A Implementation Details

A.1 Model-based Generator

State Estimation. Argoverse lacks the full state description of the vehicles while only provides their history track $\mathbf{s}_i$ by a sequence of discrete centroid positions. Therefore, our framework starts with estimating the target vehicle's current state $\mathbf{s}^0_{tar}$ to initialize the model-based trajectory generator. Since the bounding box information is not given and there exists a certain degree of data noise, which makes the state estimation even harder, we thus, in the model-based part, process the track data by Kalman Filter. Afterward, the target vehicle's current velocity and heading are estimated from the processed data, while its current high-order state variables, including acceleration and turning rate, are set to zero.

Path Search. We use the Depth-First-Search algorithm to search potential paths $\mathcal{P}^+$ that the prediction target $\mathbf{s}_{tar}$ may reach on the HD Map $\mathcal{M}$. The path search algorithm $G_{path} : (\mathcal{M}, \mathbf{s}^0_{tar}) \mapsto \mathcal{P}^+$ is partially built upon the baseline implementation in [10]. Firstly we localize a_{tar} on the map and query its surrounding lane segments as the root segments. With the lane connectivity information provided by HD map $\mathcal{M}$, we search the segment sequences along the predecessors and successors of each root segment via Depth-First-Search on $\mathcal{M}$, where the forward-searching distance D_F and backward-searching distance D_B are set to 140 and 20 meters. Following, we concatenate each pair of forward and backward segment sequences and remove redundant ones, and finally, the centerline coordinates of each segment sequence yield a potential path $\mathcal{P}_j \in \mathcal{P}^+$. By using the path search G_{path}, we expect that the resulted path set $\mathcal{P}^+$ would provide sufficient coverage to the future path space of a_{tar}. By statistic, each prediction target in the dataset is associated with 3.04 reachable paths on average.

Trajectory Generation. Given the target vehicle's current state estimation $\mathbf{s}^0_{tar}$ as an initial condition, and the searched potential paths $\mathcal{P}^+$ as dynamic references, our trajectory generator $G_{traj} : (\mathcal{P}^+, \mathbf{s}^0_{tar}, \mathcal{C}) \mapsto \mathcal{T}$ produces the longitudinal movement $s(t)$ and lateral movement $d(t)$ independently by connecting the fixed start state with different end states within the prediction horizon using parametric curves. For the longitudinal movement, we sample the target velocity $\dot{s}(T_F)$ in the range of $[\max(0, \dot{s}_0 - \delta^- T_F), \min(\hat{\dot{s}}, \dot{s}_0 + \delta^+ T_F)]$ with $\hat{\dot{s}} = 30m/s$, $\delta^- = -6m/s^2$, $\delta^+ = 6m/s^2$ and the number of samples is set to 35. For the lateral movement, we sample the target offset $d(T_F)$ in the range of $[-d_{lane}/2, d_{lane}/2]$ and the number of samples is set to 9. Because the in-place lane width cannot be queried from the Argoverse API, we fix d_{lane} to 5 meters in lateral sampling. With the generated longitudinal and lateral trajectory sets $\mathcal{T}_{lon}$ and $\mathcal{T}_{lat}$, a full trajectory $\vec{x}(s(t), d(t))$ is formed by every combinations in $\mathcal{T}_{lon} \times \mathcal{T}_{lat}$. Then we project the Frenét coordinates (s, d) back to global coordinates (x, y), to check trajectory feasibility with respect to environmental constraints $\mathcal{C}_M$ and kinematic constraints $\mathcal{C}_{tar}$. Regarding that neither the static obstacles nor the detailed vehicle information is labeled in the Argoverse Dataset, we omit to check the collision with static obstacles and adopt a general urban sedan setting to ensure dynamic feasibility, with the maximum velocity $v = 33.33m/s$, maximum acceleration/deceleration $\alpha = \pm 8m/s^2$, and curvature $\kappa = 0.33$. If more road information (static obstacles, road boundary, and traffic rules) and vehicle information (bounding box, vehicle category, or rough axle distance) could be accessed, our future trajectory space $\mathcal{T}$ would be further regularized by imposing more detailed constraints. Finally, each prediction target in the dataset obtains 484 feasible trajectories on average.

