EARefiner: An Endpoint-Aligned Diffusion Trajectory Refiner via Schrödinger Bridges for Multi-Agent Trajectory Prediction

Yuxi Xue[1] **Xiangzheng Zhou**[1] **Xiaobo Chen**[3,*] **Jianjun Qian**[1,*] **Jian Yang**[1,2,*]

[1]PCA Lab, School of Computer Science and Engineering,
Nanjing University of Science and Technology
[2]PCA Lab, School of Intelligence Science and Technology, Nanjing University
[3]School of Computer Science and Technology, Shandong Technology and Business University

{yxxue,xzhou,csjqian,csjyang}@njust.edu.cn xbchen82@gmail.com

Abstract

Accurate multi-agent trajectory prediction is essential for autonomous driving safety. Although existing forecasting models can generate plausible future trajectories, their predictions may still contain substantial errors. Diffusion-based trajectory refinement provides a promising approach to improving these predictions by progressively correcting coarse trajectory hypotheses. However, existing coarse-centered diffusion refiners can suffer from a training–inference endpoint mismatch: the forward process terminates at a noisy distribution around the coarse prediction, whereas reverse refinement starts from the deterministic coarse trajectory, potentially degrading refinement performance. To resolve this issue, this paper proposes EARefiner, an endpoint-aligned diffusion trajectory refiner. By introducing a tractable diffusion Schrödinger bridge with dual-terminal constraints, EARefiner ensures that the terminal distribution of the forward process collapses to the deterministic coarse prediction used at inference. This theoretical alignment eliminates the endpoint mismatch and enables refinement in a coarse-centered residual space. Based on the semi-linear structure of EARefiner, we derive tailored first- and second-order DPM-Solver updates to accelerate the sampling process. Extensive experiments on the ETH/UCY, SDD, NBA, Argoverse 1, and Argoverse 2 benchmarks demonstrate that EARefiner serves as an effective plug-and-play module. It consistently improves the prediction accuracy of various baseline architectures, while the proposed solver achieves competitive accuracy with substantially fewer denoiser evaluations. Our code is publicly available at https://github.com/XYX512/EARefiner.

1 Introduction

Accurately forecasting the future trajectories of interacting agents from historical observations is fundamental to motion understanding and decision-making in dynamic environments [1, 2]. With the advancement of artificial intelligence, multi-agent trajectory prediction has gradually shifted from physics-driven models toward data-driven approaches that learn complex motion patterns from observed data [3, 4]. In particular, Graph Neural Networks and Transformers have been widely adopted to capture temporal dynamics, inter-agent interactions, and contextual dependencies [5, 6, 7, 8, 9].

*Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

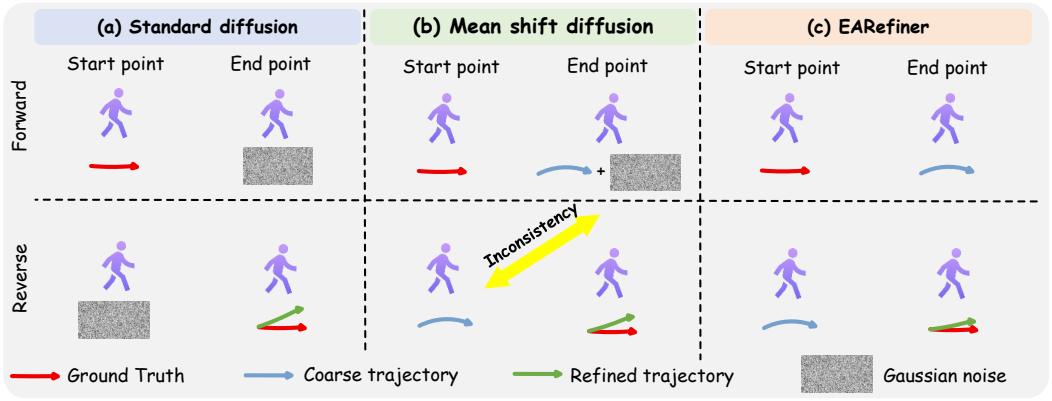

Figure 1: Comparison of standard diffusion, mean-shift diffusion, and EARefiner. Standard diffusion terminates at a noise prior, while mean-shift diffusion retains nonzero terminal variance, causing a training–inference endpoint mismatch. In contrast, EARefiner employs a tractable Schrödinger bridge whose terminal distribution collapses to the deterministic coarse prediction, ensuring exact endpoint alignment.

However, due to uncertain agent intentions and complex environments, future motion is inherently multimodal, such that the same observation history may correspond to multiple physically plausible trajectories [10, 11, 12]. To account for this uncertainty, modern predictors typically generate multiple candidate trajectories through either discriminative multi-hypothesis modeling or generative approaches. In particular, Generative Adversarial Networks and Conditional Variational Autoencoders have been widely adopted to model diverse future trajectories [13, 14, 15]. More recently, diffusion models have emerged as a promising generative approach, demonstrating strong capabilities in modeling complex trajectory distributions and producing physically and socially plausible predictions [16].

Despite these advances in multimodal trajectory generation, producing diverse and plausible predictions does not guarantee the accuracy of individual trajectory hypotheses, which may still contain substantial geometric and endpoint errors. This motivates trajectory refinement as a complementary approach to improving existing predictions. Diffusion models are well suited to this task because their iterative denoising process can progressively correct imperfect trajectories. However, sampling from pure noise is computationally expensive, motivating diffusion-based refinement methods that leverage coarse predictions from existing forecasting models as informative priors. LED [17] uses coarse trajectories generated by neural networks as sampling starting points to accelerate inference while maintaining prediction quality, illustrating the potential of diffusion-based refinement. Building on this paradigm, Diff-Refiner [18] explicitly integrates coarse predictions into the diffusion Markov chain. However, this coarse-centered diffusion process can introduce a training–inference endpoint mismatch, as illustrated in Figure 1: the forward process terminates at a noisy distribution around the coarse prediction, whereas reverse refinement starts from the deterministic coarse trajectory. This distribution mismatch may propagate through reverse integration and degrade refinement performance.

To address this endpoint mismatch, we propose EARefiner, an endpoint-aligned diffusion trajectory refiner. Specifically, we adopt the tractable diffusion Schrödinger bridge formulation of I^2SB [19] for trajectory refinement, taking the ground-truth future trajectory and the deterministic coarse prediction as its two endpoints. This formulation ensures that the forward-process terminal distribution collapses to the deterministic coarse prediction used to initialize inference. This endpoint alignment eliminates the training–inference boundary mismatch by construction and enables refinement in a coarse-centered residual space. Practically, encoded coarse and historical trajectories condition a UNet for noise prediction. Finally, we derive tailored first- and second-order DPM-Solver updates based on EARefiner's semi-linear residual probability-flow dynamics to enable efficient few-step refinement.

In this paper, we make three contributions. (1) We identify and formalize a training–inference endpoint mismatch in diffusion-based trajectory refinement and analyze its distributional discrepancy and reverse-flow sensitivity. (2) We specialize the tractable I^2SB formulation to trajectory refinement by assigning the ground-truth trajectory and deterministic coarse prediction as the two exact endpoints, thereby aligning the training endpoint with the inference initialization. (3) We derive bridge-specific

first- and second-order DPM-Solvers for efficient few-step refinement and validate EARefiner across diverse trajectory forecasting benchmarks and backbones.

2 Related Work

2.1 Multimodal Trajectory Prediction

Multi-agent trajectory prediction must account for uncertainty and multimodality in future motion. Discriminative approaches typically represent multiple motion modes and their probabilities through task-specific architectures [20, 21, 22, 23], while generative approaches model distributions over plausible future trajectories. Diffusion-based trajectory predictors have demonstrated promising capabilities in multimodal generation. For example, CGD [24] incorporates motion intentions and agent interactions into noise estimation, while [25] employs residual polar coordinates and learnable endpoints to capture multimodal motion patterns. Flow-based methods such as MoFlow [26] provide another generative alternative with efficient sampling. Despite these advances, different forecasting paradigms may still produce trajectory hypotheses with substantial prediction errors, motivating a model-agnostic refinement framework that can improve predictions across diverse baseline architectures.

2.2 Diffusion-based Trajectory Refinement

Diffusion-based trajectory refinement aims to improve existing predictions by leveraging coarse trajectories as informative priors rather than generating trajectories entirely from noise. LED [17] initializes denoising from learned trajectory estimates, while Diff-Refiner [18] further integrates coarse predictions into a mean-shift diffusion process for plug-and-play refinement. However, such coarse-centered diffusion processes can introduce a training–inference endpoint mismatch: the forward process terminates at a noisy distribution around the coarse prediction, whereas reverse refinement starts from the deterministic coarse trajectory. This discrepancy motivates a refinement process that explicitly aligns its training endpoint with the inference initialization. I^2SB [19] provides a tractable diffusion Schrödinger bridge formulation with closed-form Gaussian intermediate distributions conditioned on prescribed endpoints. Building on this formulation, EARefiner specializes the bridge to trajectory refinement by assigning the ground-truth future trajectory and deterministic coarse prediction as its two endpoints. Unlike I^2SB, which establishes the tractable bridge construction, our work identifies and analyzes the endpoint mismatch in diffusion-based trajectory refinement and leverages the bridge's exact boundary constraints to eliminate this mismatch by construction.

