WHERE TO GO AND HOW TO MOVE: GOAL–MOTION LATENT FLOW FOR MULTIMODAL TRAJECTORY PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Trajectory prediction plays a critical role in autonomous driving and intelligent surveillance. Recent generative models have shown promising results in capturing the uncertainty of future trajectories. However, many of these models do not explicitly distinguish long-term motion goals from local motion details in their trajectory representations, potentially affecting the accuracy of both local paths and endpoints. To address this challenge, we propose Goal–Motion Latent Flow (GMFlow), a novel dual-latent flow model comprising a goal–motion representation module and a conditional flow generation module. The representation module decomposes deviations from a constant-velocity reference trajectory into two complementary latents: a goal latent encoding global goals and a motion latent capturing fine-grained local variations. Conditioned on observed history and learnable mode queries, the conditional flow generation module jointly generates both latents through flow matching, producing multiple future trajectories in a single flow step. A goal-preserving decoder allows the motion latent to adjust local path shapes without changing the positions specified by the goal latent. Together, these designs enable improved prediction accuracy with efficient generation of multiple candidate trajectories. Experiments on NBA, SDD, and ETH–UCY demonstrate state-of-the-art performance, with particularly strong results in long-term endpoint prediction.

1 INTRODUCTION

Trajectory prediction aims to forecast future position sequences based on observed trajectories. This task plays a critical role in autonomous driving (Ettinger et al., 2021), intelligent surveillance (Liang et al., 2021), and human–robot interaction (Zhang et al., 2020). Predicting human motion involves inherent uncertainty arising from individual intentions and social interactions (Kothari et al., 2021). As illustrated in Figure 1, a given observed trajectory may correspond to different future goals or different motion paths toward the same goal. To capture these possibilities, stochastic trajectory prediction methods model the distribution of future trajectories conditioned on past observations and related factors to generate multiple candidate trajectories (Gupta et al., 2018; Salzmann et al., 2020).

To capture the uncertainty of future motion, researchers have developed a range of generative trajectory prediction methods. Early approaches employ generative adversarial networks (Gupta et al., 2018; Dendorfer et al., 2021) and conditional variational autoencoders (Salzmann et al., 2020; Yuan et al., 2021) to model multimodal future trajectories. More recently, diffusion models generate trajectories through iterative denoising (Gu et al., 2022), while flow-matching approaches (Lipman et al., 2023) learn continuous transformations from a source distribution to a conditional distribution of future trajecto-

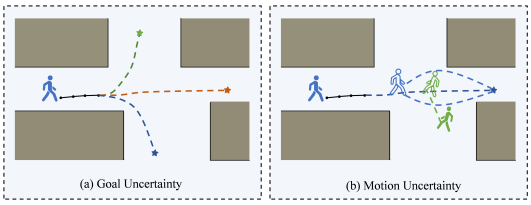

Figure 1: Goal uncertainty: different destinations. Motion uncertainty: different paths toward the same destination.

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

ries (Fu et al., 2025). However, fine-grained motion patterns and long-term goals are not explicitly distinguished.

Alongside these advances in generation, researchers have also explored rich representations of motion. Low-rank trajectory descriptors provide compact spaces for modeling future trajectories (Bae et al., 2023; 2024). Besides, some goal-guided methods first predict goals or key positions and then construct the corresponding paths (Zhao et al., 2021; Bae & Jeon, 2023; Zhang et al., 2024), while others use goal and local motion information to conditionally guide the generation of trajectories (Liu et al., 2026). However, low-rank representations compress trajectories into compact latent spaces but do not explicitly preserve the semantic distinction between long-term goals and local motion variations. Goal-guided approaches connect goals and paths through hierarchical prediction or conditional guidance. We instead seek an explicit goal–motion decomposition in which local path variations leave the specified goal positions unchanged, while both components are generated jointly to capture their dependencies.

To this end, we propose Goal–Motion Latent Flow (GMFlow), a dual-latent flow framework comprising a goal–motion representation module and a conditional flow generation module. The representation module takes a constant-velocity extrapolation of the observed trajectory as a reference and decomposes deviations from this reference into two complementary latent variables. Specifically, the goal latent encodes offsets of intermediate and final goal positions, while the motion latent captures local trajectory details between these positions. To preserve this distinction during reconstruction, a goal-preserving decoder allows the motion latent to adjust local path shapes without changing the positions specified by the goal latent. Building on this representation, the conditional flow generation module uses observed history and learnable mode queries to jointly generate both latents through flow matching. At inference time, GMFlow produces multiple candidate trajectories in a single flow step, without requiring an additional model distillation stage.

We conduct experiments on three trajectory prediction benchmarks: NBA, SDD, and ETH-UCY. The quantitative results show GMFlow achieves state-of-the-art performance. Specifically, compared with the one-step distilled MoFlow model (Fu et al., 2025), GMFlow achieves improvements of 4.2% and 7.0% in 4-second minADE_{20} and minFDE_{20} on NBA, while reducing the average prediction time from ~ 0.70 ms to ~ 0.36 ms.

The contributions of our work are as follows:

- We design a goal–motion representation module that separately encodes goal position offsets relative to a reference trajectory and local motion details between goals. A goal-preserving decoder enables local path adjustments while preserving the positions determined by the goal latent.
- We propose GMFlow, a multimodal trajectory prediction framework based on goal and motion latents. Conditioned on observed history and learnable mode queries, it jointly generates both latents through flow matching, enabling parallel multimodal prediction in a single flow step.
- We evaluate GMFlow on NBA, SDD, and ETH-UCY, demonstrating state-of-the-art prediction performance with particularly strong results in long-term endpoint prediction.