A.2 Learning-based Evaluator

The prediction evaluator $G : (\mathbf{s}^0_{tar}, \mathcal{M}, \mathcal{C}) \mapsto (\mathcal{P}, \mathcal{T})$ encodes the scene context that includes history track set $\mathcal{S}$, path set $\mathcal{P}$, and future trajectory set $\mathcal{T}$. Argoverse provides the history tracks in $\mathcal{S}$ with the time interval $\Delta T = 0.1$s, so the continuous future trajectories in $\mathcal{T}$ are discretized with the same time interval. All reachable paths in $\mathcal{P}$ are discretized with the distance interval $\Delta D = 2m$. The detailed parameter setting of the evaluation network could be referred to our codebase. We train the evaluation network with a batch size of 64. The network is optimized using Adam with the learning rate initialized as 0.001 and decayed by 10 at every 10 epoch. We use Group Normalization with a group number of 4 for normalizing the data and LeakyReLU for non-linearity. Additionally, we apply global random scaling with the scaling ratio sampled from $0.75 \sim 1.25$ for data augmentation in training.

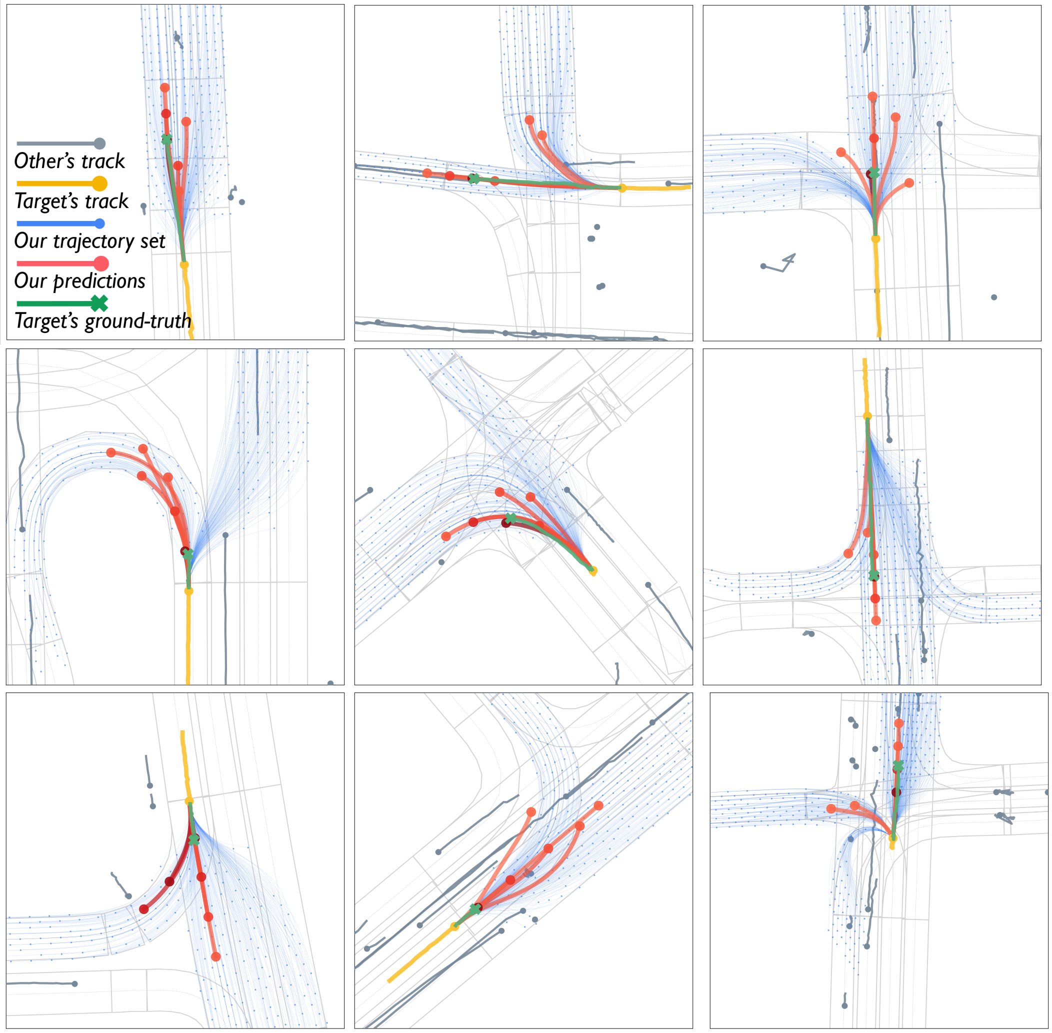
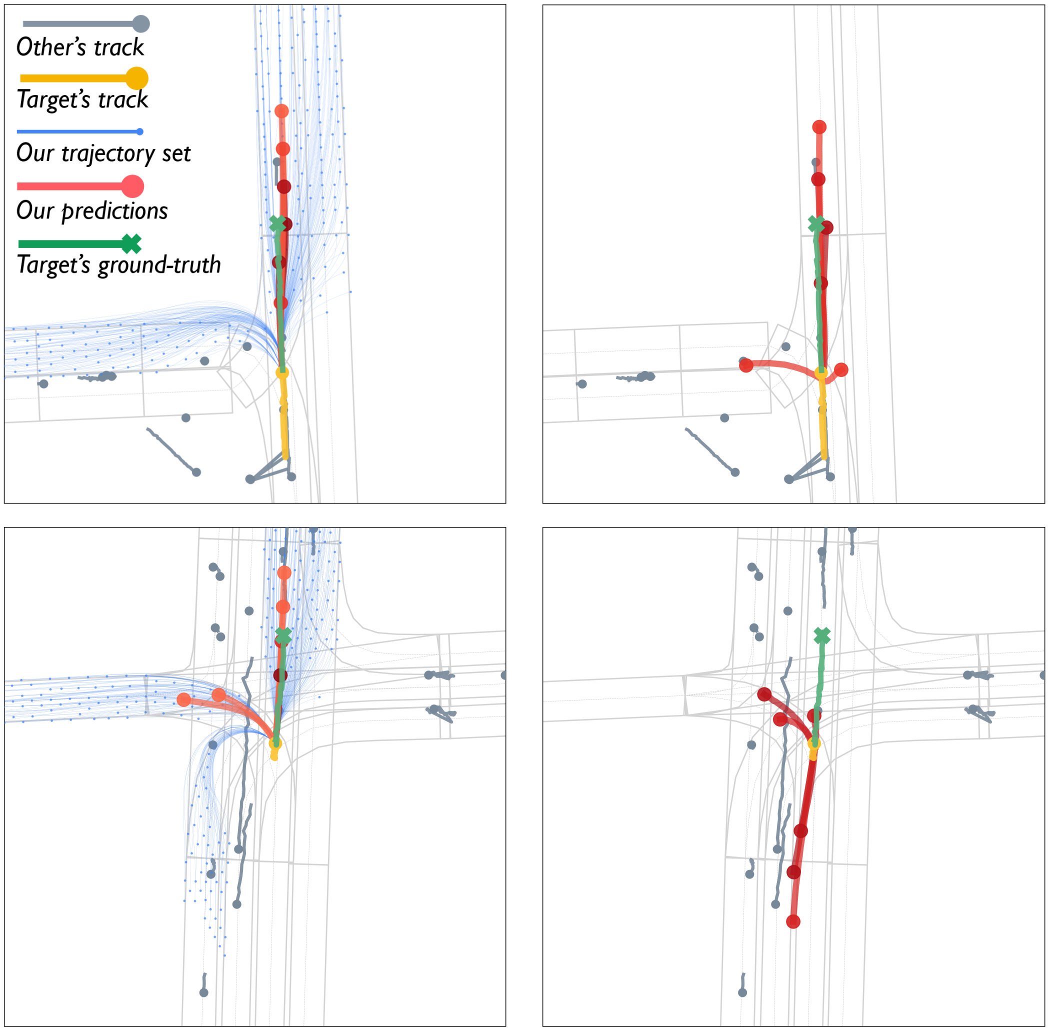