2.3 Diffusion Model Sampling Acceleration

Diffusion models [27] typically rely on iterative denoising processes that require numerous neural function evaluations (NFEs), resulting in considerable computational overhead. To improve sampling efficiency, DDIM [28] introduces a non-Markovian sampling process that enables deterministic generation with fewer steps, while Progressive Distillation [29] reduces sampling steps through teacher–student distillation. Continuous-time diffusion formulations establish the connection between stochastic differential equations (SDEs) and probability-flow ordinary differential equations (PF-ODEs) [30], enabling efficient numerical integration. Building on this formulation, DPM-Solver [31] exploits the semi-linear structure of diffusion PF-ODEs to achieve high-order integration with fewer denoising evaluations. However, the endpoint-aligned Schrödinger bridge in EARefiner follows different signal and variance schedules from standard diffusion processes, requiring corresponding adaptations of the numerical solver. Therefore, we derive bridge-specific first- and second-order DPM-Solver updates directly from the coarse-centered residual PF-ODE to enable efficient few-step trajectory refinement.

3 Preliminaries

3.1 Diffusion Models

Diffusion models construct a perturbation process that gradually transforms a data distribution into a tractable reference distribution. For a time-dependent diffusion coefficient β_t and drift f_t, a forward

stochastic differential equation (SDE) and its associated reverse-time SDE can be written as

$$\mathrm{d}Z_t = f_t(Z_t)\,\mathrm{d}t + \sqrt{\beta_t}\,\mathrm{d}W_t, \tag{1}$$

$$\mathrm{d}Z_t = \left[f_t(Z_t) - \beta_t \nabla_{Z_t} \log p_t(Z_t)\right]\mathrm{d}t + \sqrt{\beta_t}\,\mathrm{d}\bar{W}_t, \tag{2}$$

Here, W_t and $\bar{W}_t$ represent the standard Wiener process and its reverse-time counterpart, respectively, while $f_t(Z_t)$ and β_t denote the predefined drift and diffusion coefficients. Because the exact score function $\nabla_{Z_t} \log p_t(Z_t)$ remains analytically intractable, a neural network can be trained to estimate the score, commonly through a noise-prediction parameterization $\epsilon_\theta(Z_t, t)$. By numerically integrating this reverse SDE, the model effectively recovers the underlying data manifold from pure noise.

3.2 Tractable Diffusion Schrödinger Bridge

A Schrödinger bridge connects prescribed boundary distributions relative to a reference stochastic process. I^{2}SB provides a tractable diffusion-bridge construction whose intermediate conditional marginals have closed-form Gaussian expressions when both endpoint states are specified. For the drift-free reference process $\mathrm{d}Z_t = \sqrt{\beta_t}\,\mathrm{d}W_t$ over $t \in [0, 1]$, the intermediate conditional distribution given two fixed endpoints has a mean interpolating between those endpoints and a variance that vanishes at either boundary. We use this existing bridge formulation as the theoretical basis for endpoint-aligned trajectory refinement; its application-specific endpoints and the associated sampling solver are presented next.

4 Method

Figure 2 illustrates the overall framework of EARefiner. During the forward process, the method constructs a diffusion bridge originating from the ground-truth future trajectory and terminating at the coarse prediction generated by the baseline model. The reverse process initiates from this coarse prediction. By jointly encoding historical trajectories and baseline predictions to estimate the noise, the model iteratively refines the initial state to generate a refined future trajectory.

4.1 Problem Formulation

Let $X = \{X^i\}_{i=1}^N$ denote the observed trajectories of N agents, with $X^i \in \mathbb{R}^{T_{\mathrm{obs}} \times 2}$ and observation length T_{obs}. A pretrained forecaster Φ generates K coarse hypotheses per agent over T_{fut} future steps:

$$\mathcal{Y}_{\mathrm{co}} = \Phi(X) = \{Y_{\mathrm{co}}^{i,k}\}_{i=1,k=1}^{N,K}, \quad Y_{\mathrm{co}}^{i,k} \in \mathbb{R}^{T_{\mathrm{fut}} \times 2}. \tag{3}$$

Here $Y_{\mathrm{gt}}^i \in \mathbb{R}^{T_{\mathrm{fut}} \times 2}$ is the ground-truth future trajectory. EARefiner refines each hypothesis while preserving K. Agent and mode indices are omitted below, yielding the generic pair $(Y_{\mathrm{gt}}, Y_{\mathrm{co}})$.

4.2 Endpoint-Aligned Diffusion Schrödinger Bridge Formulation

As in diffusion-based restoration, trajectory refinement uses the coarse prediction Y_{co} as an informative prior rather than generating from scratch. Mean-shift diffusion incorporates this prior into the forward process as

$$Y_t = \sqrt{\overline{\alpha}_t} Y_{\mathrm{gt}} + (1 - \sqrt{\overline{\alpha}_t}) Y_{\mathrm{co}} + \sqrt{1 - \overline{\alpha}_t}\epsilon, \tag{4}$$

where $\overline{\alpha}_t$ is a predefined variance schedule, Y_{gt} is the ground-truth future trajectory, and ϵ is noise sampled from a standard Gaussian distribution.

Proposition 1 (Endpoint mismatch and reverse-flow sensitivity). Suppose the forward mean-shift process has a vanishing terminal signal coefficient ($\sqrt{\overline{\alpha}_1} = 0$) and a nonzero terminal noise scale $\gamma_1 > 0$, while inference initializes from the deterministic coarse prediction. For $d = 2T_{\mathrm{fut}}$, its conditional terminal measures satisfy

$$P_{\mathrm{train}} = \mathcal{N}(Y_{\mathrm{co}}, \gamma_1^2 \mathbf{I}), \quad P_{\mathrm{infer}} = \delta_{Y_{\mathrm{co}}}, \quad W_2^2(P_{\mathrm{train}}, P_{\mathrm{infer}}) = d\gamma_1^2. \tag{5}$$

If the reverse probability-flow ODE vector field is $L(\tau)$-Lipschitz, two reverse solutions with terminal difference e_1 satisfy

$$\|e_t\|_2 \le \|e_1\|_2 \exp\left(\int_t^1 L(\tau)\,\mathrm{d}\tau\right). \tag{6}$$

This bound characterizes sensitivity to endpoint perturbations; it does not imply inevitable error amplification or negative refinement. The proof is provided in Appendix A.1.1. The Wasserstein term measures discrepancy between conditional endpoint distributions, rather than the accuracy of an individual predicted trajectory. The sensitivity bound also explains why initialization matters when integrating the learned reverse dynamics.

To remove this endpoint discrepancy, we adopt the tractable diffusion Schrödinger bridge formulation of I^2SB, specializing its two prescribed endpoints to the ground-truth future trajectory Y_{gt} and deterministic coarse prediction Y_{co}. This specialization uses its drift-free reference process and closed-form Gaussian bridge marginals. We define the forward cumulative variance $\sigma_t^2 = \int_0^t \beta_\tau d\tau$ and the backward cumulative variance $\bar{\sigma}_t^2 = \int_t^1 \beta_\tau d\tau$. Following the tractable bridge formulation, the marginal probability density $q(X_t, t)$ can be expressed in terms of two Schrödinger potentials:

$$q(X_t, t) = \Psi(X_t, t)\overline{\Psi}(X_t, t). \tag{7}$$

To anchor the evolution path strictly to the ground-truth trajectory Y_{gt} and the predicted trajectory Y_{co}, we apply double-end conditioning. For a given boundary pair $(Y_{\mathrm{gt}}, Y_{\mathrm{co}})$, the conditional posterior distribution of the system at time t is expressed as follows:

$$q(Y_t|Y_{\mathrm{gt}}, Y_{\mathrm{co}}) \propto \exp\left(-\frac{1}{2}\left(\frac{\|Y_t - Y_{\mathrm{gt}}\|^2}{\sigma_t^2} + \frac{\|Y_t - Y_{\mathrm{co}}\|^2}{\bar{\sigma}_t^2}\right)\right). \tag{8}$$

The intermediate state at any time step in the diffusion Schrödinger bridge system follows:

$$q(Y_t|Y_{\mathrm{gt}}, Y_{\mathrm{co}}) = \mathcal{N}(\omega_t^{\mathrm{gt}} Y_{\mathrm{gt}} + \omega_t^{\mathrm{co}} Y_{\mathrm{co}}, \Sigma_t \mathbf{I}), \tag{9}$$

where the coefficients are defined as follows:

$$\omega_t^{\mathrm{gt}} = \frac{\bar{\sigma}_t^2}{\sigma_t^2 + \bar{\sigma}_t^2}, \quad \omega_t^{\mathrm{co}} = \frac{\sigma_t^2}{\sigma_t^2 + \bar{\sigma}_t^2}, \quad \Sigma_t = \frac{\sigma_t^2 \bar{\sigma}_t^2}{\sigma_t^2 + \bar{\sigma}_t^2}. \tag{10}$$

The mean coefficients sum to one, shifting the conditional mean from ground truth toward the coarse hypothesis over time. The bridge variance is positive at interior times and vanishes at both endpoints. At the endpoints, the bridge marginals collapse to $\delta_{Y_{\mathrm{gt}}}$ at $t = 0$ and $\delta_{Y_{\mathrm{co}}}$ at $t = 1$, aligning the training terminal marginal with the deterministic inference initialization. This removes the boundary mismatch, but does not remove denoiser or numerical approximation errors.

