}$ Meriam et al. (2020) at some time step:

$$\|\mathbf{J}_{\text{fric}}\|_2 = \min\left(\mu_{\text{gnd}}(m_1 + m_2)g\Delta t, \left\|\mathbf{p}^-\right\|_2\right), \quad \mathbf{p}^- := \mathbf{p}_1^- + \mathbf{p}_2^- \tag{7}$$

where $m_1, m_2$ are vehicle masses, $\mu_{\text{gnd}} \approx 0.7$ is the ground-friction coefficient, $g$ is gravitational constant, $\Delta t$ is the step duration, and $\mathbf{p}_i^-$ is the momentum of vehicle $i$ from the previous time step. The frictional impulse is applied opposite to the total momentum $\mathbf{p}^-$. Thus, $\mathbf{J}_{\text{fric}} = -\left\|\mathbf{J}_{\text{fric}}\right\|_2 \frac{\mathbf{p}^-}{\|\mathbf{p}^-\|_2}$. Let $\mathbf{p}^+_{\text{exp}}$ be the total momentum expected from the physical laws at the current time step accounting for ground friction. We then have $\mathbf{p}^+_{\text{exp}} = \mathbf{p}^- + \mathbf{J}_{\text{fric}}$. Let $\mathbf{p}^+_{\text{pred}}$ be the total momentum measured from the generated trajectory at the current time step. We define the relative momentum residual as

$$\frac{\left\|\mathbf{p}^+_{\text{exp}} - \mathbf{p}^+_{\text{pred}}\right\|_2}{\left\|\boldsymbol{p}_1^-\right\|_2 + \left\|\boldsymbol{p}_2^-\right\|_2 + \epsilon}, \qquad \epsilon = 1.0 \times 10^{-6}. \tag{8}$$

We compute the residual Eq. (8) for all the time steps in the post-collision trajectory $\boldsymbol{\tau}$, and set their maximum value as the final residual $r_{\text{mnt}}(\boldsymbol{\tau})$.

The other residuals are defined in Appendix A.6. The penalty score Eq. (4) is applied to the residuals, and the scores are averaged over each of the post-impact kinetics and motion irregularity groups, i.e.,

$$\phi_\gamma^{\text{imp}}(\boldsymbol{\tau}) = \frac{1}{3}\left[\phi_\gamma(r_{\text{mnt}}(\boldsymbol{\tau})) + \phi_\gamma(r_{\text{norm}}(\boldsymbol{\tau})) + \phi_\gamma(r_{\text{tan}}(\boldsymbol{\tau}))\right] \tag{9}$$

$$\phi_\gamma^{\text{irr}}(\boldsymbol{\tau}) = \frac{1}{3}\left[\phi_\gamma(r_{\text{acc}}(\boldsymbol{\tau})) + \phi_\gamma(r_{\text{jerk}}(\boldsymbol{\tau})) + \phi_\gamma(r_{\text{yaw}}(\boldsymbol{\tau}))\right] \tag{10}$$

The final $E_\gamma$ penalizes $\boldsymbol{\tau}$ if it violates either post-impact kinetics or motion irregularity:

$$E_\gamma(\boldsymbol{\tau}) = \max\left(\phi_\gamma^{\text{imp}}(\boldsymbol{\tau}), \phi_\gamma^{\text{irr}}(\boldsymbol{\tau})\right), \tag{11}$$

**Filtering.** In Pre-collision phase, there is no sample filtering during Grow step. However, when the final dataset from Pre-collision phase is passed to Post-collision Phase, the trajectories that do not contain collisions are filtered out. In the Post-collision Phase, filtering is done at every Grow step; only the trajectories that satisfy $\phi_\gamma^{\text{imp}}(\cdot) = 0$ in Eq. (9) remain at each step. This ensures that only the trajectories that (nearly) obey post-impact physics remain in the dataset.

### 2.4 Inference

For inference, we obtain the output trajectories from an initial trajectory of duration $T_0 = 2$ s. If the base model outputs multiple trajectories (e.g., 6 for HiVT as in Fig. 1), we compute the sum of energy functions $E_\gamma$, representing the overall penalty score, from Pre- and Post-collision phases. Finally, the trajectory with the minimum score, which is likely the most physically plausible, is selected.

## 3 Experiments

### 3.1 Setup

**Baselines.** We compare our method against trajectory prediction baselines: LaneGCN Liang et al. (2020), HiVT Zhou et al. (2022); text-to-trajectory models: InteractTraj Xia et al. (2024) and LCTGen Tan et al. (2023); trajectory-to-trajectory models: TrafficGen Feng et al. (2022) and HPNet Tang et al. (2024); collision generation models: KING Hanselmann et al. (2022). COLLIGEN uses HiVT and QCNet Zhou et al. (2023) as base models. Implementation details are in Appendix A.7.

**Training Data.** We use Argoverse (Chang et al., 2019) as the normal-driving training dataset, and use the same data configurations as the baseline models.

**Evaluation Data.** We evaluate models on vehicle collision data. To our knowledge, there are no real-world datasets that contain full pre- and post-collision trajectories. Thus, we construct a collision dataset based on the 2017–2024 NHTSA Crash Investigation Sampling System (CISS) records (National Highway Traffic Safety Administration, 2026). The NHTSA records provide information on crash events, vehicle damage, and event data recorder (EDR) reports, including impact heading, pre-impact speed, and steering histories. However, the NHTSA data is incomplete in that (i) it is mostly manually collected, containing erroneous or missing entries; (ii) post-impact records are sometimes missing or span only a short period of time ($\leq 0.3$ s). Thus, we construct pre- and post-collision trajectories from the available NHTSA data using simulators as follows.

Table 1: Quantitative comparison of generated collision trajectories.

| Method | | PreADE ↓ | PreFDE ↓ | PostADE ↓ | PostFDE ↓ | rel_dist_err ↓ |
|---|---|---|---|---|---|---|
| **Trajectory Prediction** | LaneGCN | 3.367 | 2.569 | 6.578 | 12.645 | 5.144 |
| | HiVT | 3.245 | 2.452 | 6.111 | 11.563 | 4.931 |
| | HPNet | 3.542 | 2.763 | 6.371 | 12.005 | 5.206 |
| **Traj-to-Traj** | TrafficGen | 3.380 | 2.565 | 6.456 | 12.300 | 5.098 |
| **Text-to-Trajectory** | LCTGen | 4.018 | 3.319 | 6.669 | 12.132 | 6.058 |
| | InteractTraj | 6.349 | 4.804 | 9.406 | 18.085 | 9.806 |
| **Collision Generation** | KING | 2.600 | 2.512 | 3.480 | 4.968 | 4.086 |
| **Ours** | **Ours (HiVT)** | **1.706** | **1.593** | **1.487** | **2.138** | **2.007** |
| | **Ours (QCNet)** | **1.705** | **1.681** | **1.610** | **2.254** | **2.333** |

We first reconstruct the pre-collision trajectories on a flat OpenDRIVE map (ASAM e.V., 2021) based on NHTSA data. The reconstructed pre-collision trajectories are independently replayed in two simulators to generate post-collision trajectories: CARLA 0.9.15 (Dosovitskiy et al., 2017) and Chrono (Tasora et al., 2016). Chrono is a general-purpose multiphysics simulation framework, of which we use the Chrono::Vehicle module designed specifically for vehicle simulation. Each simulator is initialized with the reconstructed vehicle states before impact to generate the subsequent collision dynamics and post-impact trajectories.

Next, the post-collision trajectory generated by each simulator is validated against the partial post-collision data from NHTSA using correlation analysis. Finally, between two trajectories generated by the simulators, the one that more closely matches the NHTSA data is selected. A total of 4,583 accident samples were generated using this procedure. Among the selected samples, 76.4% were generated by CARLA and 23.6% by Chrono. Further details on the construction, validation, and selection of evaluation data are provided in Appendix A.8.

**Evaluation Protocol and Lanes.** We use a common evaluation protocol for all methods. $t$ denotes the collision time, or the time at which the two vehicles are closest. The model input is the ground-truth trajectory segment from $t-3$ to $t-1$ seconds, and the target output is the 30-step segment from $t-1$ to $t+2$ seconds. Baselines such as LaneGCN or HiVT need *lane* information, where lanes serve as soft guidance for the model to generate future waypoints. However, NHTSA provides only partial information on lanes/maps of accident sites; instead, it provides rough schematics of road maps and trajectories. We estimate the lanes and incorporate them into evaluation samples as follows: based on initial vehicle trajectories, we estimate the lanes by extrapolating the trajectories. In cases where the NHTSA schematic is available, we combine the schematic and the estimated lanes to construct a rough road map. We visualize such road maps in our generation samples, as shown later. Additional implementation details of lanes/maps are provided in Appendix A.9.

**Evaluation Metrics.** Trajectory accuracy is evaluated using metrics PreADE and PreFDE, which measure the average and final displacement errors in the pre-collision interval; PostADE and PostFDE, which measure the corresponding errors in the post-collision interval; the relative distance error to quantify pairwise geometric consistency between the generated trajectories and the ground truth.

### 3.2 Results

**Quantitative Results.** Table 1 shows that COLLIGEN models consistently outperform the baselines in accident trajectory generation. The results show that our physics-guided self-training framework successfully adapts normal-driving trajectory models to the collision domain. The baseline methods had difficulties, especially in generating post-collision trajectories, as shown by PostADE and PostFDE. Importantly, the text-to-trajectory models, such as InteractTraj and LCTGen, show limited performance, although they are explicitly prompted to generate collisions according to the accident report. KING performs better than the other baselines. However, the KING model is limited to reconstruction only up to the moment of impact, which limits its post-collision performance. Overall, our results show the importance of the model's understanding of the physics behind collisions.

**Qualitative Results.** Figure 4 presents qualitative comparisons of generated trajectories. As shown in Figure 4, existing baselines often fail to reproduce the intended collision behavior. Some baselines produce physically implausible post-impact motion, wherein the vehicles pass through each other

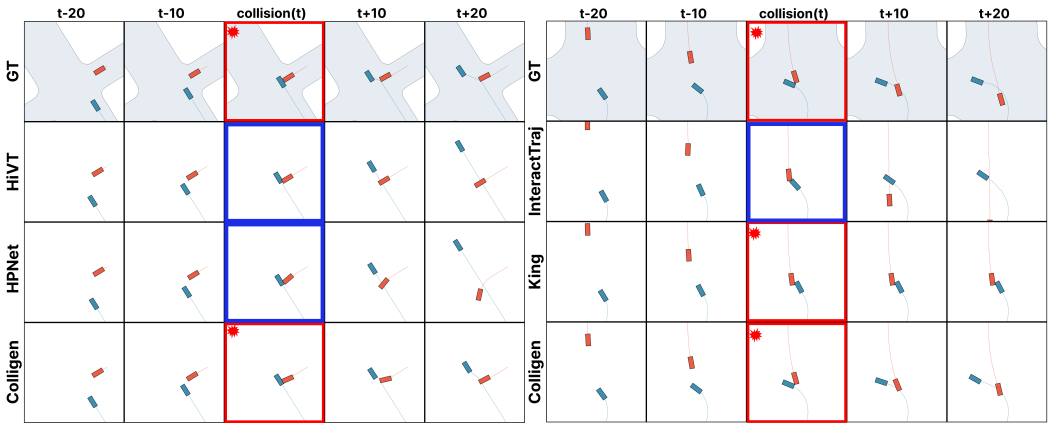



Fig 4: Qualitative comparison of COLLIGEN with baselins such as HiVT, HPNet, InteractTraj, KING, and ground truth reconstructed by CARLA/Chrono simulators from NHTSA accident reports. Red squares show the moments of impact, where the vehicle trajectories change afterwards. Blue square shows an overlap, not collision, because vehicles pass through each other. Road maps are shown only in GT for clarity of trajectory comparison.

Table 2: Ablation study on pre- / post-collision physics scoring.

| Variant | PreADE ↓ | PreFDE ↓ | PostADE ↓ | PostFDE ↓ | rel_dist_err ↓ |
|---|---|---|---|---|---|
| Pre Only | 3.019 | 4.580 | 7.221 | 9.024 | 6.916 |
| Pre + Post w/o $\phi^{\text{irr}}$ | 3.256 | 5.050 | 6.783 | 7.877 | 6.027 |
| Pre + Post w/o $\phi^{\text{imp}}$ | 2.983 | 4.797 | 8.433 | 10.973 | 7.401 |
| Pre + Post Scoring | **1.706** | **1.593** | **1.487** | **2.138** | **2.007** |

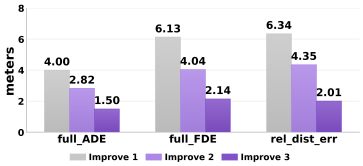



Fig 5: Performance improvement with iterative Improve steps.

after contact. In KING, the vehicles collide but remain still after the collision. In contrast, our model generates trajectories that both realize the intended collision event and remain consistent after impact. The generated post-collision motion is also visually closer to the ground-truth trajectory, suggesting that our physics-guided self-training framework improves not only collision occurrence but also post-impact trajectory plausibility. Additional qualitative results are provided in Appendix A.11.

**Self-training Results.** Fig. 5 shows the performance of COLLIGEN after completing each Improve step during the Post-collision phase. The metrics full_ADE and full_FDE denote the ADE and FDE measured over the entire pre-collision and post-collision period. We observe that the performance improves with the progress of the Improve steps. Interestingly, the largest performance increase occurs in the final Improve step. This is because the physical strictness scale $\gamma^{(k)}$ is the smallest $(k = 3)$ in the last Grow step. Our framework is the most strict in filtering out samples with poor physical plausibility, leading to a large improvement in physical accuracy.