2 Related Work

2.1 Trajectory Prediction

Early approaches predict pedestrian trajectories using predefined motion rules that account for destinations, surrounding obstacles, and interactions with other pedestrians (Helbing & Molnar, 1995; Pellegrini et al., 2009; Luber et al., 2010; Yamaguchi et al., 2011; Karamouzas et al., 2014). With the development of deep learning, recurrent neural networks have been used to learn motion patterns from observed trajectories and capture pedestrian interactions through social pooling and hidden-state refinement (Alahi et al., 2016; Zhang et al., 2019). Subsequent methods employ graph networks and attention mechanisms to further model these interactions. STGAT and Social-STGCNN represent pedestrians and their relationships as graphs to predict future trajectories (Huang et al., 2019; Mohamed et al., 2020), while STAR and AgentFormer use attention to capture dependencies across time steps and agents (Yu et al., 2020; Yuan et al., 2021). To account for multiple plausible futures,

generative methods learn trajectory distributions conditioned on observed history and contextual information. Social GAN and SoPhie use adversarial learning to generate diverse trajectories (Gupta et al., 2018; Sadeghian et al., 2019), while Trajectron++, AgentFormer, and GroupNet adopt conditional variational formulations (Salzmann et al., 2020; Yuan et al., 2021; Xu et al., 2022a). More recently, MID introduces diffusion-based trajectory prediction through iterative denoising (Gu et al., 2022). LED improves sampling efficiency with a learnable initializer bypassing early denoising steps (Mao et al., 2023). MoFlow applies conditional flow matching to multi-candidate trajectory generation and enables one-step prediction through distillation (Fu et al., 2025).

2.2 Trajectory Representations

Trajectory representations determine how future motion is parameterized and reconstructed. Some methods predict trajectory parameters and reconstruct the full trajectory from them, rather than directly predicting positions at individual time steps. For example, probabilistic Bézier curves and their Gaussian-process extensions represent trajectories through stochastic control points (Hug et al., 2020; 2022). Su et al. (2021) use polynomials to represent future states and their uncertainty, predicting polynomial coefficients to obtain temporally continuous forecasts. In these methods, positions at different time steps are linked through the same curve parameters. Beyond parametric curves, spectral and low-rank decompositions have also been explored for trajectory representation (Wong et al., 2022; Bae et al., 2023). SingularTrajectory constructs a shared motion space and combines it with a diffusion model for trajectory prediction (Bae et al., 2024). Resonance represents future trajectories as the superposition of vibration components that account for individual motion and social influences (Wong et al., 2025). Compared with these representations, we explicitly separate trajectory residuals into goal offsets and local motion coefficients. The motion basis is constrained to be zero at goal timestamps, allowing local path variations without altering the positions specified by the goal latent.

2.3 Goal and Intention Modeling

Goal-guided methods predict future destinations and use them to guide trajectory generation. PECNet estimates endpoints and generates trajectories conditioned on them (Mangalam et al., 2020). MemoNet retrieves relevant motion instances from memory to predict destinations and then generates the corresponding trajectories (Xu et al., 2022b). Y-Net models multimodal long-term goals together with goal-conditioned waypoints and paths, allowing multiple trajectories to reach the same goal (Mangalam et al., 2021). Beyond final destinations, intermediate positions and local motion cues provide additional guidance for trajectory generation. Graph-TERN predicts control points and refines the corresponding paths (Bae & Jeon, 2023), while G2LTraj predicts global key positions and recursively generates the intermediate positions between them (Zhang et al., 2024). IAD combines residual polar representations of local motion with predicted endpoints to guide trajectory denoising (Liu et al., 2026). However, these approaches either rely on fixed-anchor-based hierarchical prediction, sacrificing adaptability to subtle variations, or use conditional guidance to connect goals and paths, making it difficult to balance goal modeling and local motion variability within a joint generation framework. These limitations may hinder their ability to improve both local trajectory and endpoint prediction accuracy. In contrast, our model generates goal offsets and local motion coefficients together within a single conditional flow model. The two components are updated jointly, allowing goals and local motion to inform each other during generation.

3 Method

Figure 2 provides an overview of GMFlow. The framework consists of a goal–motion representation module and a conditional flow generation module. The representation module decomposes future trajectory residuals into goal offsets and local motion coefficients. Conditioned on observed history and factorized goal and motion queries, the generation module jointly predicts both latent components in a single flow step. A goal-preserving decoder then reconstructs future trajectories, allowing local path variations while preserving the positions specified by the goal latent.

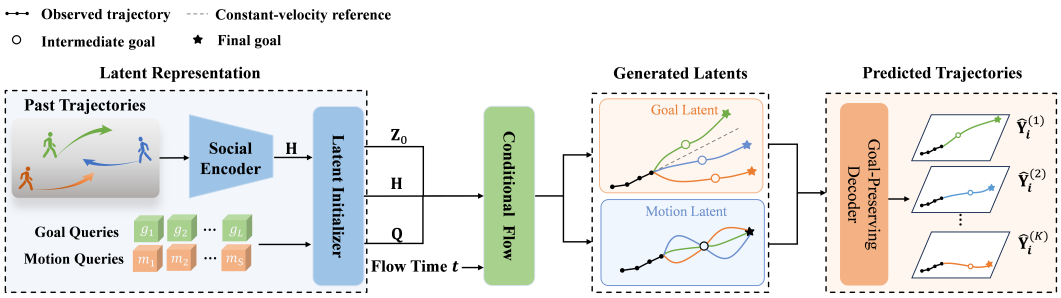

Figure 2: Overview of GMFlow. Observed trajectories and factorized goal and motion queries provide the context and initial states for conditional flow generation. The generated goal and motion latents are decoded into future trajectories by a goal-preserving decoder.