Proposition 2 (Exact endpoint alignment). For the conditional Gaussian bridge in Equation 9, assuming $\beta_t > 0$ and finite cumulative variance on $[0, 1]$, we have

$$q(Y_0 \mid Y_{\mathrm{gt}}, Y_{\mathrm{co}}) = \delta_{Y_{\mathrm{gt}}}, \qquad q(Y_1 \mid Y_{\mathrm{gt}}, Y_{\mathrm{co}}) = \delta_{Y_{\mathrm{co}}}. \tag{11}$$

The result guarantees equality of the terminal measures, not an accuracy bound or an improvement guarantee for individual predictions. The proof is provided in Appendix A.1.2.

To reverse the continuous evolution path, we construct a neural network $\epsilon_\theta(Y_t, Y_{\mathrm{co}}, X, t)$ to fit the noise added during the evolution. Following the denoising score matching principle, the end-to-end optimization objective is:

$$\mathcal{L}(\theta) = \mathbb{E}_{t, Y_{\mathrm{gt}}, Y_{\mathrm{co}}, \epsilon}[\|\epsilon_\theta(Y_t, Y_{\mathrm{co}}, X, t) - \epsilon\|_2^2]. \tag{12}$$

We denote the noise prediction at time t by $\hat{\epsilon}_t = \epsilon_\theta(Y_t, Y_{\mathrm{co}}, X, t)$. Since the analytical posterior $q(Y_t|Y_{\mathrm{gt}}, Y_{\mathrm{co}})$ remains Gaussian, we adopt the DDIM approach to implement sampling. Specifically, we first estimate the ground truth $\hat{Y}_{\mathrm{gt}}$ given the current state and predicted noise:

$$\hat{Y}_{\mathrm{gt}} = \frac{1}{\omega_t^{\mathrm{gt}}}(Y_t - \omega_t^{\mathrm{co}} Y_{\mathrm{co}} - \sqrt{\Sigma_t}\hat{\epsilon}_t). \tag{13}$$

We then transition from time step t to a state $s < t$:

$$Y_s = \omega_s^{\mathrm{gt}} \hat{Y}_{\mathrm{gt}} + \omega_s^{\mathrm{co}} Y_{\mathrm{co}} + \sqrt{\Sigma_s}\hat{\epsilon}_t + z. \tag{14}$$

For trajectory refinement, the coefficient of the random noise term z is set to zero. The exact $t = 1$ boundary is deterministic; numerical solver coefficients that become singular at this endpoint are evaluated at a nearby interior time, as detailed in Appendix A.2.

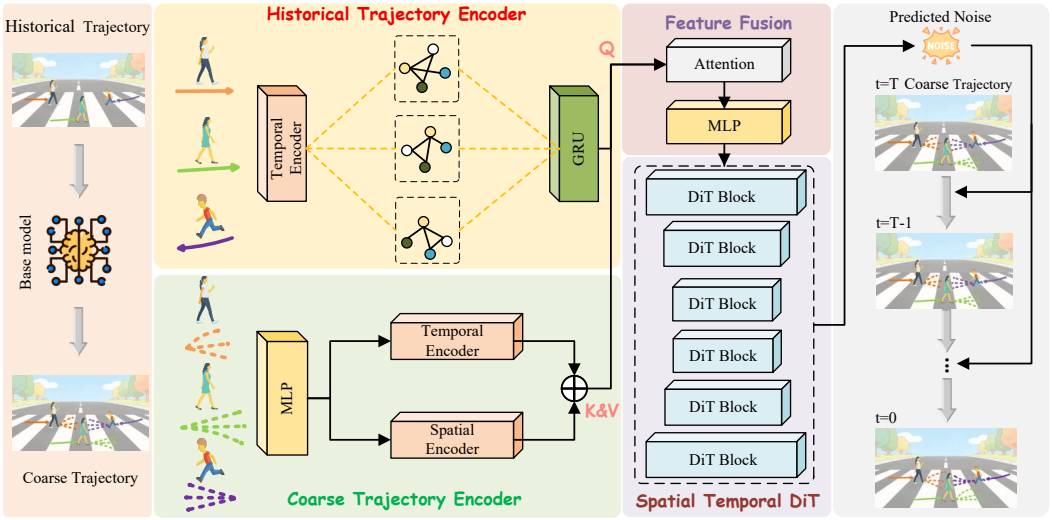

Figure 2: Illustration of the EARefiner architecture. The historical trajectory encoder and the coarse trajectory encoder process the historical trajectories and the baseline predictions, respectively. An attention mechanism fuses these encoded features and feeds them into a UNet for noise prediction. Guided by the predicted noise, the sampling process initiates from the coarse trajectory to generate the refined trajectory.

4.3 Denoising Network Architecture

To predict the noise for sampling and acquiring refined trajectories, we encode the historical trajectories and coarse predicted trajectories separately before feeding them into a UNet for noise prediction. Specifically, for the historical trajectory of the i-th agent, a temporal encoder composed of a multilayer perceptron (MLP) and an attention mechanism extracts the temporal context features:

$$H^i = \text{Attention}(\text{MLP}(X^i)). \tag{15}$$

Subsequently, the temporal context features H of all agents and the corresponding adjacency matrix A are fed into a graph neural network to derive the spatial context S:

$$S = \text{Graph}(H, A). \tag{16}$$

A Gated Recurrent Unit (GRU) fuses the acquired temporal and spatial contexts to formulate the comprehensive historical spatiotemporal context C^{h}:

$$C^{\text{h}} = \text{GRU}(H, S). \tag{17}$$

Similarly, an MLP and an attention mechanism extract the temporal context features for the coarse predicted trajectories:

$$F^i = \text{Attention}(\text{MLP}(Y^i_{\text{co}})). \tag{18}$$

Because the future interactive relationships among agents remain unknown, an attention mechanism is still employed as the spatial encoder to capture the corresponding spatial context, which is then concatenated with the temporal context to form the final complete spatiotemporal context C^{f}.

We encode the historical and coarse predicted trajectories to extract their respective context representations C^{h} and C^{f}. By employing C^{h} as the query and C^{f} as the key and value, we compute the final encoded features. These features are subsequently processed by a UNet composed of DiT blocks to predict the final noise:

$$\epsilon_\theta(Y_t, Y_{\text{co}}, X, t) = \text{UNet}\big(Y_t, t; \text{Attention}(C^{\text{h}}, C^{\text{f}})\big). \tag{19}$$

The two feature streams are complementary: the historical encoder summarizes observed motion and interactions, while the coarse encoder describes hypotheses from the backbone. Their fusion conditions noise estimation on both information sources.

4.4 DPM-Solver for EARefiner

Although DDIM-based deterministic sampling improves speed through time-step skipping, its accuracy can degrade with large step sizes. In autonomous driving, where real-time performance is critical and long-span trajectory refinement must be completed in few steps, large time steps introduce significant local truncation errors. To maintain high fidelity with minimal inference steps, we derive a DPM-Solver for EARefiner to perform precise higher-order integration. According to diffusion theory, the forward process of EARefiner corresponds to a Probability Flow ODE (PF-ODE) with an identical marginal distribution. Using the identity $\omega_t^{\mathrm{gt}} + \omega_t^{\mathrm{co}} = 1$, we transform Equation 9 into a relative residual space centered at Y_{co} as follows:

$$Y_t - Y_{\mathrm{co}} = \omega_t^{\mathrm{gt}}(Y_{\mathrm{gt}} - Y_{\mathrm{co}}) + \sqrt{\Sigma_t}\,\epsilon. \tag{20}$$

In this residual space, the bridge marginal has a diffusion-like signal-plus-noise form, where ω_t^{gt} acts as a signal scaling coefficient. This mapping allows us to perform exact integration on the semi-linear PF-ODE via variation of constants. Defining log-SNR $\lambda_t = \ln(\omega_t^{\mathrm{gt}}/\sqrt{\Sigma_t})$, we define the step size $h = \lambda_s - \lambda_t$ to derive the updated state Y_s for $s < t$. We instantiate the solver in first-order and second-order forms. Assuming the denoising network is constant within a local interval, the first-order update is:

$$Y_s = \frac{\omega_s^{\mathrm{gt}}}{\omega_t^{\mathrm{gt}}}(Y_t - Y_{\mathrm{co}}) + Y_{\mathrm{co}} - \sqrt{\Sigma_s}(e^h - 1)\epsilon_\theta(Y_t, Y_{\mathrm{co}}, X, t). \tag{21}$$

To reduce local truncation error at large step sizes, we introduce a midpoint method for a second-order approximation of the network derivative. Defining the midpoint $\lambda_m = (\lambda_t + \lambda_s)/2$ and the corresponding intermediate state Y_m, the second-order update formulas are:

$$Y_m = \frac{\omega_m^{\mathrm{gt}}}{\omega_t^{\mathrm{gt}}}(Y_t - Y_{\mathrm{co}}) + Y_{\mathrm{co}} - \sqrt{\Sigma_m}(e^{h/2} - 1)\hat{\epsilon}_t, \tag{22}$$

$$Y_s = \frac{\omega_s^{\mathrm{gt}}}{\omega_t^{\mathrm{gt}}}(Y_t - Y_{\mathrm{co}}) + Y_{\mathrm{co}} - \sqrt{\Sigma_s}(e^h - 1)\hat{\epsilon}_t - \sqrt{\Sigma_s}\left(\frac{e^h - 1}{h} - 1\right) \cdot 2(\hat{\epsilon}_m - \hat{\epsilon}_t), \tag{23}$$

where $\hat{\epsilon}_t = \epsilon_\theta(Y_t, Y_{\mathrm{co}}, X, t)$ and $\hat{\epsilon}_m = \epsilon_\theta(Y_m, Y_{\mathrm{co}}, X, m)$. These formulas indicate that EARefiner fast sampling performs an exact exponential integration on the linear part of the residual probability-flow ODE. Under standard smoothness assumptions, the second-order approximation has local truncation error $O(h^3)$ away from singular endpoints; this is a solver accuracy statement, not a guarantee of ground-truth prediction accuracy. (*The detailed derivation of the first-order and second-order DPM-Solver sampling formulas for EARefiner is provided in Appendix A.2.*) The midpoint prediction estimates changes in denoiser output over the integration interval. Accordingly, the second-order update requires two network evaluations per numerical step, whereas the first-order update requires only one. To avoid numerical singularities at $t = 1$, the solver evaluates its coefficients at a nearby interior time.