**Ablation: effect of pre-collision and post-collision scoring.** Table 2 shows the ablation on Pre- and Post-collision Scoring. Removing either component of the post-collision scores degrades performance. The variant without penalty $\phi^{\text{irr}}$ in (10) on motion irregularity shows that acceleration, jerk, and yaw-rate regularity are necessary for suppressing dynamically unrealistic post-impact motion. The result on the variant without $\phi^{\text{imp}}$ in (9) confirms the importance of physical consistency based on post-impact kinetics. Overall, this result suggests that the post-impact kinetics constraint is particularly important for producing physically plausible collision trajectories.

**Sampling methods.** We compare methods of trajectory sampling: random sampling with Monte Carlo (MC) Dropout Gal & Ghahramani (2016), Gaussian waypoint perturbation (GWP), and the proposed IMH. Sampling by MC dropout performs random deactivation of nodes in the trajectory encoder, and GWP

Table 3: Runtime comparison of sampling.

| Sampling strategy | Norm. time ↓ |
|---|---|
| Random (MC Dropout) | 16.1 |
| Gaussian waypoint perturb. | 18.9 |
| IMH | **1.0** |

sampling adds Gaussian noise to waypoints of a given trajectory. Table 3 compares the relative latency of collecting 50 valid samples. Relative to our IMH sampling, MC Dropout takes 16.1× and GWP takes 18.9× longer to complete. The result suggests that randomly generated or perturbed trajectories tend to be physically implausible and subsequently filtered out, taking it longer to obtain valid samples. This is an important distinction from the original ReST framework which uses random softmax sampling for rollouts and external reward models.

**Additional Experiments.** Additional experiments are provided in the Appendix as follows. The selection of scheduling parameters $\alpha$ (A.2) and $\gamma$ (A.3); extension to three-vehicle collisions (A.10); some failure cases of generation (A.12).

## 4 Related Work

**Trajectory Prediction and Traffic Scenario Generation.** Trajectory prediction methods such as LaneGCN (Liang et al., 2020), HiVT (Zhou et al., 2022), and HPNet (Tang et al., 2024) model map context and agent interactions effectively on large-scale benchmarks including Argoverse (Chang et al., 2019) and Waymo (Ettinger et al., 2021). However, these datasets are dominated by nominal driving. Traffic scenario generators such as TrafficGen (Feng et al., 2022) extend trajectory modeling to multi-agent scene synthesis, but primarily target the distribution of common traffic rather than rare impacts or post-collision motion. Long-tail forecasting methods such as CDKFormer (Lian et al., 2026) remain focused on predicting rare behaviors rather than explicitly generating collisions.

**Controllable and Safety-Critical Scenario Generation.** Language-conditioned generators improve controllability by mapping text to traffic scenes or trajectories, as in InteractTraj (Xia et al., 2024), LCTGen (Tan et al., 2023), LangTraj (Chang et al., 2025), Trajectory-LLM (Yang et al., 2025), and Talk2Traffic (Sheng et al., 2025). However, these methods are mostly trained on normal-driving corpora and typically represent accidents through templates, semantic conditions, or coarse controls rather than explicitly modeling collision dynamics. Datasets such as DeepAccident (Wang et al., 2024) and TUM Traffic Accid3nD (Zimmer et al., 2025) focus on safety-critical and accident-specific data. However, DeepAccident covers pre-collision motion and the collision event but does not provide post-collision trajectory supervision. TUM Traffic Accid3nD contains real-world accident recordings, but its publicly released dataset comprises only 12 accident sequences, limiting its scale for learning diverse collision dynamics. Report-driven and collision-aware supervision methods (Zhang et al., 2025; Patrikar et al., 2025; Wang et al., 2025) further demonstrate the value of accident-specific data. However, these datasets and methods do not directly address trajectory-level accident generation for self-training with both feasible pre-collision motion and physically consistent post-collision dynamics. Similarly, KING (Hanselmann et al., 2022) can generate trajectories leading up to collisions but does not model physically consistent post-collision dynamics.

**Physics-Guided Generation and Self-Training.** Physics-informed learning introduces physical structure into data-driven models (Raissi et al., 2019; Karniadakis et al., 2021), and prior motion-generation work has shown that incorporating kinematic constraints can improve trajectory realism. Meanwhile, iterative self-training methods such as ReST (Gulcehre et al., 2023), STAR (Zelikman et al., 2022), and RAFT (Dong et al., 2023) improve models through repeated generation, filtering, and fine-tuning. Our method combines these directions for realistic accident trajectory synthesis: we use physics-guided scoring to filter sampled trajectories for self-training, progressively adapting a predictor trained on normal driving into a collision-aware trajectory generator capable of modeling both pre-impact and post-impact motion.

## 5 Conclusion

We presented COLLIGEN, a physics-guided self-learning framework for generating collision trajectories in settings where real collision data are substantially scarcer than normal-driving data. By progressively adapting a normal-driving trajectory model through pre-collision and post-collision self-training, COLLIGEN produces trajectories that are more consistent with collision feasibility and post-impact physical constraints. Our framework may serve as a self-learning methodology for generating training and evaluation data for autonomous-driving systems in safety-critical scenarios. Future work includes incorporating more detailed physical modeling, maps with obstacles, and complex interactions among multiple vehicles into our generative model.

### AI USE STATEMENT

Generative AI tools were used to assist with editing the manuscript for clarity and readability, LaTeX formatting, and limited code editing. They were not used to formulate the research hypotheses, design the proposed methodology or experiments, or interpret the experimental results. All AI-assisted code was reviewed and tested by the authors, and all AI-assisted manuscript edits were manually reviewed for correctness. The authors take full responsibility for the final content of the paper, including all text, claims, code, and artifacts produced with the aid of generative AI.

### REPRODUCIBILITY STATEMENT

We provide the following resources to support the full reproducibility of our work.

- **Code.** An anonymized code is provided with this submission. The code includes the implementation of COLLIGEN and the scripts used for training and inference.
- **Dataset.** We constructed a dataset containing 4,583 collision trajectories based on NHTSA data (outlined in Sec. 3.1). We believe the dataset to be an important contribution to the research community. We will publicly release the dataset upon acceptance to support reproducibility and future research.

Additional details for reproducibility are provided in Section 2 including pseudocode in Algorithm 1 for the proposed method, Section 3.1 for the experimental setup and evaluation protocol, and Appendices A.1–A.3 for training details. Additional implementation details on trajectory reconstruction, physical residuals, evaluation-data preparation, baselines, and map reconstruction are provided in the corresponding appendices.

## REFERENCES

ASAM e.V. ASAM OpenDRIVE - road network reference specification, 2021. URL `https://asam.net`. Accessed: 2026-09-07.

Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8740–8749. IEEE, 2019.

Wei-Jer Chang, Wei Zhan, Masayoshi Tomizuka, Manmohan Chandraker, and Francesco Pittaluga. Langtraj: Diffusion model and dataset for language-conditioned trajectory simulation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 26622–26631. IEEE, 2025.

Wenhao Ding, Chejian Xu, Mansur Arief, Haohong Lin, Bo Li, and Ding Zhao. A survey on safety-critical driving scenario generation—a methodological perspective. *IEEE Transactions on Intelligent Transportation Systems*, 24(7):6971–6988, 2023.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*, 2023.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pp. 1–16. PMLR, 2017.

David S Ebert, F Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steven Worley. *Texturing and modeling*. Elsevier, 1998.

Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9710–9719, 2021.

Roy Featherstone. *Rigid body dynamics algorithms*. Springer, 2008.

Lan Feng, Quanyi Li, Zhenghao Peng, Shuhan Tan, and Bolei Zhou. Trafficgen: Learning to generate diverse and realistic traffic scenarios. *arXiv preprint arXiv:2210.06609*, 2022.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. Reinforced self-training (rest) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023.

Niklas Hanselmann, Katrin Renz, Kashyap Chitta, Apratim Bhattacharyya, and Andreas Geiger. King: Generating safety-critical driving scenarios for robust imitation via kinematics gradients. In *European Conference on Computer Vision*, pp. 335–352. Springer, 2022.

Lars Holden, Ragnar Hauge, and Marit Holden. Adaptive independent metropolis–hastings. *The Annals of Applied Probability*, 19(1):395–413, 2009. doi: 10.1214/08-AAP545.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Yuansheng Lian, Ke Zhang, and Meng Li. Cdkformer: Contextual deviation knowledge-based transformer for long-tail trajectory prediction. *Transportation Research Part C: Emerging Technologies*, 183:105430, 2026.

Ming Liang, Bin Yang, Yun Chen, Rongjie Hu, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *ECCV*, 2020.

Jun S Liu. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and computing*, 6(2):113–119, 1996.

James L Meriam, L Glenn Kraige, and Jeff N Bolton. *Engineering mechanics: dynamics*. John Wiley & Sons, 2020.

National Highway Traffic Safety Administration. NHTSA Crash Viewer. `https://crashviewer.nhtsa.dot.gov/`, 2026. Accessed: 2026-03-08.

Jay Patrikar, Apoorva Sharma, Sushant Veer, Boyi Li, Sebastian Scherer, and Marco Pavone. The case for negative data: From crash reports to counterfactuals for reasonable driving. *arXiv preprint arXiv:2509.18626*, 2025.

William H Press and Saul A Teukolsky. *Numerical recipes with source code CD-ROM 3rd edition: the art of scientific computing*. Cambridge university press, 2007.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Peter H Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

Zihao Sheng, Zilin Huang, Yansong Qu, Yue Leng, and Sikai Chen. Talk2traffic: Interactive and editable traffic scenario generation for autonomous driving with multimodal large language model. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 3788–3797. IEEE, 2025.

Shuhan Tan, Boris Ivanovic, Xinshuo Weng, Marco Pavone, and Philipp Kraehenbuehl. Language conditioned traffic generation. *arXiv preprint arXiv:2307.07947*, 2023.

Xiaolong Tang, Meina Kan, Shiguang Shan, Zhilong Ji, Jinfeng Bai, and Xilin Chen. Hpnet: Dynamic trajectory forecasting with historical prediction attention. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15261–15270. IEEE, 2024.

Alessandro Tasora, Radu Serban, Hammad Mazhar, Arman Pazouki, Daniel Melanz, Jonathan Fleischmann, Michael Taylor, Hiroyuki Sugiyama, and Dan Negrut. Chrono: An open source framework for the physics-based simulation of dynamic systems. In *High Performance Computing in Science and Engineering*, volume 9611 of *Lecture Notes in Computer Science*, pp. 19–49. Springer, 2016.

Luke Tierney. Markov chains for exploring posterior distributions. *the Annals of Statistics*, pp. 1701–1728, 1994.

Tianqi Wang, Sukmin Kim, Ji Wenxuan, Enze Xie, Chongjian Ge, Junsong Chen, Zhenguo Li, and Ping Luo. Deepaccident: A motion and accident prediction benchmark for v2x autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 5599–5606, 2024.

Yu Wang and Matthew T. Mason. Two-dimensional rigid-body collisions with friction. *Journal of Applied Mechanics*, 59(3):635–642, 1992. doi: 10.1115/1.2893771.

Zi Wang, Shiyi Lan, Xinglong Sun, Nadine Chang, Zhenxin Li, Zhiding Yu, and Jose M Alvarez. Enhancing autonomous driving safety with collision scenario integration. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10116–10123. IEEE, 2025.

Junkai Xia, Chenxin Xu, Qingyao Xu, Yanfeng Wang, and Siheng Chen. Language-driven interactive traffic trajectory generation. *Advances in Neural Information Processing Systems*, 37:77831–77859, 2024.

Chejian Xu, Wenhao Ding, Weijie Lyu, Zuxin Liu, Shuai Wang, Yihan He, Hanjiang Hu, Ding Zhao, and Bo Li. Safebench: A benchmarking platform for safety evaluation of autonomous vehicles. *Advances in Neural Information Processing Systems*, 35:25667–25682, 2022.

Kairui Yang, Zihao Guo, Gengjie Lin, Haotian Dong, Zhao Huang, Yipeng Wu, Die Zuo, Jibin Peng, Ziyuan Zhong, Xin Wang, et al. Trajectory-llm: A language-based data generator for trajectory prediction in autonomous driving. In *International Conference on Learning Representations*, volume 2025, pp. 43442–43454, 2025.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

Xiangwen Zhang, Qian Zhang, Longfei Han, Qiang Qu, and Xiaoming Chen. Accidentsim: Generating physically realistic vehicle collision videos from real-world accident reports. *arXiv preprint arXiv:2503.20654*, 2025.

Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. HiVT: Hierarchical vector transformer for multi-agent motion prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8813–8823, 2022.

Zikang Zhou, Jianping Wang, Yung-Hui Li, and Yu-Kai Huang. Query-centric trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 17863–17873, 2023.

Walter Zimmer, Ross Greer, Xingcheng Zhou, Rui Song, Hu Cao, Daniel Lehmberg, Marc Pavel, Ahmed Alaaeldin Ghita, Akshay Gopalkrishnan, Holger Caesar, et al. Towards vision zero: The tum traffic accid3nd dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 852–862, 2025.

# A APPENDIX

## A.1 LOSS FUNCTIONS OF BASE MODELS

We list the loss functions of the following base models. For HiVT Zhou et al. (2022), the negative log-likelihood loss used by HiVT can be written as

$$\mathcal{L}_{\mathrm{NLL}} = -\frac{1}{NH}\sum_{i=1}^{N}\sum_{t=T+1}^{T+H}\log P\Big(\mathbf{R}_i^{\top}\left(\mathbf{p}_i^t - \mathbf{p}_i^T\right)\Big|\,\hat{\boldsymbol{\mu}}_i^t, \hat{\mathbf{b}}_i^t\Big). \tag{12}$$

where $N$ is the number of agents, $H$ is the prediction step, and $T$ is the number of observed historical time steps. The index $i$ denotes the $i$-th agent, and $t$ denotes a future time step from $T+1$ to $T+H$. $\mathbf{p}_i^t \in \mathbb{R}^2$ is the ground-truth position of agent $i$ at time step $t$, while $\mathbf{p}_i^T$ is its position at the current time step. $\mathbf{R}_i \in \mathbb{R}^{2\times 2}$ is the rotation matrix of agent $i$'s local coordinate frame. Thus, $\mathbf{R}_i^{\top}(\mathbf{p}_i^t - \mathbf{p}_i^T)$ represents the future displacement of agent $i$ transformed into its local coordinate frame. $P(\cdot \mid \cdot)$ denotes the probability density function of the Laplace distribution. $\hat{\boldsymbol{\mu}}_i^t$ and $\hat{\mathbf{b}}_i^t$ are the predicted location and uncertainty, respectively, of the best-predicted trajectory for agent $i$ at time step $t$. For more details on the loss, see Zhou et al. (2022).