3.1 Problem Formulation

Consider a scene containing N agents. We denote their observed trajectories and ground-truth future trajectories by $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \in \mathbb{R}^{N \times T_{\text{obs}} \times 2}$ and $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N] \in \mathbb{R}^{N \times T_{\text{pred}} \times 2}$, where T_{obs} and T_{pred} denote the observation and prediction timestamps, respectively. Given $\mathbf{X}$, multimodal trajectory prediction aims to generate K possible future trajectories, $\hat{\mathcal{Y}} = \{\hat{\mathbf{Y}}^{(k)}\}_{k=1}^{K}$, with $\hat{\mathbf{Y}}^{(k)} \in \mathbb{R}^{N \times T_{\text{pred}} \times 2}$. Rather than directly modeling multimodality at the trajectory level, we decompose future motion into goal positions and local motion details, and jointly model their uncertainties through a structured latent representation.

3.2 Goal–Motion Representation

This section introduces the goal–motion representation module. We first construct a constant-velocity reference trajectory and then decompose the deviation from this reference into a goal component and a motion component.

For agent i, we construct a constant-velocity reference trajectory by linearly extrapolating the motion trend between its last two observed positions: $\mathbf{b}_i^\tau = \mathbf{x}_i^{T_{\text{obs}}} + \tau\big(\mathbf{x}_i^{T_{\text{obs}}} - \mathbf{x}_i^{T_{\text{obs}}-1}\big)$, where $\mathbf{B}_i = [\mathbf{b}_i^1, \ldots, \mathbf{b}_i^{T_{\text{pred}}}]^\top$ denotes the reference trajectory, and $\mathbf{R}_i = \mathbf{y}_i - \mathbf{B}_i$ denotes the residual between the ground-truth future and the reference trajectory.

Goal latent. The goal latent describes residual offsets at selected intermediate and final goal positions. We uniformly select M goal timestamps over the prediction horizon, with the last goal corresponding to the final prediction timestamp: $\tau_m = (m+1)T_{\text{pred}}/M$ and $\mathbf{G}_i[m,:] = \mathbf{R}_i[\tau_m,:]$ for $m = 0, \ldots, M-1$. The goal matrix $\mathbf{G}_i \in \mathbb{R}^{M \times 2}$ contains the residual offsets at all selected goal timestamps. We define the goal latent as $\mathbf{z}_{g,i} = \text{vec}(\mathbf{G}_i) \in \mathbb{R}^{2M}$. To extend the discrete goal offsets over the full prediction horizon, we use a fixed piecewise-linear interpolation matrix $\mathbf{A} \in \mathbb{R}^{T_{\text{pred}} \times M}$. The interpolated goal offsets satisfy $(\mathbf{A}\mathbf{G}_i)[\tau_m,:] = \mathbf{G}_i[m,:]$ and therefore pass through each selected goal offset. The complete interpolation rule is provided in Appendix A.1.

Motion latent. Although the interpolated goal path captures the overall motion trend, it cannot describe local path variations between adjacent goals. Therefore, we construct a goal-preserving motion basis that captures local variations while keeping all goal positions unchanged. Specifically, we derive the basis from an ordered orthonormal DCT-II basis by restricting it to non-goal timestamps, applying QR orthonormalization, and setting its values at goal timestamps to zero. The resulting basis $\mathbf{D} \in \mathbb{R}^{T_{\text{pred}} \times q}$ satisfies $\mathbf{D}^\top\mathbf{D} = \mathbf{I}_q$ and $\mathbf{D}[\tau_m,:] = \mathbf{0}$. The first property ensures orthonormal basis vectors, while the second guarantees that the motion component does not affect goal positions. The complete construction is provided in Appendix A.2.

We extract the local residual not explained by the interpolated goal component as $\mathbf{E}_i = \mathbf{R}_i - \mathbf{A}\mathbf{G}_i$, and project it onto the goal-preserving motion subspace: $\mathbf{U}_i = \mathbf{D}^\top(\mathbf{R}_i - \mathbf{A}\mathbf{G}_i) \in \mathbb{R}^{q \times 2}$. The motion latent is defined as $\mathbf{z}_{m,i} = \text{vec}(\mathbf{U}_i) \in \mathbb{R}^{2q}$. This projection represents local path variations in

a compact latent space while preserving the goal positions. When $q < T_{\text{pred}} - M$, $\mathbf{DU}_i$ provides a low-dimensional approximation of local motion details.