5 Experiments

5.1 Datasets and Evaluation Metrics

We evaluate EARefiner on five public trajectory prediction benchmarks, including ETH/UCY [32], SDD [33], NBA SportVU [34], Argoverse 1 [35], and Argoverse 2 [36], covering pedestrian, sports, and autonomous-driving scenarios. ETH/UCY comprises five subsets: ETH, HOTEL, UNIV, ZARA1, and ZARA2. For ETH/UCY and SDD, we observe 8 historical steps and predict 12 future steps with $K = 20$ hypotheses. NBA uses 10 observed and 20 predicted steps (2.0/4.0 s) with $K = 20$, while Argoverse 1 and Argoverse 2 use 20/30 steps (2.0/3.0 s) and 50/60 steps (5.0/6.0 s), respectively, with $K = 6$. We evaluate prediction accuracy using minimum average displacement error (minADE) and minimum final displacement error (minFDE) over the K predicted trajectories, both measured in meters. For the autonomous-driving benchmarks, we also report miss rate (MR).

5.2 Baseline Models and Implementation Details

We evaluate EARefiner with diverse trajectory prediction backbones, including STGCNN [37], Gatraj [38], and PCHGCN [39] on ETH/UCY and SDD, LED on NBA, HiVT [40] on Argoverse 1, and QCNet [41] on Argoverse 2. These backbones cover graph-based, diffusion-based, and autonomous-driving trajectory prediction architectures. All experiments are implemented in PyTorch and conducted on an RTX 4090 GPU. EARefiner is trained for 1,000 epochs on ETH/UCY, SDD, and NBA, and 100 epochs on Argoverse 1 and Argoverse 2, using the Adam optimizer with a learning rate of 5×10^{-4}, a batch size of 128, and a feature dimension of 128. The diffusion bridge adopts 300 steps from a 302-step mirrored linear variance schedule with $\beta \in [10^{-3}, 3 \times 10^{-2}]$. For numerical stability, the solver evaluates its coefficients at a baseline-dependent interior cutoff $t_{\max} < 1$.

5.3 Quantitative Results

Table 1: Quantitative results of baseline models with and without EARefiner on the ETH and UCY datasets ($K = 20$). Red indicates the lowest minADE/minFDE values within each backbone group (including ties); green marks values worse than the corresponding backbone.

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2
STGCNN	0.742/1.230	0.414/0.689	0.492/0.907	0.331/0.521	0.305/0.479
STGCNN + Diff-Refiner	0.683/1.182	0.332/0.631	0.462/0.934	0.270/0.512	0.233/0.461
STGCNN + EARefiner	0.674/1.177	0.312/0.583	0.453/0.906	0.262/0.501	0.222/0.429
Gatraj	0.263/0.402	0.107/0.166	0.206/0.375	0.164/0.294	0.116/0.209
Gatraj + Diff-Refiner	0.256/0.395	0.105/0.163	0.208/0.391	0.159/0.290	0.117/0.213
Gatraj + EARefiner	0.253/0.393	0.103/0.158	0.205/0.373	0.159/0.289	0.116/0.208
PCHGCN	0.428/0.681	0.173/0.286	0.216/0.382	0.174/0.305	0.130/0.228
PCHGCN + Diff-Refiner	0.411/0.662	0.165/0.277	0.218/0.389	0.178/0.301	0.135/0.231
PCHGCN + EARefiner	0.390/0.637	0.158/0.256	0.216/0.381	0.169/0.291	0.129/0.223

Table 2: Refinement performance on SDD using PCHGCN ($K = 20$, seed 66); red indicates the best values.

Method	minADE	minFDE
PCHGCN	0.305	0.504
PCHGCN + EARefiner	0.294	0.482

Table 3: Refinement performance on NBA using LED ($K = 20$); entries are minADE/minFDE, and red indicates the best values.

Horizon	LED	LED + EARefiner
1 s	0.177/0.269	0.173/0.266
2 s	0.369/0.565	0.363/0.558
3 s	0.582/0.837	0.573/0.829
4 s	0.810/1.096	0.796/1.079

Table 1 demonstrates the effectiveness of EARefiner across three representative forecasting backbones on ETH/UCY. By refining coarse trajectory hypotheses, EARefiner consistently improves the overall prediction accuracy of STGCNN, Gatraj, and PCHGCN. Particularly notable improvements are observed for STGCNN, where minADE decreases by 24.6% on HOTEL and 27.2% on ZARA2. Moreover, EARefiner further enhances the competitive performance of Gatraj and PCHGCN, demonstrating its ability to improve trajectory quality across different baseline architectures and prediction accuracy levels. On SDD, EARefiner reduces the PCHGCN minADE/minFDE from $0.305/0.504$ to $0.294/0.482$ (Table 2), demonstrating its effectiveness in complex environments with dense crowds and diverse pedestrian interactions. These results further validate the applicability of EARefiner as a plug-and-play trajectory refinement module.

To assess the versatility of EARefiner across different generative architectures, we integrate it with the diffusion-based predictor LED on NBA. As shown in Table 3, EARefiner consistently reduces both minADE and minFDE across all four prediction horizons. At 4 s, minADE/minFDE decreases from $0.810/1.096$ to $0.796/1.079$. These improvements demonstrate that EARefiner can effectively complement diffusion-based forecasting models, further enhancing their prediction accuracy without modifying the original backbone.

Table 4: Autonomous-driving refinement results and computational cost ($K = 6$). Lower is better for every metric; best values within each dataset are shown in red.

Dataset	Method	minADE	minFDE	MR	Params (M)	Latency (ms)
Argoverse 1	HiVT	0.693	1.041	0.102	2.5	54
	HiVT + EARefiner	0.677	1.029	0.095	6.1	140
Argoverse 2	QCNet	0.636	1.241	0.154	7.6	392
	QCNet + EARefiner	0.631	1.233	0.150	11.2	478

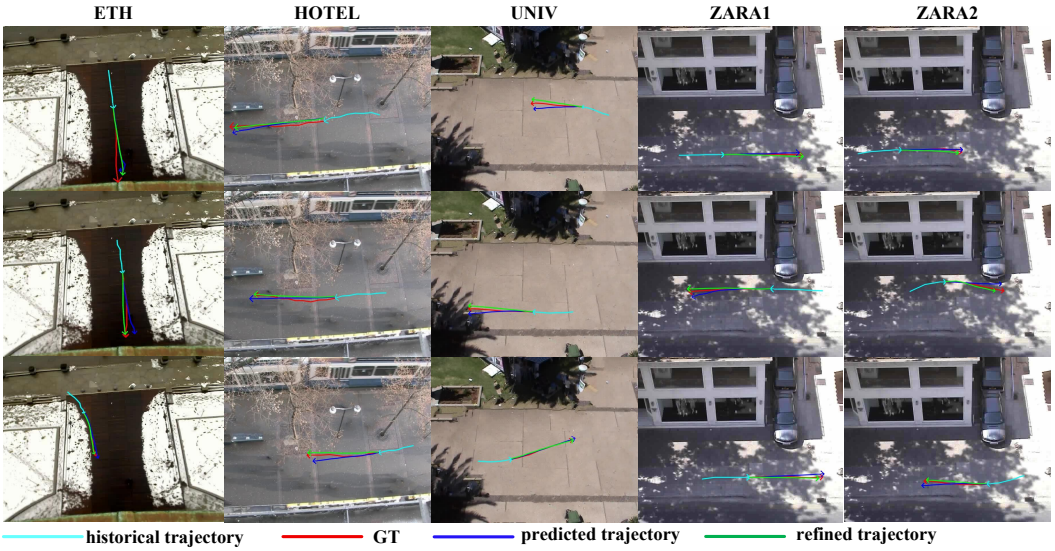

Figure 3: Prediction performance of the Gatraj model with and without EARefiner on the ETH/UCY datasets.

Table 4 presents the refinement performance of EARefiner with HiVT on Argoverse 1 and QCNet on Argoverse 2. EARefiner consistently improves minADE, minFDE, and MR on both benchmarks, demonstrating its effectiveness in challenging autonomous-driving scenarios. Specifically, HiVT achieves improved minADE/minFDE from $0.693/1.041$ to $0.677/1.029$, while QCNet improves from $0.636/1.241$ to $0.631/1.233$. These results highlight the versatility of EARefiner in enhancing diverse forecasting architectures, including highly competitive autonomous-driving predictors. Furthermore, the refinement module introduces an additional 3.6M parameters and 86 ms of inference latency in the evaluated settings, providing consistent accuracy improvements with a quantified computational cost.