Figure 5: Qualitative results under diverse scenarios on the Argoverse validation set. The HD map is depicted by light grey segments. The other agents' history tracks are shown in steel blue. The target agent's history track is shown in yellow and ground-truth future trajectory in green. The model-based generator produces the set of future trajectories $\mathcal{T}$ (blue) with feasibility guaranteed. The learning-based evaluator selects $K = 6$ trajectories from $\mathcal{T}$ as multimodal prediction results (red), with the depth of red indicating the corresponding trajectory probability.

B Qualitative Analysis

B.1 Results Under Diverse Traffic Scenarios

Fig. 5 presents visualization results of our method under complex traffic scenarios on the Argoverse validation set, which covers different driving speeds (high/low speed), maneuvering modes (overtaking, braking, lane changing, turning, and even U-turn), road scenarios (straight road, T-junction, and crossroad). From all these cases, the future trajectory set $\mathcal{T}$ (blue) reflects that the model-based generator reasonably regularizes the prediction space by imposing environmental and dynamic constraints while providing sufficient coverage for the future trajectory of the target agent. The prediction results $\mathcal{T}_{tar}$ (red) show the learning-based evaluator is capable of assigning weights for different future trajectories in $\mathcal{T}$ by modeling interactions and thereby achieving accurate multimodal future predictions. Altogether, the target's ground-truth trajectory (green) is mostly overlapped with our prediction result (red), demonstrating the effectiveness of our proposed framework.



(a) Ours-PRIME.