For QCNet Zhou et al. (2023), the training objective is

$$\mathcal{L}_{\mathrm{QCNet}} = \mathcal{L}_{\mathrm{reg}}^{\mathrm{propose}} + \mathcal{L}_{\mathrm{reg}}^{\mathrm{refine}} + \mathcal{L}_{\mathrm{cls}}$$

where $\mathcal{L}_{\mathrm{reg}}^{\mathrm{propose}}$ and $\mathcal{L}_{\mathrm{reg}}^{\mathrm{refine}}$ denote the negative log-likelihood regression losses for the proposal and refinement stages, respectively, and $\mathcal{L}_{\mathrm{cls}}$ denotes the mixture negative log-likelihood term used to score the refined modal distribution. For example, $\mathcal{L}_{\mathrm{reg}}^{\mathrm{propose}}$ is given by

$$\mathcal{L}_{\mathrm{reg}}^{\mathrm{propose}} = -\log\left[\sum_k \pi_{i,k}\prod_t \mathrm{Laplace}(\mathbf{p}_i^t|\boldsymbol{\mu}_{i,k}^t, \mathbf{b}_{i,k}^t)\right]$$

with similar definitions to HiVT. For more details on the loss, see Zhou et al. (2023).

## A.2 DATA MERGE SCHEDULE

At the $k$-th curriculum stage, we construct the fine-tuning dataset by merging the original normal-driving data $\mathcal{D}_0$ and the accepted pseudo-collision data $\mathcal{A}_k$. We sample ∼6K trajectories from Argoverse Chang et al. (2019) to construct the normal driving dataset $\mathcal{D}_0$. The size of $\mathcal{A}_k$ is controlled by parameter $\alpha^{(k)}$, where $|\mathcal{A}_k|$ is set to $\alpha^{(k)}|\mathcal{D}_0|$. The stage-wise merge ratio is defined as

$$|\mathcal{D}_0| : |\mathcal{A}_k| = \begin{cases} 4:1, & k=1,\ \alpha^{(1)} = 0.25, \\ 4:2, & k=2,\ \alpha^{(2)} = 0.5, \\ 4:3, & k=3,\ \alpha^{(3)} = 0.75. \end{cases} \tag{13}$$

We use a larger proportion of $\mathcal{D}_0$ than $\mathcal{A}_k$ throughout training because the quality of the self-generated pseudo-collision samples is not fully guaranteed at early stages. If $\mathcal{A}_k$ dominates the fine-tuning set, the model can become unstable and the convergence speed may decrease due to noisy or physically imperfect pseudo-labels. Therefore, we keep relatively reliable normal-driving data $\mathcal{D}_0$ as the major component, while gradually increasing the contribution of $\mathcal{A}_k$ as the scoring criteria become stricter across curriculum stages.

Next, we examine the selection of stage-wise merge schedule $\alpha$ in Table 4. The constant schedule $(1, 1, 1)$ introduces pseudo-collision data at the same ratio across all grow steps, leading to slower improvement. The aggressive schedule $(0.75, 0.5, 0.25)$ instead places excessive emphasis on pseudo-collision data at early stages, where reliable collision samples are still difficult to obtain, resulting in unstable convergence and inferior performance. Our progressive schedule $(0.25, 0.5, 0.75)$ gradually increases their contribution as the scoring criterion becomes stricter, yielding the best overall performance.

Table 4: Selection of the stage-wise data merge schedule $\alpha$. Our progressive schedule gradually increases the contribution of pseudo-collision data across grow steps.

| $\alpha^{(1)}, \alpha^{(2)}, \alpha^{(3)}$ | **PreADE** ↓ | **PreFDE** ↓ | **PostADE** ↓ | **PostFDE** ↓ | **rel_dist_err** ↓ |
|---|---|---|---|---|---|
| $1.00, 1.00, 1.00$ (Const.) | 1.795 | 1.665 | 1.554 | 2.232 | 2.062 |
| $\mathbf{0.25}, \mathbf{0.50}, \mathbf{0.75}$ (Ours) | **1.667** | **1.498** | **1.337** | **1.903** | **1.803** |
| $0.75, 0.50, 0.25$ (Aggr.) | 2.223 | 2.126 | 1.795 | 2.575 | 2.302 |

### A.3 Physical Strictness Schedule

As explained in Sec. 2.3 and Eq. (4), lower $\gamma$ leads to higher $\phi_\gamma(r)$ for fixed residual $r$. This leads to relatively higher energy $E_\gamma$ for the same residuals. According to Eq. (2), the model evolves towards sampling trajectories of lower energy, which leads to sampling trajectories with lower residuals, or, more strictly physically plausible trajectories. Thus, we will schedule $\gamma$ to decrease over the self-training stages, so that the model progressively imposes higher physical strictness on sampled trajectories. In addition, we will use a fixed temperature parameter $\mathcal{T}$ in Eq. (2) (set to $\mathcal{T} = 0.1$, no annealing), so that the physical strictness of sampling depends only on $\gamma$.

The stage-wise physical strictness $\gamma^{(k)}$ used at stage $k$ of self-training is scheduled as

$$\gamma^{(k)} = \begin{cases} 1.0, & k = 1, \\ 0.9, & k = 2, \\ 0.8, & k = 3. \end{cases} \tag{14}$$

As discussed in Sec. 2.3, $\gamma\mu$ can be viewed as an effective threshold for a residual to achieve a zero penalty. Since $\gamma^{(k)}$ decreases over the stages, the threshold becomes lower, meaning that it becomes tougher to achieve a zero penalty. Thus, later stages (larger $k$) apply progressively stricter physical criteria.

Next, we examine the selection of physical strictness schedule $\gamma$ in Table 5. The loose schedule $(1.2, 1.0, 0.8)$ uses a permissive threshold at the first stage, admitting more low-quality samples and resulting in slower convergence and worse overall performance. The stricter schedule $(1.0, 0.8, 0.6)$ slightly improves PreADE and PreFDE, but performs worse on PostADE, PostFDE, and `rel_dist_err`. As post-collision performance is more important for our objective, we choose $(1.0, 0.9, 0.8)$ as the final schedule.

Table 5: Selection of the stage-wise physical strictness schedule $\gamma$.

| $\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)}$ | **PreADE** ↓ | **PreFDE** ↓ | **PostADE** ↓ | **PostFDE** ↓ | **rel_dist_err** ↓ |
|---|---|---|---|---|---|
| $1.2,\ 1.0,\ 0.8$ (Loose) | 1.949 | 1.801 | 1.683 | 2.391 | 2.228 |
| $\mathbf{1.0},\ \mathbf{0.9},\ \mathbf{0.8}$ (Ours) | 1.667 | 1.498 | **1.337** | **1.903** | **1.803** |
| $1.0,\ 0.8,\ 0.6$ (Strict) | **1.660** | **1.479** | 1.425 | 1.953 | 1.986 |

### A.4 Candidate trajectory generation with quartic Bézier curves

Given trajectory $\boldsymbol{\tau}$, we fit it with the quartic (4-th order) Bézier curve and derive its control points $P_0, ..., P_4$ as follows. Let $u_t$ denote the normalized time steps, i.e., $u_t = t/N$ for $t = 0, ..., N$ with $N = 30$ or 3 seconds times the step length $\Delta t = 0.1$. Also, let the position of the trajectory at step $t$ by $\mathbf{x}_t$. We have fixed starting and ending control points as the terminal points of the trajectory, i.e., $P_0 = \mathbf{x}_0$ and $P_4 = \mathbf{x}_{N-1}$. We find the intermediate control points $P_1, P_2, P_3$ to $\mathbf{x}_t$ by solving the following least-squares problem:

$$\min_{P_1, P_2, P_3} \sum_{t=0}^{N} \left\| \mathbf{x}_t - \sum_{k=0}^{4} b_{4,k}(u_t) P_k \right\|_2^2 \tag{15}$$

where

$$b_{n,k}(u) = \binom{n}{k} u^k (1-u)^{n-k}$$

is the Bernstein polynomial of degree $n$. This fitting step recovers control points that approximate the shape of the input trajectory $\boldsymbol{\tau}$.

Next, we generate random perturbation $\Delta$ and compute the displacement of the endpoint:

$$P_4^* = P_4 + \Delta. \tag{16}$$

Instead of perturbing all control points, we preserve $P_1$ to maintain the initial direction of the trajectory and shift only $P_2$ and $P_3$ toward the sampled endpoint. Using cubic Hermite interpolation Ebert et al. (1998)

$$g(u) = 3u^2 - 2u^3, \tag{17}$$

we define the shifted control points as

$$P_2^* = P_2 + \underbrace{g\left(\frac{1}{2}\right)}_{:=g_2}\Delta = P_2 + 0.5\Delta, \tag{18}$$

$$P_3^* = P_3 + \underbrace{g\left(\frac{3}{4}\right)}_{:=g_3}\Delta \approx P_3 + 0.84\Delta. \tag{19}$$

The final sampled trajectory is then obtained from the quartic Bézier curve with control points $P_0, P_1, P_2^*, P_3^*, P_4^*$:

$$b_{4,0}(u_t)P_0 + b_{4,1}(u_t)P_1 + b_{4,2}(u_t)P_2^* + b_{4,3}(u_t)P_3^* + b_{4,4}(u_t)P_4^*, \qquad u \in [0,1]. \tag{20}$$

In summary, our construction preserves the local shape of the original trajectory by fixing $P_1$, while smoothly adapting the tail of the curve to the sampled terminal endpoint $P_4$ through the shifted control points $P_2^*$ and $P_3^*$.

**Approximation error of Bézier fitting of trajectories.** We evaluate the approximation error introduced by the quartic Bézier fitting. Specifically, we compare the simulated trajectories with their corresponding Bézier-fitted trajectories using PostADE and relative-distance error (`rel_dist_err`). The results are reported in Table 6.

Table 6: Approximation error between simulated trajectories and their Bézier-fitted trajectories.

| **Metric** | **PostADE** $\downarrow$ | **rel_dist_err** $\downarrow$ |
|---|---|---|
| Error (m) | 0.067 | 0.102 |

The approximation errors are small, with a PostADE of $0.067$ m and a relative-distance error of $0.102$ m. Thus, the discrepancy between the original and Bézier-fitted trajectories remains at approximately the centimeter scale, even for post-collision motion. This confirms that the quartic Bézier representation is sufficiently accurate for trajectories of three-second duration in our work. Moreover, the parametric representation enables smooth and controllable trajectory perturbation through the control points, avoiding the irregularity that can arise from directly perturbing individual waypoints.

### A.5 Pre-Collision residuals

**Time to Collision.** Since the output of Pre-collision Phase is used as the input history for Post-collision Phase, the generated pre-collision trajectory must contain a sufficiently long pre-impact segment. In our setting, Pre-collision Phase generates a 30-step (or $T = 3$ seconds) future trajectory, while Post-collision Phase takes a 20-step history as input. Therefore, a desirable Pre-collision Phase sample should place the collision close to the end of the 30-step horizon, so that the last 20 pre-impact steps can be used to construct the chained history for Post-collision Phase.

To enforce this property, we define the residual time-to-collision $r_{\text{ttc}}(\boldsymbol{\tau})$ as follows. Let $k^*$ denote the index of the time step in given trajectory $\boldsymbol{\tau}$ where two vehicles are the closest, we have that

$$r_{\text{ttc}}(\boldsymbol{\tau}) = \frac{T}{\Delta t} - k^* \tag{21}$$

where $T = 3$ seconds with time step size $\Delta t = 0.1$ s.