Figure 3 provides qualitative visualizations of trajectory samples from the ETH/UCY dataset, illustrating the predictive performance of the Gatraj model with and without EARefiner integration. Across all five constituent subsets, EARefiner-refined trajectories consistently demonstrate superior accuracy and closer alignment with the ground truth compared to baseline predictions.

5.4 Ablation Study

Table 5: Ablation results of different sampling methods for STGCNN+EARefiner on the ETH/UCY dataset. Performance is evaluated using minADE/minFDE. Red indicates the best values (including ties).

Sample Method	ETH	HOTEL	UNIV	ZARA1	ZARA2
DDIM	0.676/1.175	0.313/0.585	0.456/0.908	0.262/0.502	0.224/0.430
DPM-SOLVER-EARefiner	0.674/1.177	0.312/0.583	0.453/0.906	0.262/0.501	0.222/0.429

Table 6: Ablation study on network components including historical trajectory encoder (HE) and coarse prediction trajectory encoder (CE) for STGCNN+EARefiner on the ETH/UCY dataset. Performance is evaluated using ADE/FDE. Red indicates the best values (including ties).

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2
w/o HE	0.703/1.205	0.331/0.612	0.469/0.907	0.283/0.510	0.238/0.441
w/o CE	0.693/1.194	0.324/0.605	0.461/0.907	0.275/0.505	0.233/0.436
Full	0.674/1.177	0.312/0.583	0.453/0.906	0.262/0.501	0.222/0.429

We conduct extensive ablation experiments to evaluate the effectiveness of the proposed sampling algorithm and the contribution of individual network components. All experiments are executed on the ETH/UCY dataset using the same training configuration. Table 5 compares DDIM with 40 NFEs against the 5-step second-order DPM-Solver with 10 NFEs. The latter retains comparable minADE/minFDE on ETH/UCY with fewer evaluations, although ETH FDE is slightly worse and ZARA1 ADE is unchanged. Table 6 shows that removing either the history encoder (HE) or coarse encoder (CE) worsens performance. For example, without HE, ETH ADE/FDE increases from $0.674/1.177$ to $0.703/1.205$, confirming the complementary value of the two contexts.

5.5 Endpoint-Mismatch Analysis

A central motivation of EARefiner is to address the training–inference endpoint mismatch in diffusion-based trajectory refinement. To evaluate endpoint alignment, we compare MLP, standard diffusion, flow matching, Diff-Refiner, and EARefiner on SDD using fixed PCHGCN predictions. Diff-Refiner and EARefiner share identical decoder architectures, model capacity, conditioning modules, and optimization settings, differing primarily in their forward processes. While Diff-Refiner employs coarse-centered diffusion with a stochastic training endpoint, EARefiner constructs a bridge terminating at the deterministic coarse prediction, aligning training and inference endpoints. As shown in Table 7, all refinement methods improve upon PCHGCN, with EARefiner achieving the best minADE/minFDE of $0.291/0.476$, compared with $0.295/0.483$ for Diff-Refiner. These controlled results support the effectiveness of endpoint-aligned refinement. The minor numerical discrepancy from Table 2 is solely due to the use of different random seeds (seed 0 for this controlled comparison vs. seed 66 for the main table).

Table 7: Comparison of refinement formulations on SDD ($K = 20$, seed 0) using fixed PCHGCN predictions. Red indicates the best values.

Refiner	minADE	minFDE
None	0.305	0.504
MLP	0.303	0.499
Standard diffusion	0.297	0.495
Flow matching	0.302	0.499
Diff-Refiner	0.295	0.483
EARefiner	0.291	0.476

6 Conclusion

EARefiner addresses the training–inference endpoint mismatch in coarse-centered trajectory refinement by adapting the tractable I^2SB bridge to ground-truth and deterministic coarse endpoints. A dedicated DPM-Solver enables efficient few-step refinement. Results on ETH/UCY, SDD, NBA, and Argoverse 1/2 demonstrate improvements across diverse backbones and application domains. Controlled comparisons support the value of endpoint alignment, while sampling experiments show that efficient solvers can maintain competitive accuracy with substantially fewer denoiser evaluations. Nevertheless, refinement gains depend on the quality of coarse predictions, and the additional refinement network incurs computational overhead. Future work will explore adaptive refinement strategies to better handle inaccurate coarse hypotheses while further reducing inference costs.

Acknowledgments and Disclosure of Funding

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant Nos. U24A20330, 62361166670, 62176124, and 62376139.

References

[1] Yizhou Huang, Gengze Jiang, Yihua Cheng, and Kezhi Wang. Foss: Modeling long range dependencies and multimodal uncertainty in trajectory prediction via fourier state space integration. *arXiv preprint arXiv:2603.01284*, 2026.

[2] Mengmeng Liu, Hao Cheng, Lin Chen, Hellward Broszio, Jiangtao Li, Runjiang Zhao, Monika Sester, and Michael Ying Yang. Laformer: Trajectory prediction for autonomous driving with lane-aware scene constraints. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2039–2049, 2024.

[3] Pubudu N Pathirana, Andrey V Savkin, and Sanjay Jha. Mobility modelling and trajectory prediction for cellular networks with mobile base stations. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 213–221, 2003.

[4] Nachiket Deo and Mohan M Trivedi. Convolutional social pooling for vehicle trajectory prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1468–1476, 2018.

[5] Zihao Sheng, Yunwen Xu, Shibei Xue, and Dewei Li. Graph-based spatial-temporal convolutional network for vehicle trajectory prediction in autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):17654–17665, 2022.

[6] Lihuan Li, Maurice Pagnucco, and Yang Song. Graph-based spatial transformer with memory replay for multi-future pedestrian trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2231–2241, 2022.

[7] Xingchen Zhang, Panagiotis Angeloudis, and Yiannis Demiris. Dual-branch spatio-temporal graph neural networks for pedestrian trajectory prediction. *Pattern Recognition*, 142:109633, 2023.

[8] Wangxing Chen, Haifeng Sang, Jinyu Wang, and Zishan Zhao. Dstigcn: Deformable spatial-temporal interaction graph convolution network for pedestrian trajectory prediction. *IEEE Transactions on Intelligent Transportation Systems*, 26(5):6923–6935, 2025.

[9] Ruochen Li, Tanqiu Qiao, Stamos Katsigiannis, Zhanxing Zhu, and Hubert PH Shum. Unified spatial–temporal edge-enhanced graph networks for pedestrian trajectory prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, 35(7):7047–7060, 2025.

[10] Hanwool Woo, Yonghoon Ji, Hitoshi Kono, Yusuke Tamura, Yasuhide Kuroda, Takashi Sugano, Yasunori Yamamoto, Atsushi Yamashita, and Hajime Asama. Lane-change detection based on vehicle-trajectory prediction. *IEEE Robotics and Automation Letters*, 2(2):1109–1116, 2017.

[11] Vinit Katariya, Mohammadreza Baharani, Nichole Morris, Omidreza Shoghli, and Hamed Tabkhi. Deeptrack: Lightweight deep learning for vehicle trajectory prediction in highways. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):18927–18936, 2022.

[12] Dongwei Xu, Xuetian Shang, Yewanze Liu, Hang Peng, and Haijian Li. Group vehicle trajectory prediction with global spatio-temporal graph. *IEEE Transactions on Intelligent Vehicles*, 8(2):1219–1229, 2022.

[13] Mengyi Zhao, Hao Tang, Pan Xie, Shuling Dai, Nicu Sebe, and Wei Wang. Bidirectional transformer gan for long-term human motion prediction. *ACM transactions on multimedia computing, communications and applications*, 19(5):1–19, 2023.

[14] Zhou Zhou, Gang Huang, Zhaoxin Su, Yongfu Li, and Wei Hua. Dynamic attention-based cvae-gan for pedestrian trajectory prediction. *IEEE Robotics and Automation Letters*, 8(2):704–711, 2022.

[15] Mihee Lee, Samuel S Sohn, Seonghyeon Moon, Sejong Yoon, Mubbasir Kapadia, and Vladimir Pavlovic. Muse-vae: Multi-scale vae for environment-aware long term trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2221–2230, 2022.

[16] Rongqing Li, Changsheng Li, Dongchun Ren, Guangyi Chen, Ye Yuan, and Guoren Wang. Bcdiff: Bidirectional consistent diffusion for instantaneous trajectory prediction. *Advances in neural information processing systems*, 36:14400–14413, 2023.

[17] Weibo Mao, Chenxin Xu, Qi Zhu, Siheng Chen, and Yanfeng Wang. Leapfrog diffusion model for stochastic trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5517–5526, 2023.

[18] Xiangzheng Zhou, Xiaobo Chen, and Jian Yang. Diff-refiner: Enhancing multi-agent trajectory prediction with a plug-and-play diffusion refiner. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10779–10785. IEEE, 2025.

[19] Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A Theodorou, Weili Nie, and Anima Anandkumar. I^2sb: Image-to-image schrödinger bridge. *arXiv preprint arXiv:2302.05872*, 2023.

[20] Renzhi Wang, Senzhang Wang, Hao Yan, and Xiang Wang. Wsip: Wave superposition inspired pooling for dynamic interactions-aware trajectory prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4685–4692, 2023.

[21] Zichen Wang, Hao Miao, Senzhang Wang, Renzhi Wang, Jianxin Wang, and Jian Zhang. C2f-tp: A coarse-to-fine denoising framework for uncertainty-aware trajectory prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pages 12810–12817, 2025.