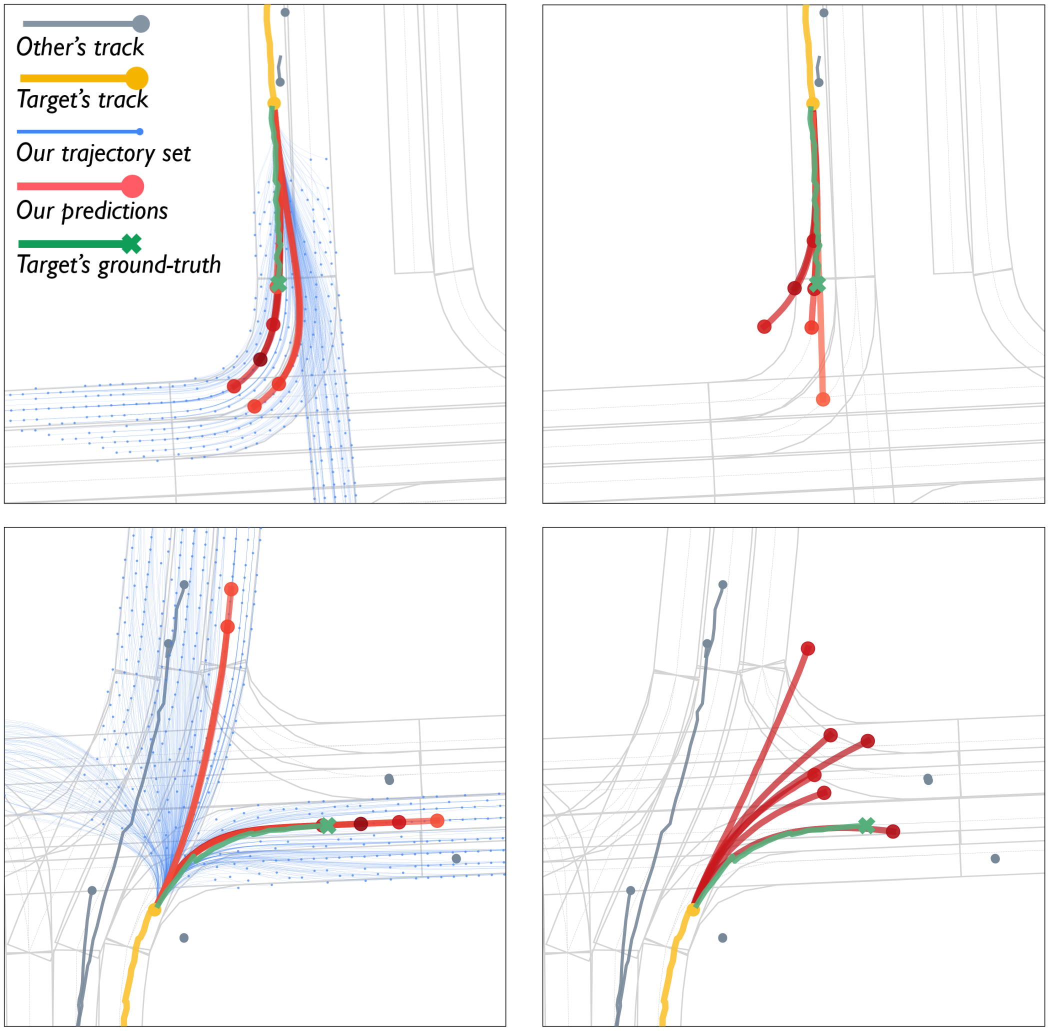
(b) LaneGCN [34].

Figure 6: Qualitative comparisons between ours (*left*) and LaneGCN (*right*) on the Argoverse validation set to show the effect of kinematic constraints.

B.2 Comparison with Fully Learning-based Prediction

Compared with the mainstream learning-based methods that generate unconstrained trajectory predictions by neural networks, the main difference of our proposed PRIME framework is to explicitly constrain the prediction space and thereby ensure trajectory feasibility. Here, we use LaneGCN [34] as a representative for the typical fully learning-based prediction models, considering it makes the best performance on multiple evaluation metrics in Table 1, and among the current state-of-the-art methods, it is open-source. We demonstrate some common failures of kinematically and environmentally infeasible predictions in Fig. 6 and Fig. 7.

Due to kinematic constraints, vehicles cannot take a sudden turn at high speed (1st-row in Fig. 6), or reverse the moving direction (2nd-row in Fig. 6). Also, the prediction results of turning with across lane boundaries (1st-row in Fig. 7), or heading towards reverse lanes (2nd-row in Fig. 7) are incompliant with environmental constraints. Moreover, the counter-intuitive bidirectional trajectories predicted by LaneGCN (2nd-row in Fig. 6) also reveal that the fully learning-based prediction relies on relative long-range tracks for regressing trajectories, but it may degrade under short-range tracks.