**Motion irregularity.** Let $\mathbf{pos}_i^{(t)} = (x_i^{(t)}, y_i^{(t)})$ denote the position of vehicle $i$ for $i = 1, 2$ at time step $t$. Let $\mathbf{v}_i^{(t)} = (v_{x,i}^{(t)}, v_{y,i}^{(t)})$, $\mathbf{a}_i^{(t)} = (a_{x,i}^{(t)}, a_{y,i}^{(t)})$, and $\mathbf{j}_i^{(t)} = (j_{x,i}^{(t)}, j_{y,i}^{(t)})$ denote its velocity, acceleration, and jerk, respectively, where $i \in \{1, 2\}$. The velocity of vehicle $i$ is computed from consecutive positions as

$$\mathbf{v}_i^{(t)} = \frac{\mathbf{pos}_i^{(t+1)} - \mathbf{pos}_i^{(t)}}{\Delta t} = \left( \frac{x_i^{(t+1)} - x_i^{(t)}}{\Delta t}, \frac{y_i^{(t+1)} - y_i^{(t)}}{\Delta t} \right). \tag{22}$$

The acceleration magnitude is then defined as

$$a_i^{(t)} = \left\| \frac{\mathbf{v}_i^{(t+1)} - \mathbf{v}_i^{(t)}}{\Delta t} \right\|_2 = \left\| \left( \frac{v_{x,i}^{(t+1)} - v_{x,i}^{(t)}}{\Delta t}, \frac{v_{y,i}^{(t+1)} - v_{y,i}^{(t)}}{\Delta t} \right) \right\|_2. \tag{23}$$

Similarly, the jerk magnitude is defined as

$$j_i^{(t)} = \left\| \frac{\mathbf{a}_i^{(t+1)} - \mathbf{a}_i^{(t)}}{\Delta t} \right\|_2 = \left\| \left( \frac{a_{x,i}^{(t+1)} - a_{x,i}^{(t)}}{\Delta t}, \frac{a_{y,i}^{(t+1)} - a_{y,i}^{(t)}}{\Delta t} \right) \right\|_2. \tag{24}$$

To compute the yaw rate, we first estimate the heading angle from the displacement direction:

$$\psi_i^{(t)} = \operatorname{atan2}\left( y_i^{(t+1)} - y_i^{(t)}, x_i^{(t+1)} - x_i^{(t)} \right). \tag{25}$$

where $\operatorname{atan2}$ denotes the four-quadrant inverse tangent Press & Teukolsky (2007). Because heading angles are periodic, we apply an angle-wrapping function to avoid artificial discontinuities at the $\pi$ and $-\pi$ boundary:

$$\operatorname{wrap}(\theta) = \operatorname{atan2}(\sin\theta, \cos\theta). \tag{26}$$

The yaw rate is then computed as

$$\omega_i^{(t)} = \frac{\left| \operatorname{wrap}\left( \psi_i^{(t+1)} - \psi_i^{(t)} \right) \right|}{\Delta t}. \tag{27}$$

We define the acceleration, jerk, and yaw-rate residuals as the maximum values over the duration of Pre-collision Phase:

$$r_{\text{acc}}(\boldsymbol{\tau}) = \max_t \left[ \max\left( a_1^{(t)}, a_2^{(t)} \right) \right], \tag{28}$$

$$r_{\text{jerk}}(\boldsymbol{\tau}) = \max_t \left[ \max\left( j_1^{(t)}, j_2^{(t)} \right) \right], \tag{29}$$

$$r_{\text{yaw}}(\boldsymbol{\tau}) = \max_t \left[ \max\left( \omega_1^{(t)}, \omega_2^{(t)} \right) \right]. \tag{30}$$

### A.6 Normal and Tangential Post-Collision Residuals

We provide detailed definitions of the normal and tangential post-collision residuals used in Section 2.3. For each collision event, we compute the pre-impact velocities $\mathbf{v}_1^-, \mathbf{v}_2^-$, the predicted post-impact velocities $\mathbf{v}_{1,\text{pred}}^+, \mathbf{v}_{2,\text{pred}}^+$, and the physically expected post-impact velocities under a planar rigid-body collision approximation.

Let the relative pre-impact velocity be

$$\mathbf{v}_{\text{rel}} = \mathbf{v}_1^- - \mathbf{v}_2^-. \tag{31}$$

We decompose this relative velocity into normal and tangential components with respect to the collision frame:

$$v_{\text{rel},n} = \mathbf{v}_{\text{rel}} \cdot \hat{\mathbf{n}}, \qquad v_{\text{rel},t} = \mathbf{v}_{\text{rel}} \cdot \hat{\mathbf{t}}, \tag{32}$$

where $\hat{\mathbf{n}}$ and $\hat{\mathbf{t}}$ denote the normal and tangent directions of the collision.

The normal impulse can be computed as standard formula of 2D collision Meriam et al. (2020):

$$J_n = -\frac{(1+e)v_{\mathrm{rel},n}}{\frac{1}{m_1}+\frac{1}{m_2}}, \tag{33}$$

where $e$ is the coefficient of restitution. In practice, $e$ may be selected from a small grid by choosing the value that minimizes the resulting normal and tangential velocity residuals. We use a representative value such as $e = 0.15$ during training.

The tangential impulse models collision-surface friction. There are two cases following the Coulomb's law of friction: sticking and sliding Wang & Mason (1992); Featherstone (2008). It depends on the impulse at the collision: in the sticking case, two objects stick together and move along after collision. In the sliding case, two objects slide together after collision.

We first compute the sticking impulse. This has a similar form as Eq. (33) except that the restitution term is set to 0 due to sticking:

$$J_t^{\mathrm{stk}} = -\frac{v_{\mathrm{rel},t}}{\frac{1}{m_1}+\frac{1}{m_2}}. \tag{34}$$

Next, the tangential impulse is then given by the Coulomb friction rule Featherstone (2008):

$$J_t = \begin{cases} J_t^{\mathrm{stk}}, & \left|J_t^{\mathrm{stk}}\right| \le \mu_{\mathrm{imp}}|J_n|, \\ \mu_{\mathrm{imp}}J_n, & J_t^{\mathrm{stk}} > \mu_{\mathrm{imp}}J_n, \\ -\mu_{\mathrm{imp}}J_n, & J_t^{\mathrm{stk}} < -\mu_{\mathrm{imp}}J_n \end{cases} \tag{35}$$

where $\mu_{\mathrm{imp}}$ is the impact-surface friction coefficient. We use $\mu_{\mathrm{imp}} = 0.4$ in training.

The total collision impulse is

$$\mathbf{J} = J_n\hat{\mathbf{n}} + J_t\hat{\mathbf{t}}. \tag{36}$$

Using this impulse, the physically expected post-impact velocities are

$$\mathbf{v}_{1,\mathrm{exp}}^{+} = \mathbf{v}_1^{-} + \frac{\mathbf{J}}{m_1}, \qquad \mathbf{v}_{2,\mathrm{exp}}^{+} = \mathbf{v}_2^{-} - \frac{\mathbf{J}}{m_2}. \tag{37}$$

The normal and tangential residuals are then evaluated by projecting the difference between predicted and physically expected post-impact velocities onto the collision frame. In the windowed implementation, these residuals are evaluated over the active post-impact contact window, and the maximum value in the window is used. The mean of normal and tangential residuals over the vehicles are given by

$$r_{\mathrm{norm}}(\boldsymbol{\tau}) = \frac{1}{2}\left[\left|\left(\mathbf{v}_{1,\mathrm{pred}}^{+} - \mathbf{v}_{1,\mathrm{exp}}^{+}\right)\cdot\hat{\mathbf{n}}\right| + \left|\left(\mathbf{v}_{2,\mathrm{pred}}^{+} - \mathbf{v}_{2,\mathrm{exp}}^{+}\right)\cdot\hat{\mathbf{n}}\right|\right], \tag{38}$$

$$r_{\mathrm{tan}}(\boldsymbol{\tau}) = \frac{1}{2}\left[\left|\left(\mathbf{v}_{1,\mathrm{pred}}^{+} - \mathbf{v}_{1,\mathrm{exp}}^{+}\right)\cdot\hat{\mathbf{t}}\right| + \left|\left(\mathbf{v}_{2,\mathrm{pred}}^{+} - \mathbf{v}_{2,\mathrm{exp}}^{+}\right)\cdot\hat{\mathbf{t}}\right|\right]. \tag{39}$$

### A.7 Baseline Implementation Details

This appendix provides additional implementation details for the baselines used in the Experiments section. We divide the baselines into three groups: text-to-trajectory generation, trajectory-to-trajectory generation, and trajectory prediction.

For the text-to-trajectory baselines InteractTraj Xia et al. (2024) and LCTGen Tan et al. (2023), we inject the same ground-truth trajectory segment from $-3$ to $-1$ seconds relative to collision while following each model's native input format. Specifically, InteractTraj receives the segment through its map, interaction, and vehicle code inputs, whereas LCTGen receives the same information through its actor and map vector representation. Because these text-to-trajectory baselines do not explicitly enforce alignment to the ground-truth frame, we apply Procrustes alignment Schönemann (1966) during evaluation.

TrafficGen Feng et al. (2022) is a trajectory-to-trajectory generation model, so we provide the same $-3$ to $-1$ second segment directly as input and compare its predicted 30-step output against the ground-truth segment from $-1$ to $+2$ seconds. Since it directly conditions on the input trajectory segment, we do not apply the additional alignment step used for the text-to-trajectory baselines.

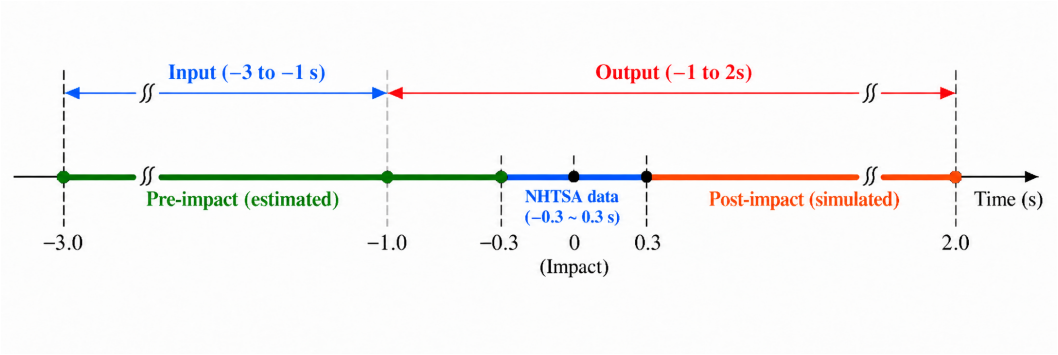



Fig 6: Overview of the GT data generation procedure from NHTSA data.

For the trajectory prediction baselines LaneGCN Liang et al. (2020), HiVT Zhou et al. (2022), and HPNet Tang et al. (2024), we use the same input-output protocol as for TrafficGen: each model receives the ground-truth trajectory segment from $-3$ to $-1$ seconds as input and predicts the 30-step segment from $-1$ to $+2$ seconds. Because these models also directly condition on the observed trajectory history, we evaluate them without the additional Procrustes alignment step.

For KING Hanselmann et al. (2022), we keep V1 fixed to its ground-truth trajectory and optimize the steering and acceleration sequences of V2. Following the original KING formulation, The trajectory of V2 is parameterized by unrolling a differentiable kinematic bicycle model, and its control sequence is optimized using gradients through the kinematics model. Except for fixing V1 rather than rolling out a reactive ego policy, we retain the original KING optimization procedure as closely as possible during inference.

### A.8 Construction and Validation of Evaluation Data

#### A.8.1 Ground-Truth Data Construction

We construct the ground-truth (GT) data for evaluation based on the NHTSA data. Our goal is to combine real-world crash records with physics-based simulation to reconstruct vehicle trajectories throughout the collision event. The overall process consists of four steps: (i) reconstructing the collision event from NHTSA data, (ii) generating the prolonged pre-impact trajectory via backward integration, (iii) simulating the subsequent vehicle motion using CARLA and Chrono, and (iv) validating the reconstructed results to select samples for the final GT data.

The data construction procedure is summarized below. Throughout the procedure, $t = 0$ denotes the instant of impact. The timeline of our process is illustrated in Figure 6.

1. **Collision-event reconstruction ($-0.3$ to $0.3$ s).** We first extract the NHTSA pre- and post-impact data within the interval from $-0.3$ to $0.3$ s around the collision instant. These observations are used to reconstruct the collision event and the vehicle states immediately before and after impact.
2. **Pre-impact trajectory generation ($-3$ to $0$ s).** Using the reconstructed pre-impact states within the interval from $-0.3$ to $0$ s, we perform backward integration to generate the pre-impact trajectory from $-3$ to $0$ s. This trajectory describes the vehicle motion leading to the collision.
3. **Physics-based forward simulation ($-1$ to $2$ s).** The reconstructed trajectory from $-3$ to $-1$ s is provided as input to the CARLA and Chrono simulators. The physics engines of the simulators generate the subsequent trajectory from $-1$ to $2$ s. The generated trajectory covers the final part of pre-impact motion, the collision event, and the post-impact response for $2$ s after impact.
4. **Validation and selection.** After simulation, we validate the simulated trajectories by comparing them against NHTSA data. Specifically, we take the post-impact segment from $0$ to $0.3$ s of a simulated trajectory. The segment is compared against the recorded NHTSA data based on the acceptance criteria described in Appendix A.8.2. Only samples that satisfy the validation requirements are retained in the final GT dataset.

Table 7: NHTSA/CISS reference signals and their corresponding reconstructed quantities used for GT validation.

| Validation target | NHTSA/CISS reference | Reconstructed quantity |
|---|---|---|
| Pre-impact speed | EDRPRECRASH PCODE=1010 | Speed magnitude from $v_x$ and $v_y$ |
| Post-impact longitudinal $\Delta V(t)$ | EDRPOSTCRASH PCODE=2010 | Longitudinal $\Delta V(t)$ |
| Post-impact lateral $\Delta V(t)$ | EDRPOSTCRASH PCODE=2020 | Lateral $\Delta V(t)$ |
| Post-impact $\Delta V(t)$ | EDRPOSTCRASH PCODE=2030/2040 | Acceleration-derived longitudinal/lateral $\Delta V(t)$ |
| Vehicle-level $\Delta V$ | GV/CDC DVLONG, DVLAT, DVTOTAL | Longitudinal, lateral, and total $\Delta V$ |
| Impact heading | CDC HEADINGANG | YAW_COMPASS_DEG at contact |

In summary, our procedure generates reconstructed GT trajectories spanning $-3$ to 2 s around impact for downstream tasks by combining NHTSA crash records, backward trajectory generation, and physics-based forward simulation.

### A.8.2 Validation of the Reconstructed Ground-Truth Data

**Overview.** We validate the reconstructed ground-truth (GT) data generated with CARLA/Chrono against the measurement data from the NHTSA Crash Investigation Sampling System (CISS). Table 7 summarizes the validation targets, the CISS reference fields used for each target, and the corresponding reconstructed quantities.

Our validation covers both pre-impact vehicle motion and post-impact collision response. Pre-impact quantities assess whether the reconstructed vehicle state prior to contact is consistent with the CISS record. Post-impact $\Delta V$ measures whether the reconstructed collision response reproduces the observed change in vehicle motion. When direct post-impact $\Delta V$ measurements are unavailable or incomplete, acceleration-derived $\Delta V$ is used as a surrogate reference as summarized in Table 7.