Goal-preserving decoding. Given the predicted goal matrix $\hat{\mathbf{G}}_i$ and motion coefficients $\hat{\mathbf{U}}_i$, we reconstruct the future trajectory as $\hat{\mathbf{y}}_i = \mathbf{B}_i + \mathbf{A}\hat{\mathbf{G}}_i + \mathbf{D}\hat{\mathbf{U}}_i$. Since $\mathbf{D}[\tau_m, :] = \mathbf{0}$, the motion component vanishes at all goal timestamps: $\hat{\mathbf{y}}_i[\tau_m, :] = \mathbf{B}_i[\tau_m, :] + \hat{\mathbf{G}}_i[m, :]$. Therefore, the goal latent determines the intermediate and final goal positions, while the motion latent only adjusts local path variations between them. This goal-preserving property enables explicit separation between where to go and how to move. In our implementation, we set $M = 2$ and $q = 8$, corresponding to one intermediate goal, one final goal, and a 16-dimensional motion latent.

3.3 Conditional Flow Generation

The goal and motion latents defined in Section 3.2 are obtained from the ground-truth future and are therefore unavailable at inference time. The conditional flow generation module learns to generate these latents from observed history and produces multiple goal–motion combinations in parallel.

3.3.1 Latent Initialization

As illustrated in Figure 3, a history encoder first extracts motion and interaction features from the observed trajectories. Two prediction heads then estimate the initialization centers of the goal and motion latents as $\mathbf{H} = \mathcal{E}_\psi(\mathbf{X})$, $\bar{\mathbf{z}}_g = f_g(\mathbf{H})$, and $\bar{\mathbf{z}}_m = f_m(\mathbf{H}, \bar{\mathbf{z}}_g)$. Here, $\mathbf{H}$ denotes the history features, while $\bar{\mathbf{z}}_g$ and $\bar{\mathbf{z}}_m$ are the initialization centers rather than the final predictions. Conditioning the motion center on the estimated goal allows the initialization to adapt local motion patterns to different goal positions.

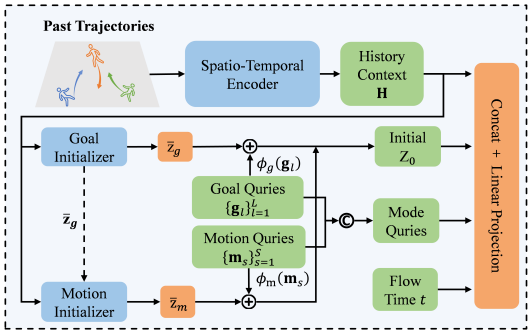

Figure 3: Latent initialization in GMFlow. History features and factorized queries produce initial latents and mode features for conditional flow generation.

To represent multiple possible futures, we introduce L learnable goal queries $\{\mathbf{g}_l\}_{l=1}^{L}$ and S learnable motion queries $\{\mathbf{m}_s\}_{s=1}^{S}$. Both query sets are randomly initialized and learned during the second training stage. For each query pair (l, s), lightweight projection heads construct its initial latent $\mathbf{z}_{i,0}^{(l,s)} = [\bar{\mathbf{z}}_{g,i} + \phi_g(\mathbf{g}_l); \bar{\mathbf{z}}_{m,i} + \phi_m(\mathbf{m}_s)]$ and fused mode feature $\mathbf{q}_{l,s} = \phi_q([\mathbf{g}_l; \mathbf{m}_s])$. The two outputs serve complementary roles. The projected queries determine the candidate-specific initial state, while $\mathbf{q}_{l,s}$ is provided to the shared flow network throughout generation as a persistent condition for the corresponding goal–motion combination. Although the initialization is factorized, the shared conditional flow network subsequently updates the goal and motion components jointly. Combining all L goal queries with all S motion queries produces $K = LS$ candidates. Their initial latents and mode features are stacked along the candidate dimension to obtain $\mathbf{Z}_{i,0}$ and $\mathbf{Q}$, respectively.

3.3.2 Conditional Flow Network

Let $\mathbf{z}_i^*$ denote the normalized joint latent obtained from the ground-truth future trajectory. During training, we construct a linear path from each candidate initialization to $\mathbf{z}_i^*$. Replicating the ground-truth latent along the candidate dimension gives $\mathbf{Z}_{i,t} = (1-t)\mathbf{Z}_{i,0} + t\mathbf{z}_i^*$ with $t \sim \mathcal{U}(0, 1)$, the target velocity is $\mathbf{V}_i^* = \mathbf{z}_i^* - \mathbf{Z}_{i,0}$, and the network prediction is $\hat{\mathbf{V}}_{i,t} = F_\theta(\mathbf{Z}_{i,t}, t, \mathbf{H}, \mathbf{Q})$. The conditional flow network F_θ predicts the velocities of all K candidates in a single forward pass. The mode features in $\mathbf{Q}$ distinguish different goal–motion combinations. The network first exchanges information among agents within each candidate and then exchanges information among candidates for each agent, allowing it to model both agent interactions and differences between future modes.

At inference time, the model constructs all candidate initializations in parallel and performs a single Euler update from $t = 0$ as $\mathbf{Z}_{i,1} = \mathbf{Z}_{i,0} + F_\theta(\mathbf{Z}_{i,0}, 0, \mathbf{H}, \mathbf{Q})$. The resulting joint latents are split

into goal and motion components, transformed back to their original scales, and passed to the goal-preserving decoder. GMFlow therefore generates all K candidate trajectories in a single flow step, without iterative integration or an additional model distillation stage.

3.4 Model Training

We train GMFlow in two stages. First, we train the history encoder and the two latent-center prediction heads without mode queries. We then freeze these components and jointly train the goal and motion queries, their projection and fusion heads, and the conditional flow network. The predicted centers are combined with learned query offsets to form candidate-specific initial states.