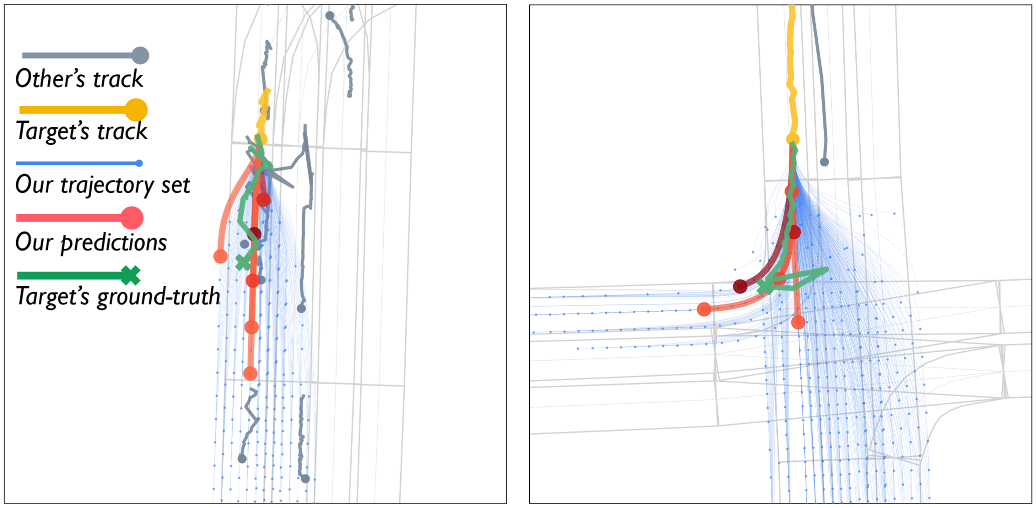


(a) Ours-PRIME.

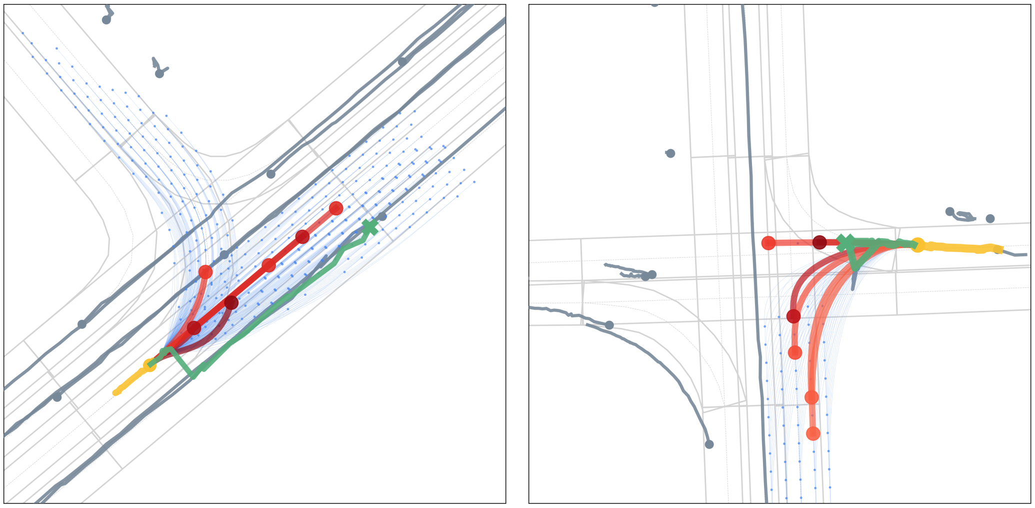
(b) LaneGCN [34].

Figure 7: Qualitative comparisons between ours (*left*) and LaneGCN (*right*) on the Argoverse validation set to show the effect of environmental constraints.

In some of the above examples, although it looks PRIME and LaneGCN show comparable performance when evaluated by minADE_6 and minFDE_6, their impacts on the downstream planning differ a lot. The infeasible trajectories generated by LaneGCN bring massive uncertainty in the predicted future states, which would cause redundant burdens for an autonomous vehicle to make decisions and motion plans. Especially in dense traffic where multiple surrounding vehicles need to be predicted, the negative impact of infeasible predictions would be further aggravated. By contrast, PRIME regularizes the future trajectory space (blue) by given constraints and thus makes accurate and reasonable future predictions (red).



(a) Ground truth trajectory with position oscillation.