**Validation procedure.** Let $x$ denote the reference value obtained from the NHTSA record and $y$ the corresponding reconstructed value. For each signal, we evaluate both point-wise agreement and distribution-level consistency between the reference and reconstructed observations. The residual is defined as

$$d = y - x. \tag{40}$$

The mean absolute error (MAE) is used to quantify the absolute reconstruction error. In addition, the Bland–Altman limits of agreement (LoA) are computed to characterize systematic bias and the dispersion of the residuals:

$$\mathrm{LoA} = \mathrm{mean}(d) \pm 1.96\,\mathrm{SD}(d). \tag{41}$$

To characterize the linear relationship between the reference and reconstructed quantities, we additionally fit an ordinary least-squares (OLS) regression,

$$y = ax + b, \tag{42}$$

where $a$ denotes the regression slope and $b$ denotes the intercept. We report the slope $a$, intercept $b$, and Pearson correlation coefficient $r$ for each applicable signal.

We chose these metrics to capture complementary aspects of reconstruction fidelity, explained as follows. The MAE measures absolute reconstruction error; the Bland–Altman bias and LoA characterize systematic offset and residual dispersion; the OLS slope and intercept capture systematic scale and offset errors relative to the reference data; the Pearson correlation coefficient measures the consistency with which variation in the reference observations is reproduced by the reconstruction. We perform a validation of generated trajectories based on these metrics, and in the following, we explain the validation criteria. Only the trajectories that meet all the validation criteria are retained.

For pre-collision trajectories, validation is performed separately for vehicle position, heading, steering, speed, and yaw rate. For sequential quantities, the statistics are computed over all temporally aligned observations available for the corresponding signal.

For post-collision trajectories, we compute the statistics by pooling the longitudinal and lateral observations across vehicles within each accident scenario. In particular, post-impact $\Delta V$ is regarded as a primary validation quantity because it provides a compact representation of both the magnitude and direction of the collision-induced change in vehicle motion.

We apply acceptance thresholds to each metric considering the physical scale and units of each quantity. For pre-impact speed and post-impact $\Delta V$, the MAE, absolute Bland–Altman bias, and absolute OLS intercept must each be no greater than $0.75\ \mathrm{km/h}$. The $95\%$ LoA half-width must not exceed $1.50\ \mathrm{km/h}$. The OLS regression slope must satisfy

$$0.95 \leq a \leq 1.05, \tag{43}$$

and the Pearson correlation coefficient must satisfy $r \geq 0.99$. Pre-impact yaw rate is evaluated using the same numerical thresholds, with dimensional error quantities expressed in $\mathrm{deg/s}$. The remaining signals are evaluated using their corresponding signal-specific acceptance thresholds.

Regression and correlation criteria are applied only when the corresponding statistics are identifiable. For example, if the reference observations exhibit insufficient variation, the OLS slope or Pearson correlation coefficient may not be meaningfully defined. In such cases, acceptance is determined using the remaining applicable error and agreement criteria, including the MAE, Bland–Altman bias, and LoA.

Only the trajectories that satisfy all the acceptance criteria are retained. For each scenario, if both trajectories from CARLA and Chrono are valid, we select the trajectory that better matches CISS data using the following voting process. A trajectory receives one vote for each of the following cases: it has a lower MAE, lower absolute Bland-Altman bias, lower LoA half-width, lower $|b|$, lower $|a-1|$, or higher $r$. Finally, the trajectory with more votes is selected.

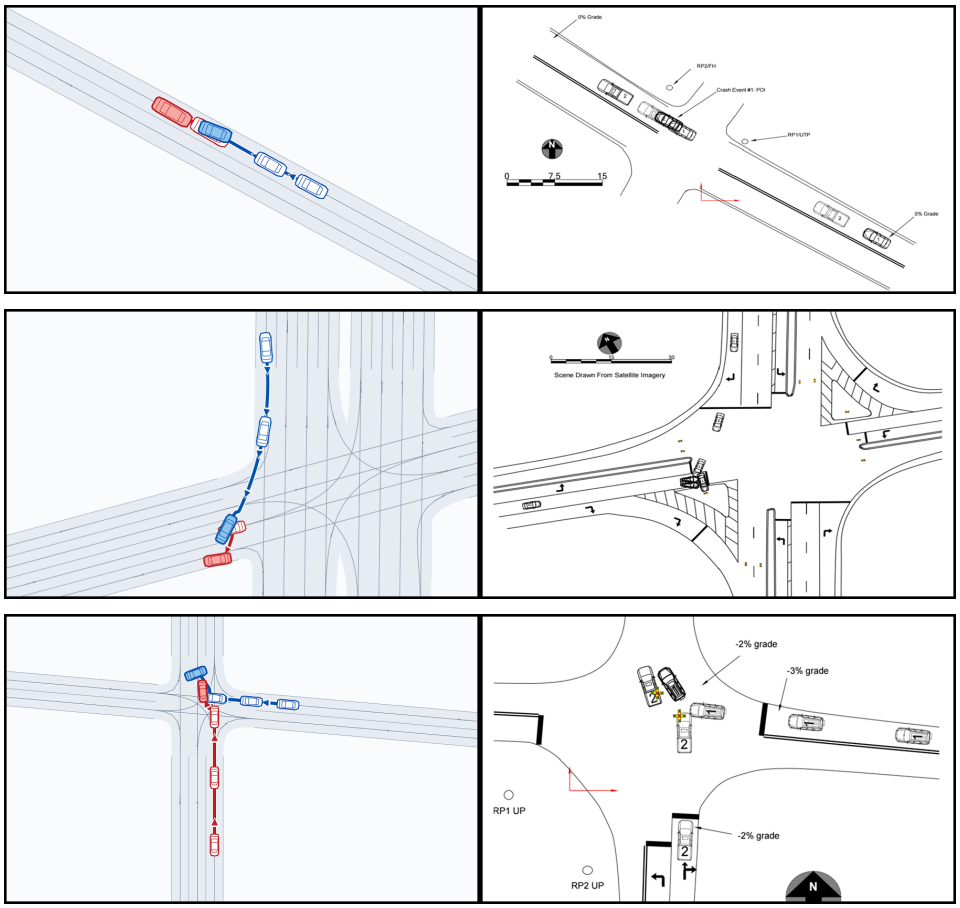



Fig 7: Comparison between the reconstructed lane-level maps and the corresponding NHTSA scene diagrams. For each example, the left image shows the reconstructed map used as input to the map-dependent trajectory models, while the right image shows the corresponding scene diagram provided by NHTSA.

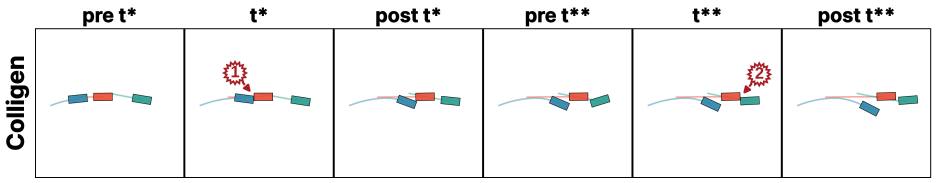
### A.9 Reconstructing Lanes and Maps

We augmented all 4,583 generated collision scenarios with reconstructed lane-level maps as explained below. The reconstructed maps and lanes concern five map-dependent models: LaneGCN, HiVT, TrafficGen, HPNet, and InteractTraj. Since these models explicitly utilize road geometry and lane connectivity as part of their input representation, the quality of the reconstructed map can directly influence the generated vehicle trajectories and, consequently, the resulting collision scenarios.

The road maps and lanes were reconstructed using static road attributes provided in the CISS records of NHTSA data together with the observed vehicle trajectories before the collision. In particular, road attributes such as the number of lanes, traffic flow direction, initial lane position, road alignment, intersection type, lane markings, and traffic-control information were used to determine the overall road structure. The vehicle trajectories observed approximately 2 s before the collision were used to estimate the road heading and to align the reconstructed lane geometry with the observed scene. Based on the estimated road direction and the initial lane information, lane centerlines were placed around the vehicle's initial position. If exact geometric information was unavailable, we assumed a lane width of 3.5 m, a median width of 2 m where applicable, and a curvature radius of 250 m for curved road segments. Intersections and lane-to-lane connectivity were then generated according to the corresponding CISS road and intersection attributes. Finally, the reconstructed road geometry was converted into lane graphs containing lane centerlines, boundaries, predecessor–successor relationships, and left–right adjacency information required by the target models.

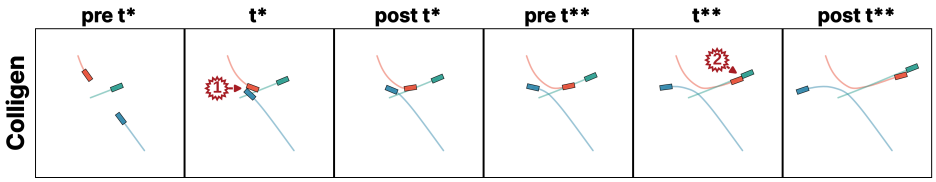
Fig. 7 compares an example of the reconstructed map with the original scene information provided by NHTSA. The left side shows the lane-level map reconstructed for our generated collision data, whereas the right side shows the corresponding scene diagram available on the NHTSA website. The reconstructed maps are synthesized from structured CISS attributes and pre-collision trajectories instead of the original accident diagrams. Thus, they are not intended to reproduce the exact road geometry of each crash scene. Instead, our reconstructed maps provide a standardized lane-level representation that preserves the major road topology, lane configuration, and driving direction required by map-dependent trajectory models.

**Three Vehicles Case 1**



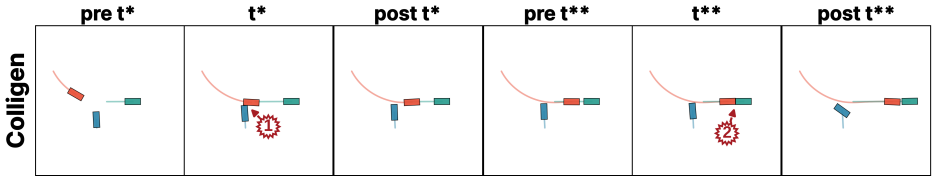

Blue, Red, and Green vehicles were traveling sequentially in the same direction. Blue vehicle first rear-ended Red vehicle, and the resulting impact caused Red vehicle to subsequently collide with the rear of Green vehicle.

**Three Vehicles Case 2**




At a T-shaped intersection, Red and Blue vehicles were approaching each other from opposite directions and collided. After the initial collision, Red vehicle was pushed continuously toward the left and subsequently collided with the rear of another Green vehicle positioned ahead.

**Three Vehicles Case 3**




At a Y-shaped intersection, the Blue vehicle lightly struck the rear of Red vehicle. Red vehicle continued moving forward after the initial impact and subsequently collided with the rear of Green vehicle.

Fig 8: Representative examples of sequential collision generation involving three vehicles. The model first generates a collision between two vehicles, and the inferred result is then used as input to generate a subsequent collision involving the third vehicle.

### A.10 Extension to Three-Vehicle Collision Scenarios

We extended our model to generate collision scenarios involving three vehicles. Although our current framework does not explicitly generate a scene in which all three vehicles collide simultaneously, it can generate trajectories for scenarios in which two collisions occur consecutively.

Suppose there are three vehicles: V1, V2, and V3. We first generated a collision scene between V1 and V2. The generation of the first collision was then fed back into the model as a new input, allowing the model to generate a second collision between V2 and V3 that extends the first collision event. In this way, the overall three-vehicle scenario can be constructed as a chained collision process, where the second collision emerges as a continuation of the first one.

Figure 8 presents three examples of our three-vehicle collision generation. In each example, the model first infers the initial collision between two vehicles and then generates a subsequent interaction involving the third vehicle. These results suggest that our framework can be extended beyond simple two-vehicle collision generation and has the potential to model more complex multi-stage accident scenarios.

### A.11 Additional Qualitative Results

In the following, we provide additional qualitative examples of Colligen. The experimental setup is consistent with that described in Section 3. For the ground-truth (GT) cases, we visualize the corresponding map to provide the original road context. For the remaining generated cases, the map is omitted only from the visualization so that the generated collision trajectories and vehicle interactions can be observed more clearly. The map information is fully preserved in both the model input and the original model output, and is omitted only from the final visualization for clarity.