Latent initialization loss. Given a ground-truth future trajectory, we obtain its goal and motion latents, $\mathbf{z}_g^*$ and $\mathbf{z}_m^*$, using the representation defined in Section 3.2. The goal initialization head directly predicts the goal latent, while the motion initialization head is trained under both the predicted and ground-truth goal conditions. The initialization loss is $\mathcal{L}_{\text{init}} = \ell(\bar{\mathbf{z}}_g, \mathbf{z}_g^*) + (\lambda_m^{\text{init}}/2)[\ell(\bar{\mathbf{z}}_m, \mathbf{z}_m^*) + \ell(\bar{\mathbf{z}}_m^*, \mathbf{z}_m^*)]$, where $\bar{\mathbf{z}}_m^* = f_m(\mathbf{H}, \mathbf{z}_g^*)$ and ℓ denotes the smooth L_1 loss. Training the motion initializer under both conditions improves its robustness to goal estimation errors at inference time. After this stage, the parameters of the initialization module are fixed.

Component-aligned flow-matching loss. Unlike conventional trajectory-level supervision, we assign component-specific supervision according to the semantic roles of the goal and motion latents. The motion component is supervised by the candidate with the lowest ADE because it is responsible for modeling local variations over the full trajectory. For the goal component, we select the r candidates with the lowest FDE, allowing multiple candidates with accurate endpoints to receive goal supervision, and denote their index set by $\mathcal{S}_i$, where $|\mathcal{S}_i| = r$. The candidate weights are $w_{i,k}^m = \mathbb{I}[k = \arg\min_j a_{i,j}]$ and $w_{i,k}^g = \mathbb{I}[k \in \mathcal{S}_i] \exp(-e_{i,k}/\eta) / \sum_{j \in \mathcal{S}_i} \exp(-e_{i,j}/\eta)$, where $a_{i,k}$ and $e_{i,k}$ denote ADE and FDE of candidate k, respectively. η is a temperature parameter. Candidate errors and their corresponding weights are treated as constants during backpropagation. Both goal and motion assignments are performed at the candidate level, where each candidate corresponds to a goal–motion pair $k = (l, s)$.

We split the k-th row of $\hat{\mathbf{V}}_{i,t}$ into the predicted goal velocity $\hat{\mathbf{v}}_{g,i,t}^{(k)}$ and motion velocity $\hat{\mathbf{v}}_{m,i,t}^{(k)}$. The corresponding target velocities are $\mathbf{v}_{g,i}^{*(k)} = \mathbf{z}_{g,i}^* - \mathbf{z}_{g,i,0}^{(k)}$ and $\mathbf{v}_{m,i}^{*(k)} = \mathbf{z}_{m,i}^* - \mathbf{z}_{m,i,0}^{(k)}$. The conditional flow network is trained using

$$\mathcal{L}_{\text{FM}} = \mathbb{E} \sum_{k=1}^{K} \left[w_{i,k}^g \left\| \hat{\mathbf{v}}_{g,i,t}^{(k)} - \mathbf{v}_{g,i}^{*(k)} \right\|_2^2 + \lambda_m^{\text{flow}} w_{i,k}^m \left\| \hat{\mathbf{v}}_{m,i,t}^{(k)} - \mathbf{v}_{m,i}^{*(k)} \right\|_2^2 \right].$$

This component-aligned objective aligns the training signals with the intended roles of the two latent components: the goal component learns where to go through endpoint-aware supervision, while the motion component learns how to move through full-trajectory supervision.

4 Experiments

4.1 Experimental Settings

Datasets. We evaluate GMFlow on three public trajectory prediction benchmarks: NBA SportVU, the Stanford Drone Dataset (SDD), and ETH–UCY. NBA contains trajectories of ten players and the ball recorded during professional basketball games. Following MoFlow (Fu et al., 2025), we observe 10 frames (2 s) and predict the subsequent 20 frames (4 s), reporting results at prediction horizons of 1, 2, 3, and 4 seconds. SDD (Robicquet et al., 2016) provides annotated trajectories from bird's-eye-view videos recorded on the Stanford University campus. We use its preprocessed trajectory sequences, with 8 observed frames and 12 future frames. ETH–UCY (Pellegrini et al., 2009; Lerner et al., 2007) contains pedestrian trajectories from five scenes: ETH, HOTEL, UNIV, ZARA1, and ZARA2. We adopt the leave-one-scene-out protocol, holding out one scene for testing and using the remaining four for training. As on SDD, we observe 8 frames and predict the following 12 frames.

Table 1: Quantitative comparison on ETH–UCY. Results are reported as minADE_{20}/minFDE_{20} in meters; lower is better. AVG denotes the mean over the five scenes. Bold and underlined values indicate the best and second-best results, respectively.