[22] Xiaobo Chen, Huanjia Zhang, Feng Zhao, Yu Hu, Chenkai Tan, and Jian Yang. Intention-aware vehicle trajectory prediction based on spatial-temporal dynamic attention network for internet of vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):19471–19483, 2022.

[23] Hashmatullah Sadid and Constantinos Antoniou. Dynamic spatio-temporal graph neural network for surrounding-aware trajectory prediction of autonomous vehicles. *IEEE Transactions on Intelligent Vehicles*, 2024.

[24] Yanghong Liu, Xingping Dong, Yutian Lin, Mang Ye, Kaihao Zhang, and Bo Du. Condition-guided diffusion for multi-modal pedestrian trajectory prediction incorporating intention and interaction priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[25] Yu Liu, Zhijie Liu, Xiao Ren, Youfu Li, and He Kong. Intention-aware diffusion model for pedestrian trajectory prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 18469–18477, 2026.

[26] Yuxiang Fu, Qi Yan, Lele Wang, Ke Li, and Renjie Liao. Moflow: One-step flow matching for human trajectory forecasting via implicit maximum likelihood estimation based distillation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 17282–17293, 2025.

[27] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[28] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[29] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[30] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[31] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in neural information processing systems*, 35:5775–5787, 2022.

[32] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th international conference on computer vision*, pages 261–268. IEEE, 2009.

[33] Alexandre Robicquet, Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In *European conference on computer vision*, pages 549–565. Springer, 2016.

[34] Pei Xu, Jean-Bernard Hayet, and Ioannis Karamouzas. Socialvae: Human trajectory prediction using timewise latents. In *European Conference on Computer Vision*, pages 511–528. Springer, 2022.

[35] 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)*, pages 8740–8749. IEEE, 2019.

[36] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, et al. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv preprint arXiv:2301.00493*, 2023.

[37] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14424–14432, 2020.

[38] Hao Cheng, Mengmeng Liu, Lin Chen, Hellward Broszio, Monika Sester, and Michael Ying Yang. Gatraj: A graph-and attention-based multi-agent trajectory prediction model. *ISPRS Journal of Photogrammetry and Remote Sensing*, 205:163–175, 2023.

[39] Wangxing Chen, Haifeng Sang, and Zishan Zhao. Pchgcn: Physically constrained higher-order graph convolutional network for pedestrian trajectory prediction. *IEEE Internet of Things Journal*, 2025.

[40] Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8813–8823. IEEE, 2022.

[41] 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*, pages 17863–17873, 2023.

A Technical appendices and supplementary material

Appendix A.1: Theoretical Analysis and Mathematical Proofs

We use Y_{gt} and Y_{co} for ground truth and a fixed coarse hypothesis, respectively, and let $d = 2T_{\mathrm{fut}}$. The statements concern conditional endpoint distributions, not a universal performance guarantee for learned reverse dynamics.

A.1.1 Proof of Proposition 1: Endpoint mismatch and reverse-flow sensitivity

For a coarse-centered forward process with a vanishing terminal signal coefficient and positive terminal variance γ_1^2, the training endpoint conditional on Y_{co} is $P_{\mathrm{train}} = \mathcal{N}(Y_{\mathrm{co}}, \gamma_1^2\mathbf{I})$, while deterministic inference starts from $P_{\mathrm{infer}} = \delta_{Y_{\mathrm{co}}}$. Since the second measure is a point mass, every coupling between these measures has the deterministic second component Y_{co}. Hence,

$$W_2^2(P_{\mathrm{train}}, P_{\mathrm{infer}}) = \mathbb{E}_{Z\sim P_{\mathrm{train}}}\|Z - Y_{\mathrm{co}}\|_2^2 = d\gamma_1^2. \tag{24}$$

Now consider two reverse PF-ODE solutions driven by the same vector field $F(Y,t)$, starting from terminal states that differ by e_1. Assume

$$\|F(Y,t) - F(\widetilde{Y},t)\|_2 \le L(t)\|Y - \widetilde{Y}\|_2 \tag{25}$$

for an integrable nonnegative $L(t)$ along the integration interval. Grönwall's inequality yields

$$\|e_t\|_2 \le \|e_1\|_2 \exp\left(\int_t^1 L(\tau)\,\mathrm{d}\tau\right). \tag{26}$$

The inequality is an upper bound on sensitivity. It neither proves that the discrepancy grows exponentially in a given instance nor implies inevitable degradation of refined trajectories.

A.1.2 Proof of Proposition 2: Exact endpoint alignment

The tractable bridge conditioned on the endpoint pair has Gaussian marginal

$$q(Y_t \mid Y_{\mathrm{gt}}, Y_{\mathrm{co}}) = \mathcal{N}\big(\omega_t^{\mathrm{gt}} Y_{\mathrm{gt}} + \omega_t^{\mathrm{co}} Y_{\mathrm{co}}, \Sigma_t\mathbf{I}\big)\,, \tag{27}$$

where

$$\omega_t^{\mathrm{gt}} = \frac{\bar\sigma_t^2}{\sigma_t^2 + \bar\sigma_t^2}, \quad \omega_t^{\mathrm{co}} = \frac{\sigma_t^2}{\sigma_t^2 + \bar\sigma_t^2}, \quad \Sigma_t = \frac{\sigma_t^2\bar\sigma_t^2}{\sigma_t^2 + \bar\sigma_t^2}. \tag{28}$$

By definition, $\sigma_0^2 = 0$ and $\bar\sigma_1^2 = 0$. For a positive finite total variance, the coefficients satisfy

$$(\omega_0^{\mathrm{gt}}, \omega_0^{\mathrm{co}}, \Sigma_0) = (1,0,0), \qquad (\omega_1^{\mathrm{gt}}, \omega_1^{\mathrm{co}}, \Sigma_1) = (0,1,0). \tag{29}$$

Therefore, the endpoint laws are $\delta_{Y_{\mathrm{gt}}}$ and $\delta_{Y_{\mathrm{co}}}$, respectively, and the training terminal law coincides with the deterministic inference initialization. In the coarse-centered residual space,

$$Y_t - Y_{\mathrm{co}} = \omega_t^{\mathrm{gt}}(Y_{\mathrm{gt}} - Y_{\mathrm{co}}) + \sqrt{\Sigma_t}\,\epsilon, \qquad \epsilon \sim \mathcal{N}(0,\mathbf{I}), \tag{30}$$

so $Y_1 - Y_{\mathrm{co}} = 0$ almost surely. This identity concerns the prescribed bridge distribution. A learned denoiser and its finite-step numerical solver may still introduce errors, and exact endpoint alignment alone does not guarantee lower prediction error for every sample.

A.1.3 Additional Residual Concentration Analysis

Beyond exact endpoint alignment, we characterize the residual magnitude at interior bridge times. This bound describes the prescribed bridge distribution and does not guarantee per-agent forecasting improvements. Let $d = 2T_{\mathrm{fut}}$, $\Delta = Y_{\mathrm{gt}} - Y_{\mathrm{co}}$, and

$$R_t = Y_t - Y_{\mathrm{co}} = \omega_t^{\mathrm{gt}}\Delta + \sqrt{\Sigma_t}\,\epsilon, \qquad \epsilon \sim \mathcal{N}(0,\mathbf{I}_d). \tag{31}$$

The triangle inequality yields

$$\|R_t\|_2 \le \omega_t^{\mathrm{gt}}\|\Delta\|_2 + \sqrt{\Sigma_t}\|\epsilon\|_2. \tag{32}$$

Since $\|\epsilon\|_2^2 \sim \chi_d^2$, the Laurent–Massart tail inequality gives, for any $x > 0$,

$$\Pr\Big[\|\epsilon\|_2^2 \geq d + 2\sqrt{dx} + 2x\Big] \leq e^{-x}. \tag{33}$$

Define $\rho(x) = \sqrt{d + 2\sqrt{dx} + 2x}$. Setting $x = \log(1/\delta)$, for $0 < \delta < 1$, gives

$$\Pr\Big[\|R_t\|_2 \leq \omega_t^{\mathrm{gt}} \|\Delta\|_2 + \sqrt{\Sigma_t}\, \rho(\log(1/\delta))\Big] \geq 1 - \delta. \tag{34}$$

For a finite grid $\{t_j\}_{j=1}^M$, replacing δ by δ/M and applying the union bound gives the simultaneous bound, with probability at least $1 - \delta$,

$$\|R_{t_j}\|_2 \leq \omega_{t_j}^{\mathrm{gt}} \|\Delta\|_2 + \sqrt{\Sigma_{t_j}}\, \rho(\log(M/\delta)) \tag{35}$$

for all $j = 1, \ldots, M$. Both ω_t^{gt} and Σ_t vanish at $t = 1$, consistently with the exact terminal residual $R_1 = 0$.

Appendix A.2: Derivation of DPM-Solver-EARefiner Formulas

In this section, we detail the derivation of the first- and second-order solver updates for EARefiner presented in the Method section. The derivation uses variation of constants and a log-SNR transformation.