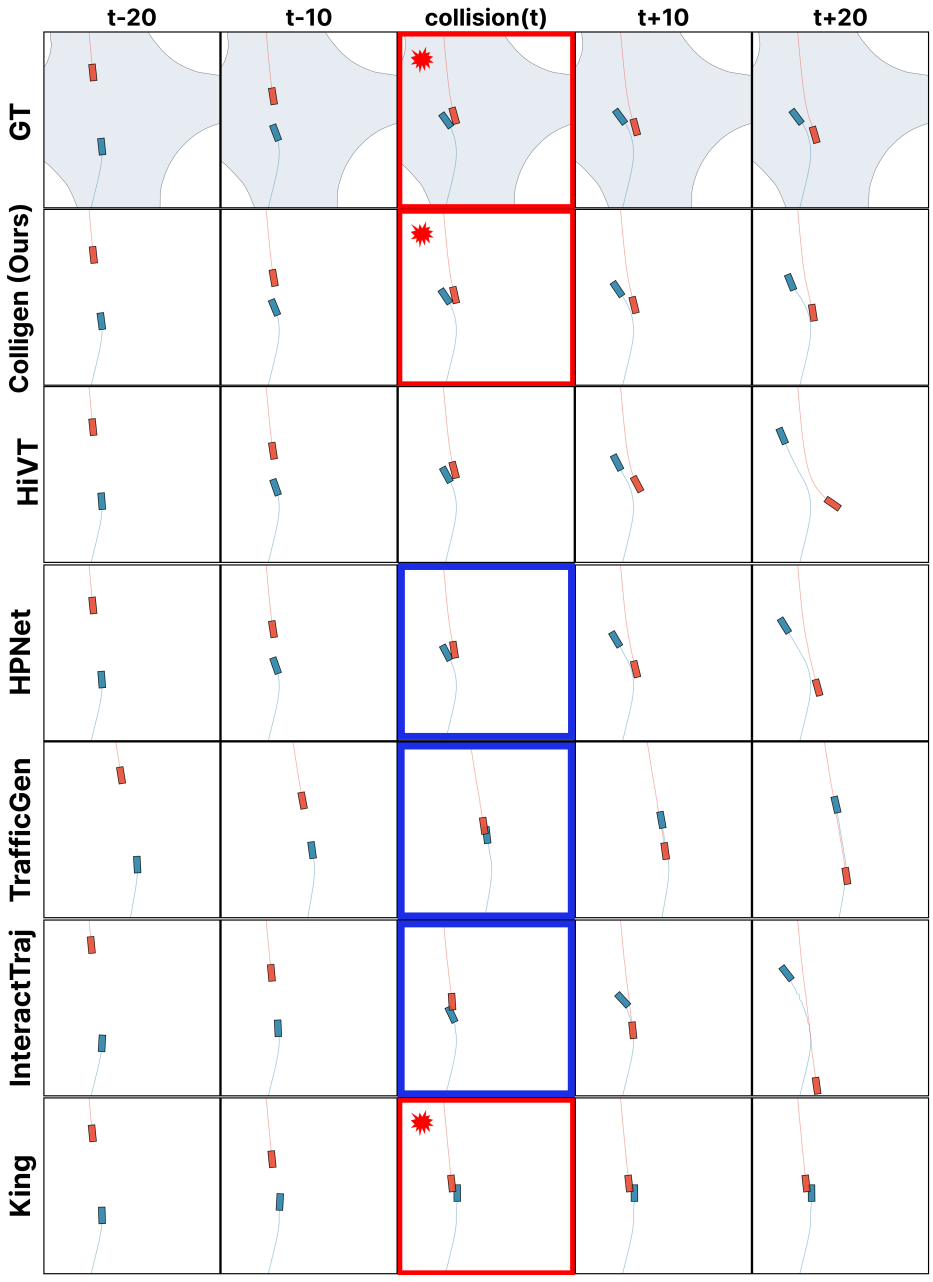



Fig 9: Blue vehicle was heading northbound on a four lane two way roadway attempting to turn left to continue west. Red vehicle was heading southbound on a four lane highway. The front of Red vehicle contacted the right side of Blue vehicle in the intersection.

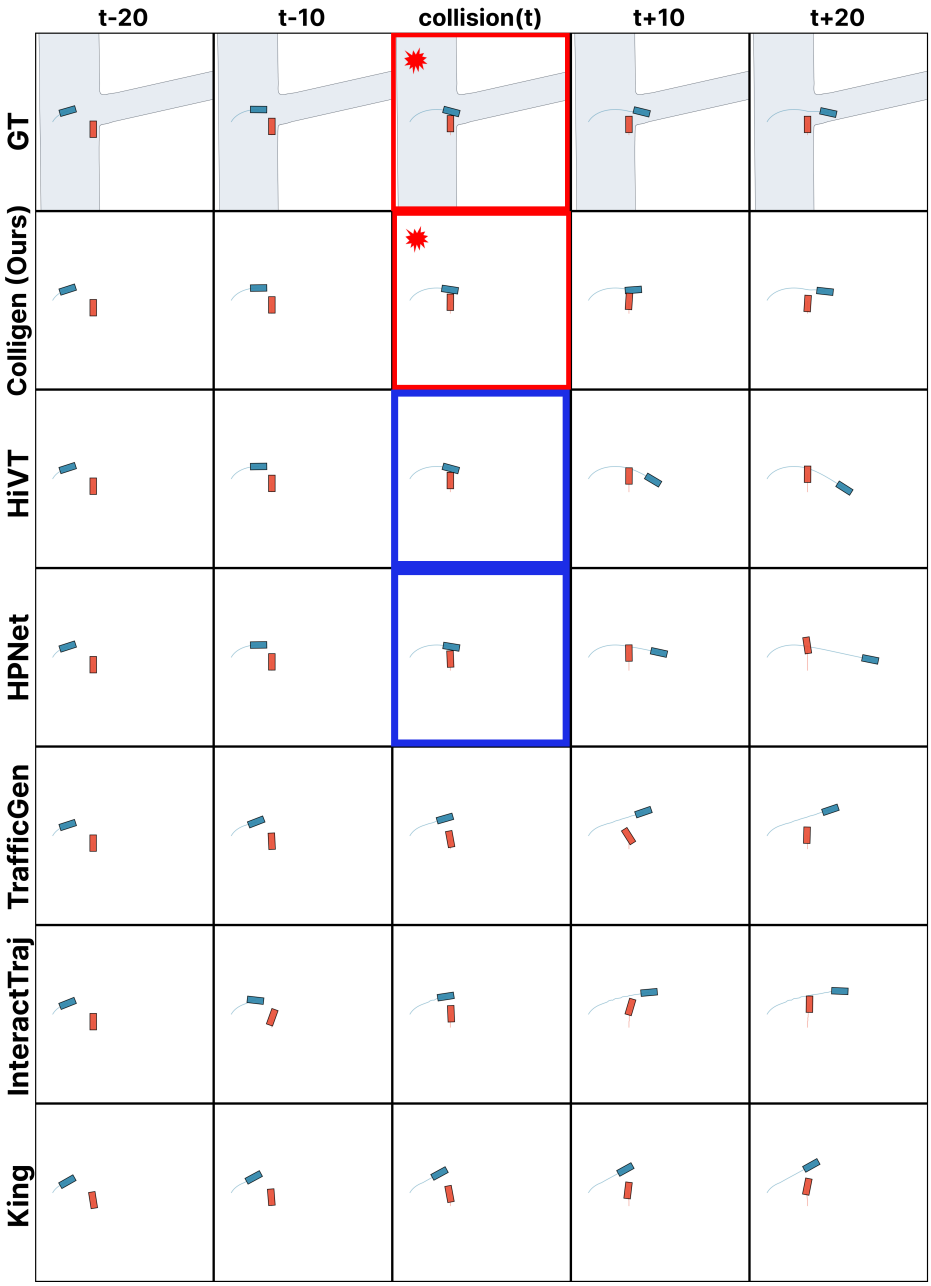



Fig 10: While Blue vehicle was making a U-turn, it collided with Red vehicle, which was traveling slowly in the right lane in the same direction.

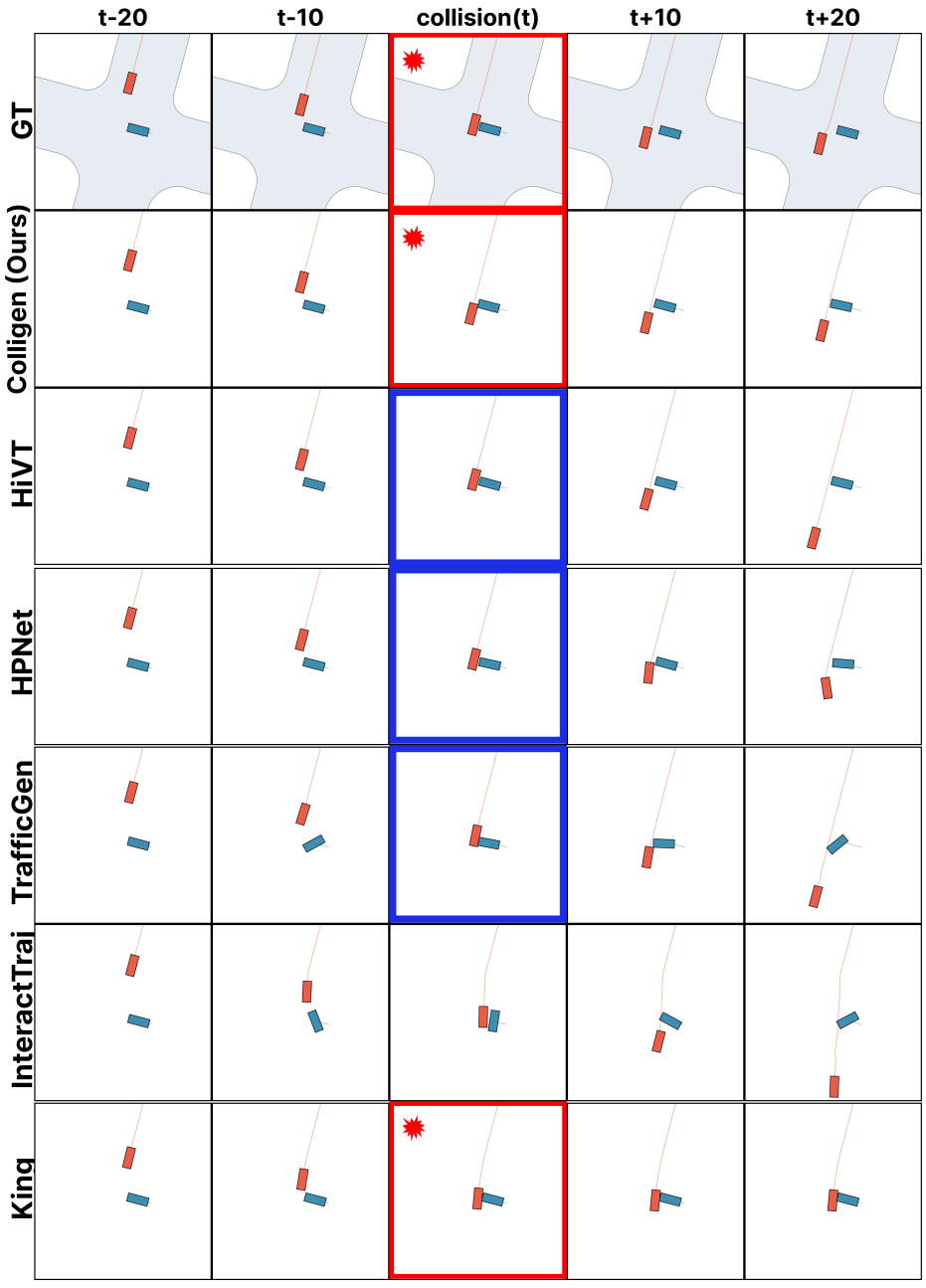



Fig 11: Blue vehicle was traveling westbound and Red vehicle was traveling southbound; both approaching the same intersection. In the intersection, the front end of Blue vehicle came into contact with the left side of Red vehicle.

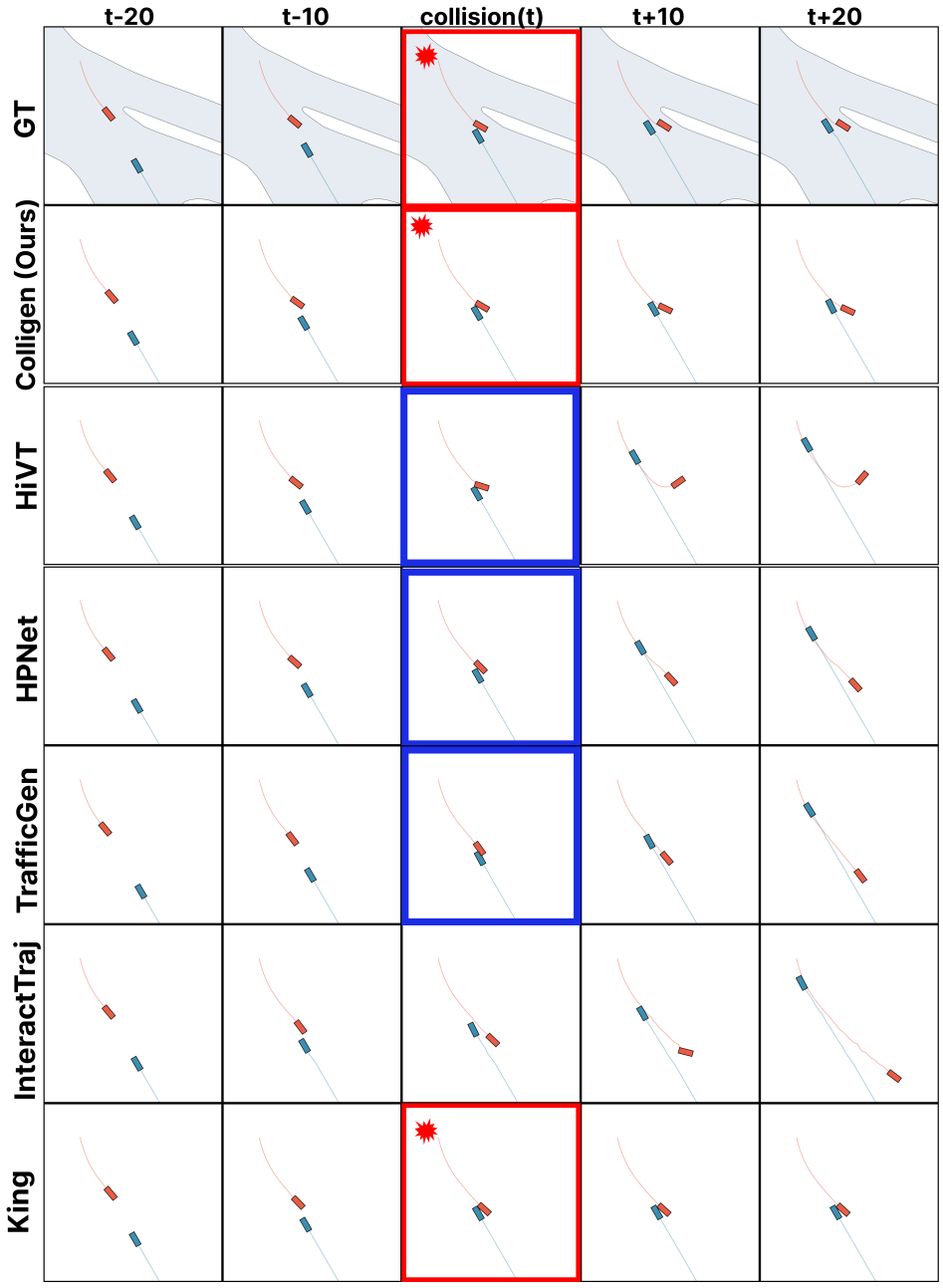



Fig 12: Blue vehicle was traveling northbound in lane one on a six-lane, two-way roadway approaching an intersection controlled by a traffic light. Red vehicle was traveling southbound in lane three on a five-lane two-way roadway intending to turn left and head eastbound at the mentioned intersection. In the middle of the intersection the front plane of Blue vehicle contacted the right side of Red vehicle. Blue vehicle came to final rest in the intersection facing north, Red vehicle came to final rest facing south.