Subset	PECNet (Mangalam et al., 2020)	MemoNet (Xu et al., 2022b)	MID (Gu et al., 2022)	LED (Mao et al., 2023)	Graph-TERN (Bae & Jeon, 2023)	EigenTrajectory (Bae et al., 2023)	HighGraph (Kim et al., 2024)	PPT (Lin et al., 2024)	MoFlow (Fu et al., 2025)	IAD (Liu et al., 2026)	Ours
ETH	0.54/0.87	0.40/0.61	0.39/0.66	0.39/0.58	0.42/0.58	0.36/0.53	0.40/0.55	0.36/ 0.51	0.40/0.57	0.34/0.52	**0.32/0.49**
HOTEL	0.18/0.24	**0.11**/0.17	0.13/0.22	**0.11**/0.17	0.14/0.23	0.12/0.19	0.13/0.17	**0.11/0.15**	**0.11**/0.17	0.15/0.24	0.12/ 0.16
UNIV	0.35/0.60	0.24/0.43	0.22/0.45	0.26/0.43	0.26/0.45	0.24/0.43	0.20/**0.33**	0.22/0.40	0.23/0.39	0.20/0.36	**0.19**/ 0.35
ZARA1	0.22/0.39	0.18/0.32	0.17/0.30	0.18/0.26	0.21/0.37	0.19/0.33	0.17/0.27	0.17/0.30	0.15/0.26	0.15/ 0.24	**0.13/0.21**
ZARA2	0.17/0.30	0.14/0.24	0.13/0.27	0.13/0.22	0.17/0.29	0.14/0.24	**0.11**/0.21	0.12/0.21	0.12/0.22	**0.11**/ 0.20	**0.11/0.18**
AVG	0.29/0.48	0.21/0.35	0.21/0.38	0.21/0.33	0.24/0.38	0.21/0.34	0.20/ 0.30	0.20/0.31	0.20/0.32	0.19/0.31	**0.17/0.28**

Table 2: Quantitative comparison on the NBA dataset. Results are reported as minADE_{20}/minFDE_{20} in meters; lower is better. Bold and underlined values denote the best and second-best results.

Time	Social-STGCNN (Mohamed et al., 2020)	PECNet (Mangalam et al., 2020)	Trajectron++ (Salzmann et al., 2020)	MemoNet (Xu et al., 2022b)	NPSN (Bae et al., 2022)	GroupNet (Xu et al., 2022a)	MID (Gu et al., 2022)	LED (Mao et al., 2023)	MoFlow-IMLE (Fu et al., 2025)	IAD (Liu et al., 2026)	GMFlow (Ours)
1.0 s	0.34/0.48	0.40/0.71	0.30/0.38	0.38/0.56	0.35/0.58	0.26/0.34	0.28/0.37	0.18/0.27	0.18/ 0.25	0.21/ 0.25	**0.16/0.23**
2.0 s	0.71/0.94	0.83/1.61	0.59/0.82	0.71/1.14	0.68/1.23	0.49/0.70	0.51/0.72	0.37/0.56	0.34/ 0.47	0.32/**0.45**	**0.31/0.45**
3.0 s	1.09/1.77	1.27/2.44	0.85/1.24	1.00/1.57	1.01/1.76	0.73/1.02	0.71/0.98	0.58/0.84	0.52/ 0.67	**0.48**/0.75	0.49/**0.65**
Total (4.0 s)	1.53/2.26	1.69/2.95	1.15/1.57	1.25/1.47	1.31/1.79	0.96/1.30	0.96/1.27	0.81/1.10	0.71/ 0.86	0.79/0.99	**0.68/0.80**

Evaluation metrics. Following prior work (Fu et al., 2025), we adopt average displacement error (ADE) and final displacement error (FDE) to evaluate prediction accuracy. ADE measures the mean Euclidean distance between predicted and ground-truth positions over all prediction time steps, while FDE measures the Euclidean distance between the predicted and ground-truth endpoints. For multimodal evaluation, we report minADE_{20} and minFDE_{20}, which are the minimum ADE and FDE, respectively, among 20 candidate trajectories for each agent.

4.2 Quantitative Evaluation

We compare GMFlow with representative trajectory prediction methods, including Social-STGCNN (Mohamed et al., 2020), PECNet (Mangalam et al., 2020), Trajectron++ (Salzmann et al., 2020), Y-Net (Mangalam et al., 2021), MemoNet (Xu et al., 2022b), NPSN (Bae et al., 2022), GroupNet (Xu et al., 2022a), MID (Gu et al., 2022), LED (Mao et al., 2023), EigenTrajectory (Bae et al., 2023), Graph-TERN (Bae & Jeon, 2023), HighGraph (Kim et al., 2024), PPT (Lin et al., 2024), MoFlow (Fu et al., 2025) and IAD (Liu et al., 2026).

We present the quantitative results on the three benchmarks in Tables 1, 2, and 3. As shown in Table 2, our method performs particularly well on NBA, achieving the lowest minFDE at all four prediction horizons and the lowest minADE at 1, 2, and 4 seconds. Compared with one-step distilled MoFlow, the four-second prediction errors are reduced by 4.2% in minADE and 7.0% in minFDE. On ETH–UCY, our method achieves the best average minADE/minFDE of 0.17/0.28 across the five scenes. The results on SDD further show its effectiveness, with minADE and minFDE of 6.69 and 10.28, respectively, improving on the previous best results of 6.85 and 10.65.

4.3 Qualitative Evaluation

We further provide qualitative comparisons on NBA and ETH–UCY to examine whether the predicted trajectories recover both future destinations and local path shapes.

As shown in Figure 4, our predictions follow the ground-truth trajectories more closely in the NBA scene. Differences are most apparent in curved player trajectories and ball movements, where several baseline predictions deviate from the true paths, while ours better capture the changes in direction and final positions. Figure 5 presents examples from four ETH–UCY scenes. For pedestrians moving along nearly straight paths, our predictions remain close to the ground truth. In cases involving turns, they also follow the path curvature and reach endpoints near the true destinations.