A.2.1 Equivalent Probability Flow ODE

The residual variable transformation of EARefiner is given by:

$$\tilde{Y}_t = Y_t - Y_{\mathrm{co}} = \omega_t^{\mathrm{gt}} \tilde{Y}_0 + \sqrt{\Sigma_t}\epsilon. \tag{36}$$

This Gaussian marginal distribution admits a compatible linear forward SDE and associated PF-ODE, formulated as:

$$\frac{\mathrm{d}\tilde{Y}_t}{\mathrm{d}t} = f(t)\tilde{Y}_t + \frac{g^2(t)}{2\sqrt{\Sigma_t}}\epsilon_\theta(Y_t, Y_{\mathrm{co}}, X, t), \tag{37}$$

where the linear drift coefficient is defined as:

$$f(t) = \frac{\mathrm{d}\ln \omega_t^{\mathrm{gt}}}{\mathrm{d}t}. \tag{38}$$

Here $g^2(t) = \frac{\mathrm{d}\Sigma_t}{\mathrm{d}t} - 2f(t)\Sigma_t$ is the diffusion variance rate of the compatible SDE; the corresponding score is represented through the noise predictor. Multiplying both sides by the integrating factor $\exp(-\int f(t)\mathrm{d}t) = \frac{1}{\omega_t^{\mathrm{gt}}}$ and integrating over the time interval $[t, s]$ yields the analytical equation:

$$\frac{\tilde{Y}_s}{\omega_s^{\mathrm{gt}}} - \frac{\tilde{Y}_t}{\omega_t^{\mathrm{gt}}} = \int_t^s \frac{g^2(r)}{2\omega_r^{\mathrm{gt}}\sqrt{\Sigma_r}}\epsilon_\theta(Y_r, Y_{\mathrm{co}}, X, r)\mathrm{d}r. \tag{39}$$

A.2.2 Log-SNR Substitution and Exact Integration Equation

Utilizing the logarithmic signal-to-noise ratio λ_r and the differential identity of diffusion models:

$$\lambda_r = \ln\left(\frac{\omega_r^{\mathrm{gt}}}{\sqrt{\Sigma_r}}\right), \quad \frac{g^2(r)}{2\omega_r^{\mathrm{gt}}\sqrt{\Sigma_r}} = -e^{-\lambda_r}\frac{\mathrm{d}\lambda_r}{\mathrm{d}r}, \tag{40}$$

we convert the time integration perfectly into an integration over λ. Defining the step size as $h = \lambda_s - \lambda_t$ and rearranging the terms yields the exact closed-form integration solution:

$$\tilde{Y}_s = \frac{\omega_s^{\mathrm{gt}}}{\omega_t^{\mathrm{gt}}}\tilde{Y}_t - \omega_s^{\mathrm{gt}} \int_{\lambda_t}^{\lambda_s} e^{-\lambda}\epsilon_\theta(Y_\lambda, Y_{\mathrm{co}}, X, \lambda)\mathrm{d}\lambda. \tag{41}$$

A.2.3 First-order Truncation: EARefiner-DPM-1

Within a small time jump window, we apply a zero-order Taylor expansion to the denoising network at the starting point λ_t. Assuming $\epsilon_\theta(Y_\lambda, Y_{\text{co}}, X, \lambda) \approx \hat{\epsilon}_t$ remains constant, we extract it outside the integral to obtain:

$$\tilde{Y}_s = \frac{\omega_s^{\text{gt}}}{\omega_t^{\text{gt}}}\tilde{Y}_t - \omega_s^{\text{gt}}\hat{\epsilon}_t \int_{\lambda_t}^{\lambda_s} e^{-\lambda} \mathrm{d}\lambda. \tag{42}$$

Evaluating this exponential integral gives:

$$\int_{\lambda_t}^{\lambda_s} e^{-\lambda} \mathrm{d}\lambda = e^{-\lambda_t} - e^{-\lambda_s} = e^{-\lambda_s}(e^h - 1). \tag{43}$$

Using the established relation $e^{-\lambda_s} = \frac{\sqrt{\Sigma_s}}{\omega_s^{\text{gt}}}$, the prefix term ω_s^{gt} is exactly canceled. Mapping Y_s back to the original space yields the first-order formula:

$$Y_s = \frac{\omega_s^{\text{gt}}}{\omega_t^{\text{gt}}}(Y_t - Y_{\text{co}}) + Y_{\text{co}} - \sqrt{\Sigma_s}(e^h - 1)\hat{\epsilon}_t. \tag{44}$$

A.2.4 Second-order Approximation: EARefiner-DPM-2

To achieve higher-order accuracy, we perform a first-order Taylor expansion at λ_t, expressed as:

$$\epsilon_\theta(Y_\lambda, Y_{\text{co}}, X, \lambda) \approx \hat{\epsilon}_t + \frac{\lambda - \lambda_t}{h} \cdot h\epsilon_t'. \tag{45}$$

Substituting this approximation into the exact integration equation and integrating term by term results in:

$$\tilde{Y}_s = \frac{\omega_s^{\text{gt}}}{\omega_t^{\text{gt}}}\tilde{Y}_t \quad -\omega_s^{\text{gt}}\left[\hat{\epsilon}_t \int_{\lambda_t}^{\lambda_s} e^{-\lambda}\,\mathrm{d}\lambda \qquad + h\epsilon_t' \int_{\lambda_t}^{\lambda_s} e^{-\lambda}\frac{\lambda - \lambda_t}{h}\,\mathrm{d}\lambda\right]. \tag{46}$$

Through integration by parts, the analytical solution for the second integral is calculated as:

$$e^{-\lambda_s}\left(\frac{e^h - 1}{h} - 1\right). \tag{47}$$

To avoid the highly expensive computation of the neural network Jacobian matrix required to obtain ϵ_t', we introduce the midpoint $\lambda_m = \frac{\lambda_t + \lambda_s}{2}$. We estimate the derivative using the midpoint difference:

$$h\epsilon_t' \approx 2(\hat{\epsilon}_m - \hat{\epsilon}_t), \tag{48}$$

where the midpoint state Y_m is derived from the aforementioned first-order formula with a step size of $h/2$, given by:

$$Y_m = \frac{\omega_m^{\text{gt}}}{\omega_t^{\text{gt}}}(Y_t - Y_{\text{co}}) + Y_{\text{co}} - \sqrt{\Sigma_m}(e^{h/2} - 1)\hat{\epsilon}_t. \tag{49}$$

Substituting the derivative approximation back and mapping the result to the original space yields the second-order derivative-corrected formula used by EARefiner:

$$Y_s = \frac{\omega_s^{\text{gt}}}{\omega_t^{\text{gt}}}(Y_t - Y_{\text{co}}) + Y_{\text{co}} \quad -\sqrt{\Sigma_s}(e^h - 1)\hat{\epsilon}_t - 2\sqrt{\Sigma_s}\left(\frac{e^h - 1}{h} - 1\right)(\hat{\epsilon}_m - \hat{\epsilon}_t). \tag{50}$$

A.2.5 Terminal coefficient evaluation

At $t = 1$, the bridge state is well defined ($Y_1 = Y_{\text{co}}$), but $\omega_1^{\text{gt}} = \Sigma_1 = 0$ makes the solver coefficient ratios and log-SNR undefined. For numerical stability, coefficients are evaluated at a baseline-dependent interior cutoff $t_{\max} < 1$, while the state is initialized from the coarse trajectory. This regularizes the coefficient evaluation but approximates the terminal step rather than integrating exactly from $t = 1$.

Appendix A.3: Additional Experimental Analyses

A.3.1 Architecture and Training Configuration

Table 8: EARefiner architecture and optimization settings used in the experiments.

Component	Configuration
Core DiT	6 blocks; hidden dimension 128; 8 attention heads
Each block	Temporal and spatial/mode-agent attention; conditional cross-attention; MLP; AdaLN-Zero
DiT MLP	Expansion ratio 4; hidden dimension 512
Time embedding	MLP 1–128–128 with SiLU
Input/output	Noisy/coarse projections 2–128; output 128–64–2 with GELU
Condition fusion	Cross-attention, dimension 128, 4 heads
Coarse encoder	MLP 2–128–128; temporal and mode/agent attention, 4 heads
Historical temporal encoder	MLP to 128; three-layer, eight-head temporal attention
Historical spatial/fusion	GraphConv 128–128, 2 propagation steps; one-layer GRU, hidden dimension 128
Bridge schedule	300 retained steps from 302; $\beta \in [10^{-3}, 3 \times 10^{-2}]$
Training	Adam; batch size 128; learning rate 5×10^{-4}
Training epochs	1,000 on ETH/UCY, SDD, and NBA; 100 on Argoverse 1/2
Terminal clipping	To avoid numerical singularities at $t = 1$, the solver evaluates its coefficients at a nearby interior time (e.g., $t_{\max} = 0.95$).

Table 8 summarizes the architectural and optimization settings of EARefiner. The six-block DiT combines temporal and spatial/mode-agent attention with conditional cross-attention, allowing the denoiser to incorporate historical motion and coarse trajectory information during refinement. The historical and coarse encoders provide complementary conditioning features, while the endpoint-aligned bridge defines the refinement process. For the PCHGCN configuration, the backbone contains 15,260 parameters, the refiner contains 3,666,464 parameters, and the full pipeline contains 3,681,724 parameters. In the controlled comparison of Table 7, Diff-Refiner and EARefiner share the decoder architecture, model capacity, conditioning inputs, and optimization settings, enabling a focused evaluation of their different forward processes.

A.3.2 Statistical Robustness Across Training Seeds

Table 9: EARefiner robustness across training seeds on SDD with a fixed PCHGCN checkpoint.