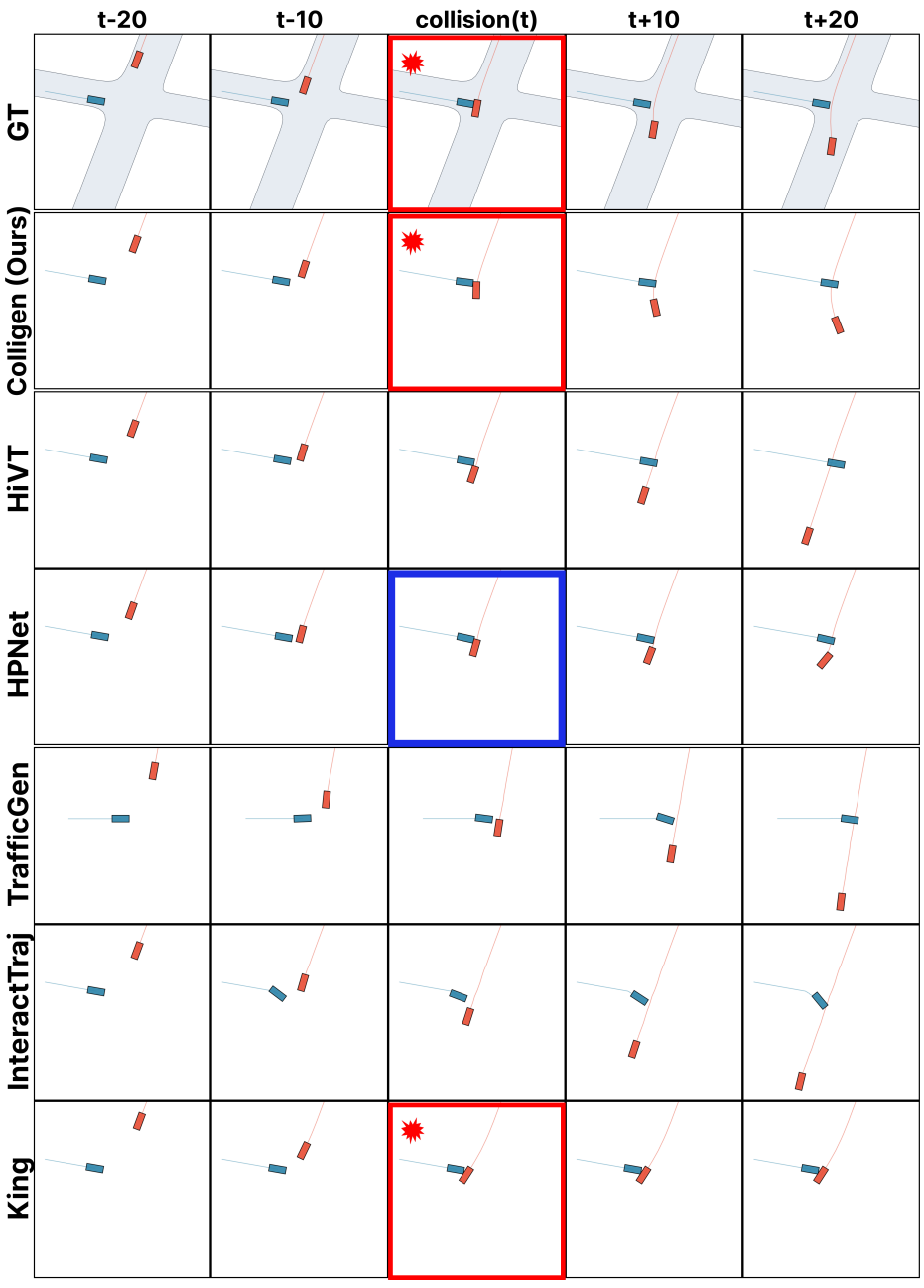



Fig 13: Red vehicle was traveling from north to south when it collided with Blue vehicle, which was entering the intersection from the right. The right side of Red vehicle was struck in the collision.

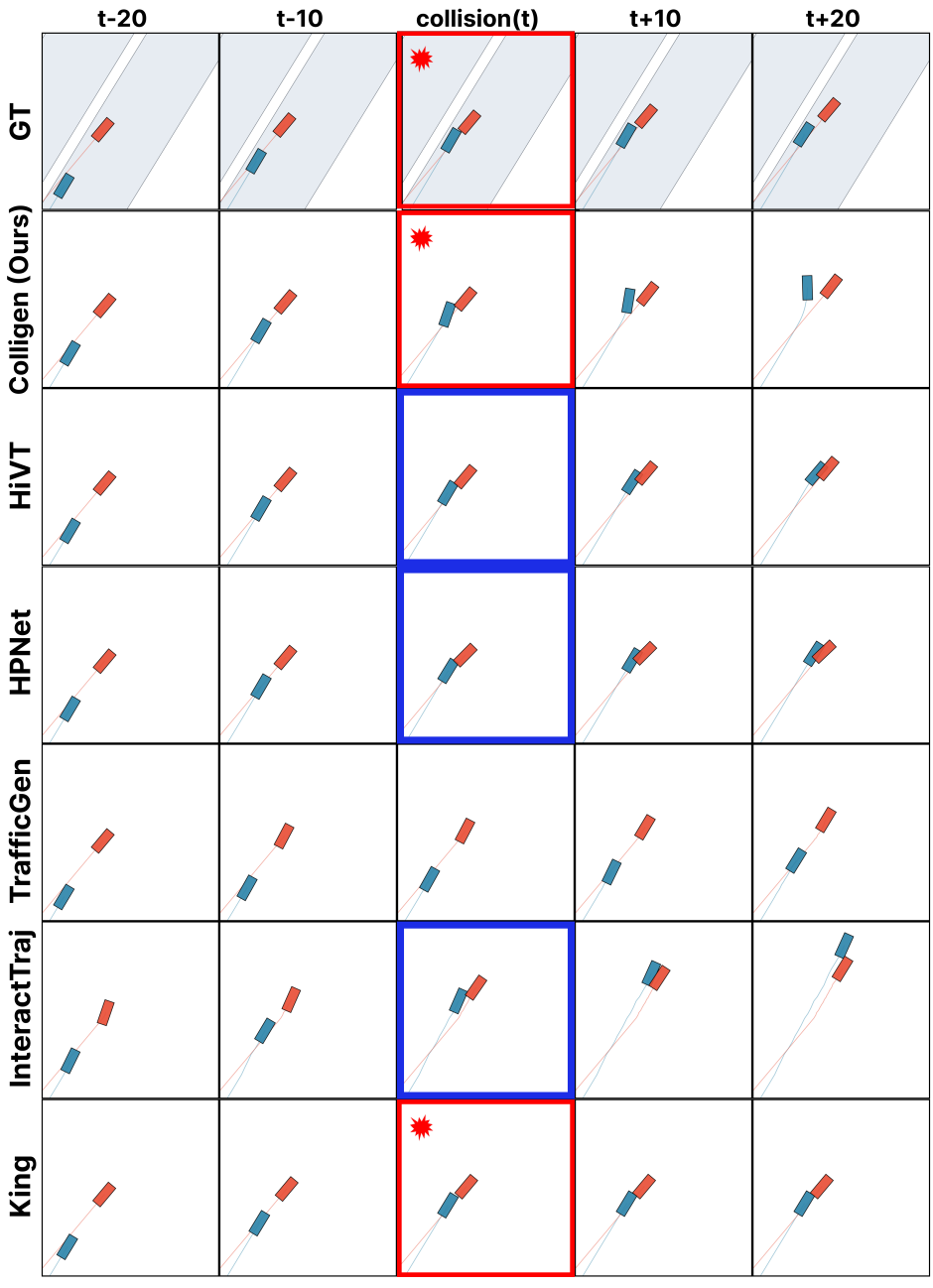



Fig 14: Blue vehicle rear-ended Red vehicle, which was traveling ahead in the same direction in an adjacent lane.

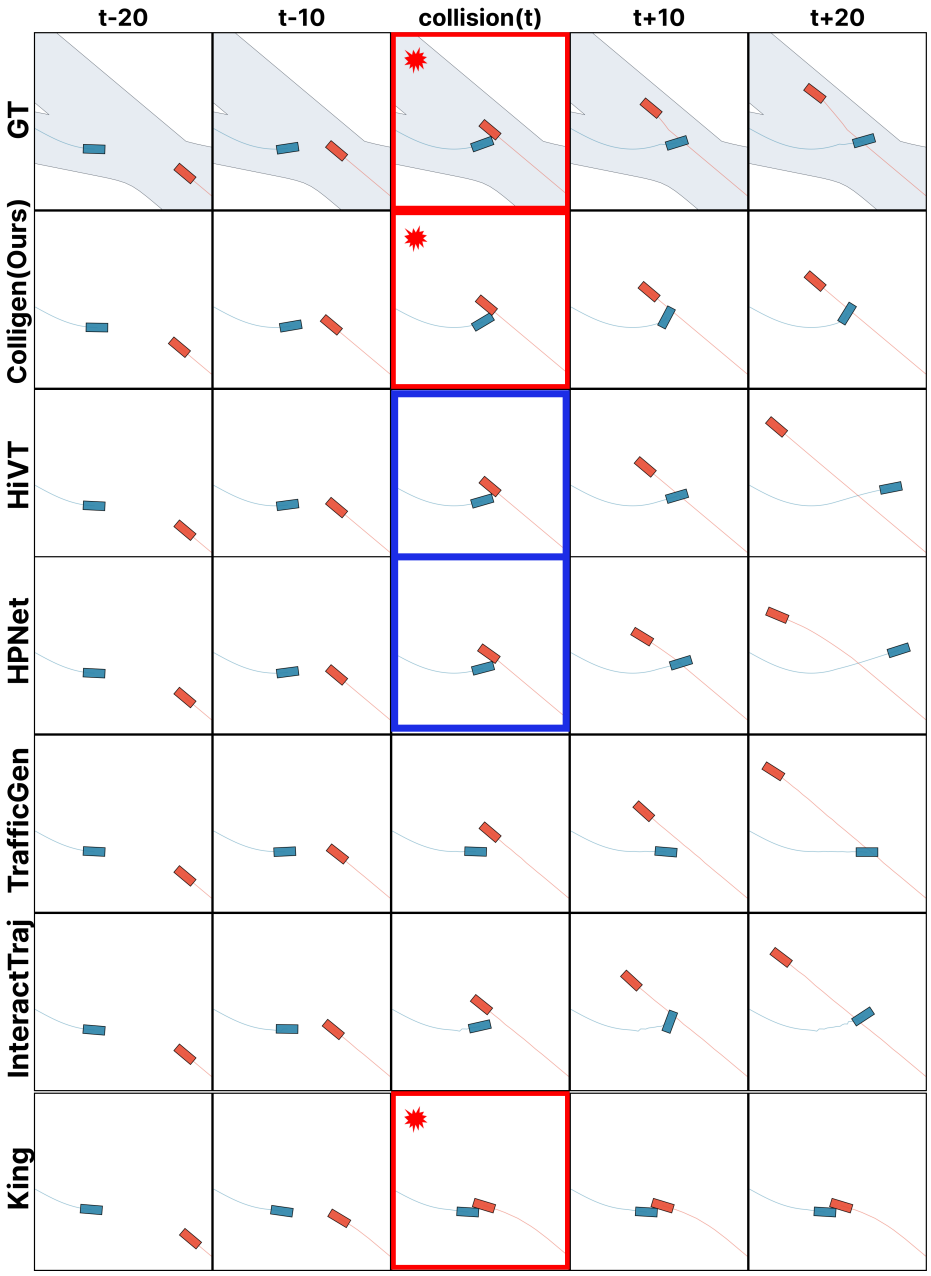



Fig 15: Blue vehicle entered from the access road on the right and struck the left side of Red vehicle, which was approaching from the opposite direction.