4.4 Ablation Studies

Table 3: Quantitative comparison on SDD. Results are reported as $\text{minADE}_{20}/\text{minFDE}_{20}$ in pixels; lower is better. †Y-Net uses additional scene context.

Metric	PECNet (Mangalam et al., 2020)	Y-Net† (Mangalam et al., 2021)	MemoNet (Xu et al., 2022b)	Graph-TERN (Bae & Jeon, 2023)	MID (Gu et al., 2022)	LED (Mao et al., 2023)	EigenTrajectory (Bae et al., 2023)	PPT (Lin et al., 2024)	MoFlow (Fu et al., 2025)	IAD (Liu et al., 2026)	GMFlow (Ours)
$\text{minADE}_{20}/\text{minFDE}_{20}$ (pixels)	9.96/15.88	7.85/11.85	8.56/12.66	8.43/14.26	7.61/14.30	8.48/11.66	8.05/13.25	7.03/ 10.65	7.50/11.96	6.85/11.22	**6.69/10.28**

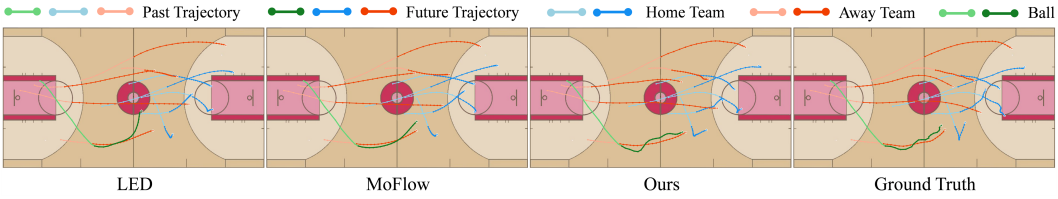

Figure 4: Qualitative comparison on NBA. We visualize the observed trajectories and the future trajectories predicted by LED, MoFlow, and GMFlow, together with the ground truth. Colors distinguish the home team, away team, and ball. GMFlow more closely follows the future motion directions and endpoints of the interacting players.

We conduct ablation studies on NBA to assess the contribution of individual components and the effect of the goal–motion representation configuration. Unless otherwise specified, we use $M = 2$ goal timestamps, corresponding to one intermediate goal and the final goal. The goal-preserving motion basis has dimension $q = 8$, resulting in a four-dimensional goal latent and a 16-dimensional motion latent. We standardize each latent dimension using its mean and standard deviation on the training set, with standard deviations clamped to a minimum of 10^{-4}, and invert this transformation before decoding.

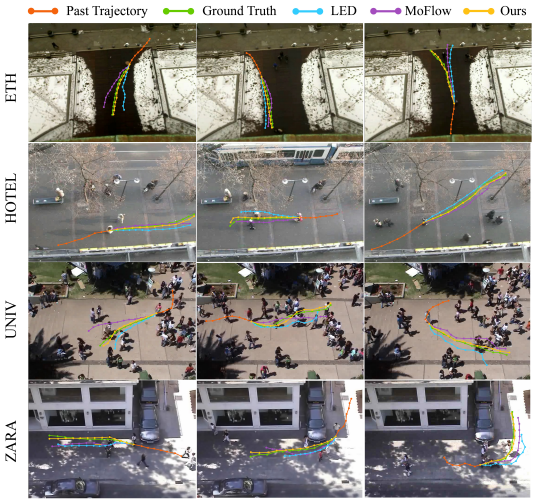

Figure 5: Qualitative results on ETH–UCY. Rows: ETH, HOTEL, UNIV, and ZARA.

Core components. Table 4 compares the full model with four variants: (1) **w/o Goal–Motion Decomposition**, which replaces the two latent components with a unified residual latent of the same total dimension; (2) **w/o Goal-Preserving Constraint**, which removes the requirement that the motion basis vanish at goal timestamps; (3) **w/o Factorized Queries**, which uses K independent mode queries, allowing each candidate to learn its own goal and motion offsets rather than sharing them across query pairs; and (4) **w/o Component-Aligned Supervision**, which uses a common candidate assignment criterion based on full-trajectory error for both components.

The results in Table 4 show that removing any component increases both errors. Replacing the goal–motion decomposition with a unified residual latent increases minADE/minFDE from 0.683/0.801 to 0.725/0.856, supporting the proposed structured representation. Removing the goal-preserving constraint leads to a larger increase in minADE. Unlike the unified baseline, this variant retains the two latent components but allows motion updates to alter the positions specified by the goal latent. The result suggests that separating the latent variables alone is insufficient; their distinct roles also need to be maintained during decoding. The degradation under a shared assignment criterion further supports the use of component-specific supervision when learning this representation.

Representation configuration. Table 5 compares combinations of goal timestamps $M \in \{1, 2, 4\}$ and motion-basis dimensions $q \in \{4, 8, 12\}$ to examine how goal temporal granularity and motion capacity affect prediction accuracy. The corresponding goal and motion latents have dimensions

$2M$ and $2q$, respectively. The results suggest that intermediate goal structure is beneficial, while increasing goal granularity further does not consistently improve performance, possibly because overly dense goals absorb variations that can instead be modeled by the local motion component.