Method	Seed	minADE	minFDE
PCHGCN	fixed	0.305	0.504
PCHGCN + EARefiner	0	0.291	0.476
PCHGCN + EARefiner	66	0.294	0.482
PCHGCN + EARefiner	666	0.291	0.478
PCHGCN + EARefiner	mean$\pm$std	0.292 ± 0.002	0.479 ± 0.003

Table 9 evaluates the robustness of EARefiner across three independent training seeds while keeping the PCHGCN checkpoint fixed. EARefiner improves both minADE and minFDE over the backbone in all three runs, achieving average values of 0.292 ± 0.002 and 0.479 ± 0.003, respectively. The small standard deviations indicate consistent refinement performance under different random initializations. These results demonstrate that the improvements are reproducible across the evaluated seeds rather than depending on a particular training run. The main SDD evaluation in Table 2 uses seed 66, while the controlled comparison in Table 7 uses seed 0.

A.3.3 Bootstrap Confidence Intervals and Agent-Level Refinement

Table 10: SDD bootstrap confidence intervals and agent-level refinement behavior for seed 0. Improvements are PCHGCN error minus EARefiner error; confidence intervals are for the aggregate improvement.

Metric	PCHGCN	EARefiner	Improvement	95% CI	Improved agents	Worsened agents
minADE	0.305	0.291	0.014	[0.008, 0.019]	66.80%	33.20%
minFDE	0.504	0.476	0.028	[0.014, 0.040]	68.02%	31.98%

Table 10 further examines the reliability of EARefiner through bootstrap confidence intervals and agent-level statistics. EARefiner achieves aggregate improvements of 0.014 in minADE and 0.028 in minFDE, with corresponding 95% confidence intervals of $[0.008, 0.019]$ and $[0.014, 0.040]$. Both intervals remain above zero, supporting the reliability of the observed aggregate improvements under the adopted scene-level resampling protocol. At the agent level, EARefiner improves ADE for 66.80% of agents and FDE for 68.02%, indicating that the accuracy gains extend to a majority of evaluated trajectories. Although individual predictions may occasionally deteriorate, the overall results demonstrate a consistent positive refinement effect.

A.3.4 Dependence on Coarse-Prediction Quality

Table 11: Sensitivity to PCHGCN coarse-prediction quality on SDD (seed 0). Improvement denotes coarse error minus refined error.

Quartile	Coarse minADE/minFDE	EARefiner minADE/minFDE	Improvement
Q1: best 25%	0.027/0.048	0.017/0.030	0.010/0.018
Q2: 25–50%	0.146/0.269	0.128/0.226	0.018/0.043
Q3: 50–75%	0.275/0.461	0.259/0.425	0.016/0.036
Q4: worst 25%	0.780/1.250	0.774/1.251	0.006/-0.001

Table 11 investigates how the quality of initial coarse predictions influences refinement performance. EARefiner improves both minADE and minFDE across the first three quartiles, demonstrating its effectiveness for coarse hypotheses with different initial error levels. The largest minFDE reduction occurs in Q2, decreasing from 0.269 to 0.226, while Q3 also exhibits clear improvements in both metrics. For Q4, EARefiner still reduces minADE from 0.780 to 0.774, although minFDE increases slightly from 1.250 to 1.251. These observations indicate that endpoint-aligned refinement is particularly effective when coarse hypotheses preserve useful future motion information, while substantially inaccurate initial predictions present a more challenging refinement setting.

A.3.5 Reverse-Process Diagnostics

Table 12: Coarse-centered RMS residuals during five-step reverse integration on SDD (seed 0).

Step	Mean	95th pct.	Maximum	NaN/Inf rate
1	0.014	0.027	0.050	0.000
2	0.107	0.235	0.317	0.000
3	0.243	0.535	0.733	0.000
4	0.271	0.598	0.822	0.000
5	0.274	0.604	0.830	0.000

Table 12 examines the evolution of coarse-centered residuals during the five-step reverse refinement process. Starting from the deterministic coarse trajectory with zero initial residual, EARefiner progressively adjusts the predicted trajectories, with the mean RMS residual increasing from 0.014 after the first step to 0.274 after the fifth step. The smaller changes during the final two steps indicate that the magnitude of trajectory corrections becomes more stable toward the end of sampling. Moreover, the NaN/Inf rate remains zero throughout the reverse process, providing empirical evidence

of numerical stability under the evaluated configuration. These results illustrate how EARefiner progressively transforms coarse hypotheses into refined predictions through a short reverse integration process.

A.3.6 Full NFE, Runtime, and FLOP Comparison

Table 13: Matched-NFE refinement accuracy and efficiency on SDD ($K = 20$, seed 0). NFE counts denoiser evaluations, including midpoint evaluations for Bridge-DPM-2. Red denotes the best minADE/minFDE within each NFE budget.

NFE	Sampler	minADE/minFDE	Latency (ms/scene)	GFLOPs/scene
5	Bridge-DDIM	0.29808/0.48397	50.661	10.530
5	Bridge-DPM-1	0.29249/0.47776	98.194	10.530
5	Bridge-DPM-2	0.29101/0.47629	99.882	10.530
10	Bridge-DDIM	0.29343/0.47912	95.494	20.461
10	Bridge-DPM-1	0.29048/0.47420	172.088	20.461
10	Bridge-DPM-2	0.29014/0.47391	199.396	20.461
20	Bridge-DDIM	0.29147/0.47591	177.934	40.324
20	Bridge-DPM-1	0.28998/0.47355	345.365	40.324
20	Bridge-DPM-2	0.28983/0.47322	396.044	40.324
40	Bridge-DDIM	0.29040/0.47423	361.209	84.021
40	Bridge-DPM-1	0.28985/0.47329	657.251	84.021
40	Bridge-DPM-2	0.28976/0.47304	740.882	84.021

Table 13 compares Bridge-DDIM, Bridge-DPM-1, and Bridge-DPM-2 under different denoiser evaluation budgets. Bridge-DPM-2 consistently achieves the lowest minADE and minFDE at every evaluated NFE, demonstrating its accuracy advantage under matched evaluation budgets. Notably, with only 5 NFEs, Bridge-DPM-2 achieves $0.29101/0.47629$, outperforming 20-NFE Bridge-DDIM ($0.29147/0.47591$) in minADE, while achieving a slightly higher minFDE. At 10 NFEs, Bridge-DPM-2 reaches $0.29014/0.47391$, surpassing 40-NFE Bridge-DDIM ($0.29040/0.47423$) in both metrics. More importantly, this configuration reduces inference latency from 361.209 to 199.396 ms/scene and computational cost from 84.021 to 20.461 GFLOPs/scene. Although DPM-based updates incur higher wall-clock overhead at identical NFEs, their improved accuracy per denoiser evaluation enables competitive performance with substantially fewer evaluations. These results demonstrate the effectiveness of the proposed solver for accurate few-step trajectory refinement.

A.3.7 Prediction Diversity Analysis

For K trajectory hypotheses $\hat{\mathbf{Y}}^{(k)}$, average pairwise displacement (APD) and final pairwise displacement (FPD) are defined as

$$\mathrm{APD} = \frac{2}{K(K-1)} \sum_{i<j} \left[\frac{1}{T} \sum_{t=1}^{T} \|\hat{\mathbf{y}}_t^{(i)} - \hat{\mathbf{y}}_t^{(j)}\|_2 \right], \tag{51}$$

$$\mathrm{FPD} = \frac{2}{K(K-1)} \sum_{i<j} \|\hat{\mathbf{y}}_T^{(i)} - \hat{\mathbf{y}}_T^{(j)}\|_2. \tag{52}$$

Table 14: Forecast accuracy and trajectory diversity on SDD ($K = 20$, seed 0). Lower is better for minADE/minFDE; APD and FPD describe hypothesis spread.

Method	minADE	minFDE	APD	FPD
PCHGCN	0.305	0.504	1.71817	3.58892
PCHGCN + EARefiner	0.291	0.476	1.45730	2.94827

Table 14 evaluates whether the accuracy improvements of EARefiner are accompanied by changes in trajectory diversity. Compared with PCHGCN, EARefiner reduces minADE/minFDE

from $0.305/0.504$ to $0.291/0.476$, while APD and FPD decrease from $1.71817/3.58892$ to $1.45730/2.94827$. The refined predictions retain approximately 84.82% of the original APD and 82.15% of the original FPD, indicating that the refinement process preserves nonzero hypothesis diversity while producing a more concentrated trajectory distribution. These findings suggest that EARefiner enhances prediction accuracy without collapsing all hypotheses into a single trajectory. Since larger pairwise distances do not necessarily indicate more plausible predictions, the diversity measurements are interpreted alongside minADE and minFDE.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes,the main claims presented in the abstract and introduction accurately reflect the core contributions and scope of the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discussed the limitations of EARefiner in the conclusion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: For each theorem and corresponding conclusive formula presented in the paper, we have provided detailed proofs in the Technical Appendices and Supplementary Materials section.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Yes,the paper fully discloses all necessary information to reproduce the main experimentalresults, supporting the main claims and conclusions.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: All datasets are publicly available. And we will release the code if the paper is accepted.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Yes, the paper provides all necessary training and test details, including data splits, hyperparameters, and optimizer type, to fully understand the results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We do not report error bars.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provided the type of compute workers and memory in the paper, but did not provide the time of execution.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, the research in the paper fully conforms to the NeurIPS Code of Ethics in every respect.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We elaborated on the positive impact of this study on autonomous driving in the conclusion section.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This paper has no relevant risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes, we clearly indicate the baseline methods and training/testing data used in the paper. Their licenses permit use within academic scope.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code and model will be released if the article is accepted.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing experiments and research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing experiments and research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core of the research paper does not involve LLMs, and all ideas and core designs do not involve LLMs. The use of LLMs is only for language polishing of the article.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.