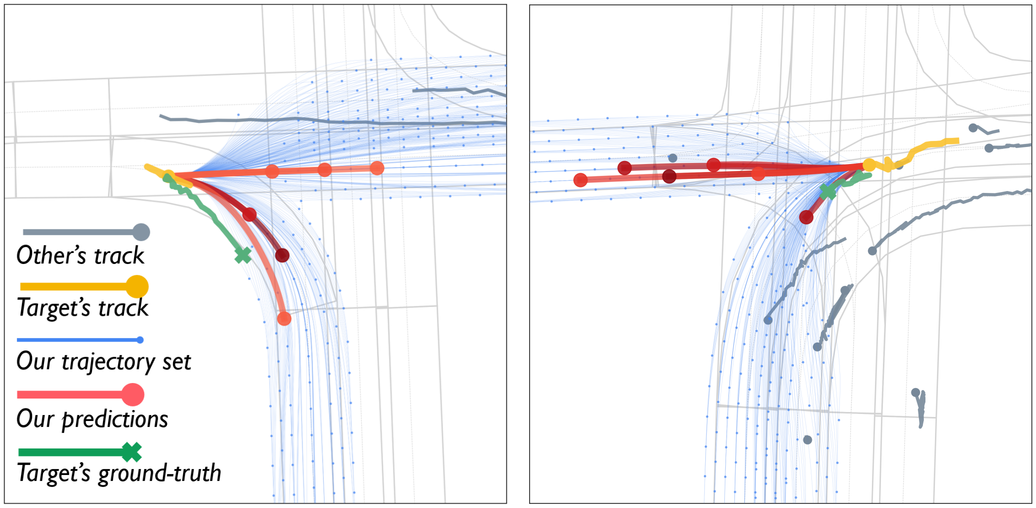


(b) Ground truth trajectory with id switch.

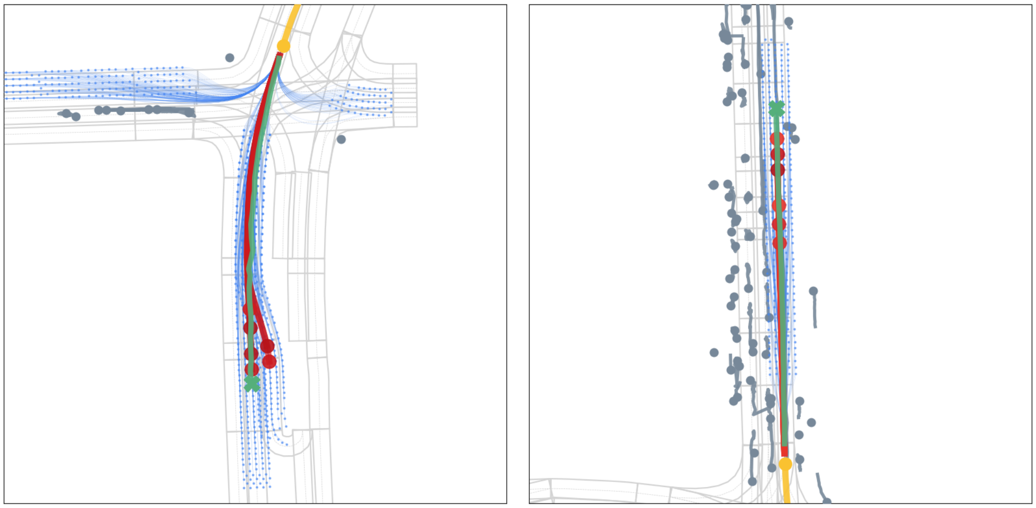
Figure 8: Qualitative results under cases with defective ground truth on the Argoverse validation set. Compared with the ground truth trajectories with position oscillation (*upper*) or id switch (*lower*), the smooth trajectories predicted by PRIME are more realistic and reasonable.