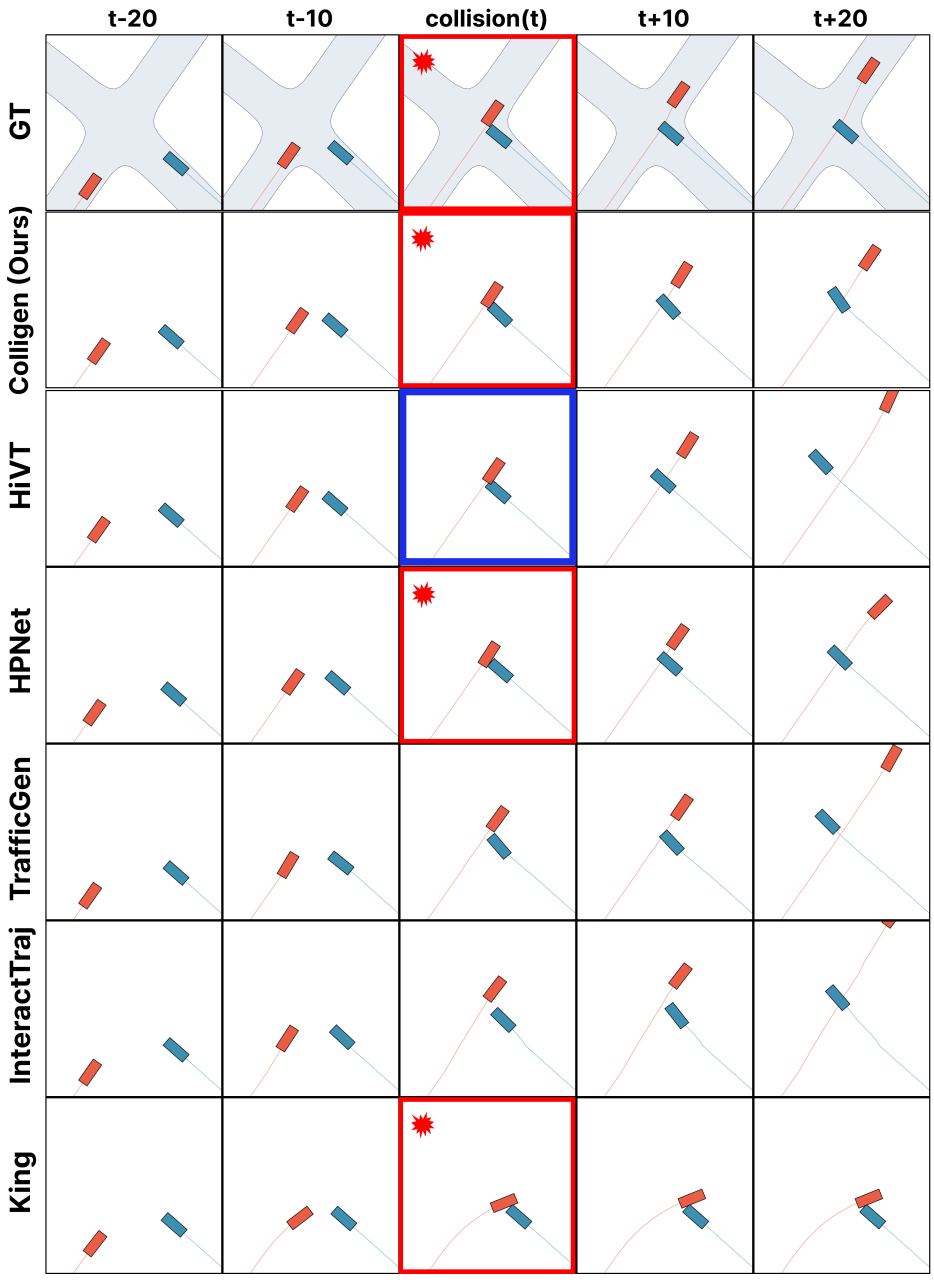



Fig 16: Blue vehicle was traveling from right to left through the intersection when it struck the right side of Red vehicle, which was also traveling through the intersection.

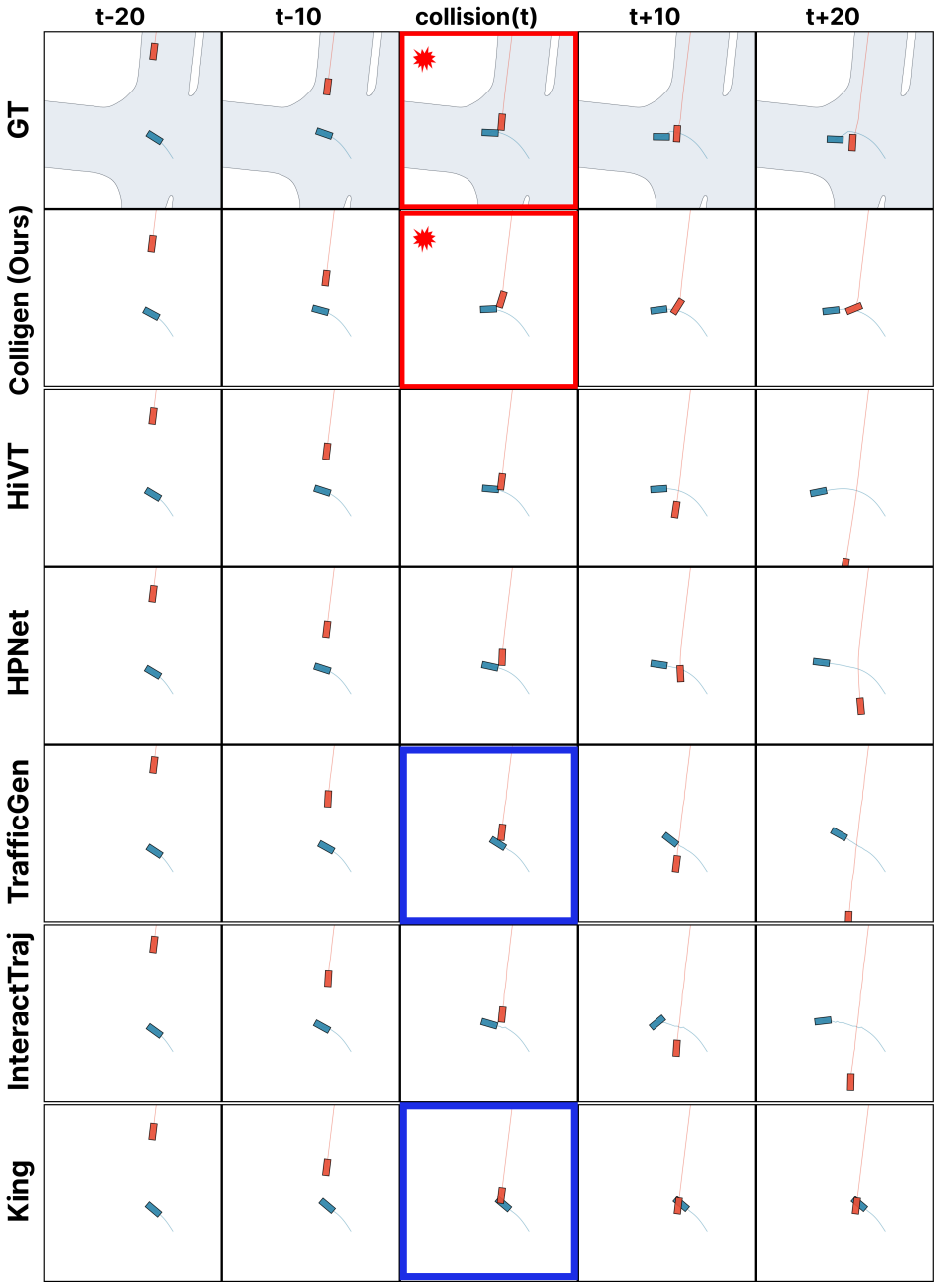



Fig 17: Blue vehicle was making a left turn at the intersection when it collided with Red vehicle, which was traveling straight from the opposite direction. The collision impacted the right side of Blue vehicle.

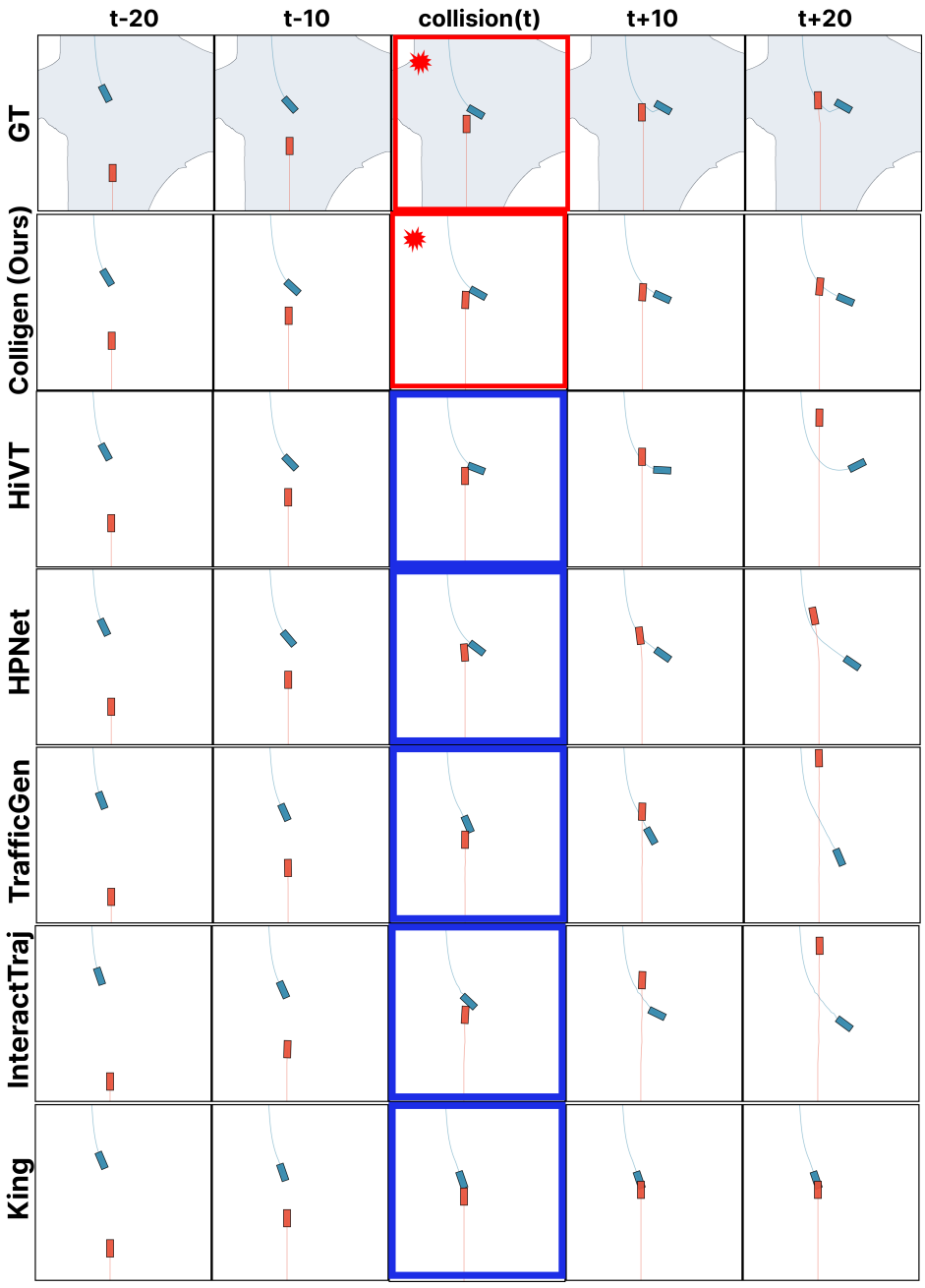



Fig 18: Blue and Red vehicles were approaching each other from opposite directions. At the intersection, Blue vehicle made a left turn, and Red vehicle struck the right side of the blue vehicle.

### A.12 FAILURE CASES

We present some failure cases of trajectory generation.

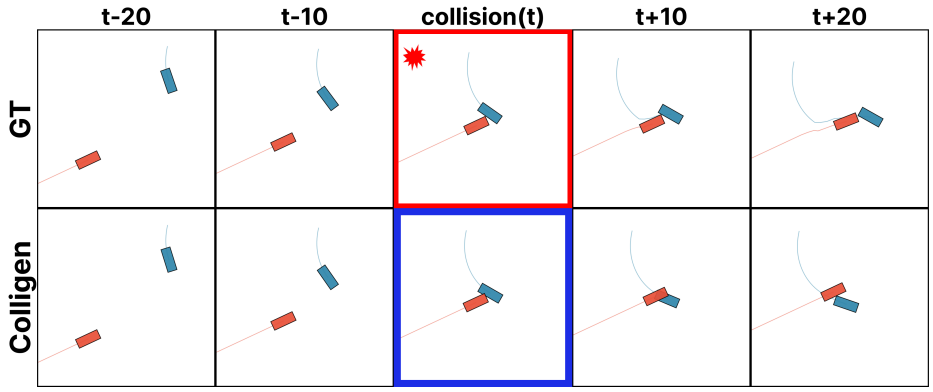



Fig 19: The GT shows a post-collision trajectory such that the vehicles come to a stop. However, our method generates a trajectory that Red vehicle grazes the rear part of Blue vehicle but continues to move on.

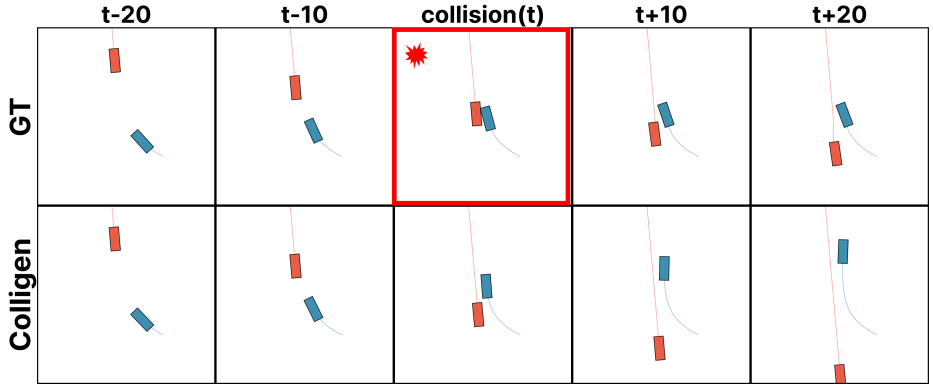



Fig 20: The GT shows a side collision between two oncoming vehicles. However, our model generates a trajectory that the two vehicles narrowly miss each other and, consequently, no collision is observed.