B.3 Impacts Caused by Defect Data

Although Argoverse is one of the most recognized benchmarks for trajectory prediction due to its high-quality trajectory and map annotation, some of its ground truth trajectories are not completely correct. The common issues result from the tracking method used for annotating the data, including position oscillation (Fig. 8a) and id switch (Fig. 8b) that the ground truth trajectory is suddenly switched to a neighboring agent. Such defect cases would lead to worse performance indicators (ADE/FDE-based metrics) of our method in the quantitative evaluation, but it is evident that the smooth trajectories predicted by PRIME are more realistic and reasonable.



(a) Inaccurate heading estimation (left) and velocity estimation (right).



(b) Inaccurate predictions under high-speed driving.

Figure 9: Qualitative analysis for failure cases (minFDE$_6 > 2$m) on the Argoverse validation set. The failures mostly originate from (a) inaccurate state estimation for the history tracks with oscillating positions and (b) large future prediction space under the high-speed scenarios.

B.4 Failure Cases

Lastly, we demonstrate the failure cases on the Argoverse validation set in Fig. 9. The failures are mostly related to the estimation deviation for the target vehicle's current state $\mathbf{s}_{tar}^{0}$ and the large future prediction space under high-speed scenarios.

Although the sampling-based strategy in our generator could compensate for inaccurate state estimation to some extent, estimating the heading and velocity from sequences of centroid positions given in Argoverse would be intractable when serious data noise exists. For example, the position oscillation of a short-distance history track would make the heading direction hard to estimate, as shown in Fig. 9a (left). As a result, the ground truth trajectory locates out of the resulted prediction space's span range. When the position sequence vibrates too much, the accuracy of velocity estimation would even be affected. As exemplified in Fig. 9a (right), the future trajectory space does not cover the ground truth trajectory due to the inaccurate estimation for the target's low velocity, leading to a relatively large displacement error in the prediction results. While in the autonomous driving systems, the vehicle's bounding box given by detection provides geometry information in addition to discrete positions, which would enable more robust and accurate state estimation for prediction targets.

The other type of failure cases occurs in high-speed driving. As illustrated in Fig. 9b, the prediction target moves towards its forward open space at high speed. Its 3-second future trajectory space is much larger and naturally leads to a higher probability of missed predictions ($\text{minFDE}_6 > 2\text{m}$). Nonetheless, it could be observed that our predictions, locating within a compact feasible trajectory space, accurately capture the target's intention with an acceptable displacement error, which makes sense for the downstream decision-making and planning.

C Runtime Analysis

The inference frequency of our prediction framework depends on the scene complexity, sampling density, and computing power. Running with Intel i7-7820X, the generation of a single trajectory with a single thread spends $0.1 \sim 0.2$ ms on average. With each trajectory sample produced independently, the model-based trajectory generator could be highly parallelized to provide full coverage to the future prediction space with satisfactory real-time performance. For the learning-based evaluator, it is implemented by a lightweight network with only 1.02 million parameters. Its inference time on NVIDIA 2080TI is $8 \sim 12$ ms. Overall, the whole framework of PRIME could well satisfy the real-time requirements for autonomous driving.

D Limitation and Future Work

The framework could be further improved from the following aspects. We use some fixed parameters in the model-based generator, but better strategies can be applied when the required information is given. Firstly, the distance thresholds in the path search phase could be adjusted according to the target vehicle's state, and the resulted paths could be pruned by considering the lane connectivity given by curbs, fences, etc. Secondly, the trajectory generation phase could be refined by adjusting the lateral and longitudinal sampling boundary based on the speed limits and in-place lane width, and adopting different sampling densities according to the target's impact (e.g., distance) on the autonomous vehicle. All these adjustments would contribute to alleviating the computational cost in the model-based generator. For the learning-based evaluator, separating the function of trajectory generation enables it to achieve good performance using a lightweight network with 1.02M parameters, which also leaves space for optimizing the network structure. We plan to extend scene encoding from reachable paths to a lane graph (as proposed in VectorNet [32] and LaneGCN [33]), where a complete context encoding is expected to bring performance improvement.