Query configuration. Table 6 compares different combinations of goal queries L and motion queries per goal S, keeping the total number of candidates fixed at $K = LS = 20$. This allows us to examine the trade-off between covering more goal hypotheses and generating more motion variations for each goal. Increasing the number of goal queries generally improves minFDE, suggesting better coverage of goal uncertainty. Meanwhile, retaining multiple motion queries per goal yields the best minADE at 10×2, supporting the benefit of modeling motion uncertainty within each goal.

Table 4: Core-component ablations on NBA. Results are minADE_{20}/minFDE_{20} at 4s.

Variant	minADE_{20}	minFDE_{20}
w/o Goal–Motion Decomposition	0.725	0.856
w/o Goal-Preserving Constraint	0.746	0.858
w/o Factorized Queries	0.717	0.849
w/o Component-Aligned Supervision	0.733	0.854
Full GMFlow	**0.683**	**0.801**

Table 5: Goal–motion configuration on NBA (minADE_{20}/minFDE_{20} at 4s).

$M\backslash q$	**4**	**8**	**12**
1	0.729/0.851	0.720/0.852	0.724/0.855
2	0.725/0.809	**0.683/0.801**	0.724/0.804
4	0.722/0.821	0.697/0.823	0.718/0.820

Table 6: Query allocation on NBA at 4s ($LS = 20$).

$L \times S$	minADE_{20}	minFDE_{20}
1×20	0.716	0.849
5×4	0.702	0.823
$\mathbf{10 \times 2}$	**0.683**	0.801
20×1	0.691	**0.790**

Inference analysis. One-step GMFlow outperforms direct latent residual regression under matched architecture and training settings (Table 7). Four Euler steps provide only small gains, while eight increase errors, supporting our one-step choice. Table 8 compares inference cost with MID (Gu et al., 2022), LED (Mao et al., 2023), and MoFlow variants (Fu et al., 2025). GMFlow uses one prediction-network evaluation (NFE) and takes 0.36 ms.

Table 7: Regression and Euler steps on NBA at 4s.

Method	NFE	minADE_{20}	minFDE_{20}
Direct regression	1	0.784	0.961
GMFlow (1 step)	1	0.683	0.801
GMFlow (2 steps)	2	0.681	0.797
GMFlow (4 steps)	4	**0.680**	**0.793**
GMFlow (8 steps)	8	0.685	0.806

Table 8: Inference cost on NBA.

Method	NFE	Time (ms)
MID	100	796.00
LED	6	22.38
MoFlow	100	33.20
MoFlow-IMLE	1	0.70
GMFlow (Ours)	1	**0.36**

5 Conclusion

In this paper, we presented GMFlow, a goal–motion latent flow framework for multimodal trajectory prediction. It decomposes trajectory residuals into goal and motion latents, with a goal-preserving decoder that refines local paths while preserving specified goal positions. Factorized queries and component-aligned supervision guide joint latent generation, enabling single-step prediction without additional distillation. Experiments on NBA, ETH–UCY, and SDD demonstrate state-of-the-art performance, particularly in long-term endpoint prediction. Ablation studies validate the key design choices, highlighting the value of structured goal–motion representations for accurate and efficient prediction.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language editing, improving readability, refining the presentation of the manuscript, and providing feedback on the clarity of the methodology and experimental presentation. We reviewed all AI-assisted suggestions and independently verified the technical content, experimental results, and scientific claims. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 961–971, 2016.

Inhwan Bae and Hae-Gon Jeon. A set of control points conditioned pedestrian trajectory prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 6155–6165, 2023.

Inhwan Bae, Jin-Hwi Park, and Hae-Gon Jeon. Non-probability sampling network for stochastic human trajectory prediction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6467–6477. IEEE, 2022.

Inhwan Bae, Jean Oh, and Hae-Gon Jeon. EigenTrajectory: Low-rank descriptors for multi-modal trajectory forecasting. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9983–9995. IEEE, 2023.

Inhwan Bae, Young-Jae Park, and Hae-Gon Jeon. SingularTrajectory: Universal trajectory predictor using diffusion model. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17890–17901. IEEE, 2024.

Patrick Dendorfer, Sven Elflein, and Laura Leal-Taixé. MG-GAN: A multi-generator model preventing out-of-distribution samples in pedestrian trajectory prediction. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13138–13147. IEEE, 2021.

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

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17282–17293. IEEE, 2025.

Tianpei Gu, Guangyi Chen, Junlong Li, Chunze Lin, Yongming Rao, Jie Zhou, and Jiwen Lu. Stochastic trajectory prediction via motion indeterminacy diffusion. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17092–17101. IEEE, 2022.

Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially acceptable trajectories with generative adversarial networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2255–2264. IEEE, 2018.

Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51: 4282, 1995.

Yingfan Huang, Huikun Bi, Zhaoxin Li, Tianlu Mao, and Zhaoqi Wang. STGAT: Modeling spatial-temporal interactions for human trajectory prediction. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6271–6280. IEEE, 2019.

Ronny Hug, Wolfgang Hübner, and Michael Arens. Introducing probabilistic bézier curves for n-step sequence prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 10162–10169, 2020.

Ronny Hug, Stefan Becker, Wolfgang Hübner, Michael Arens, and Jürgen Beyerer. Bézier curve gaussian processes. *arXiv preprint arXiv:2205.01754*, 2022.

Ioannis Karamouzas, Brian Skinner, and Stephen J Guy. Universal power law governing pedestrian interactions. *Physical review letters*, 113:238701, 2014.

Sungjune Kim, Hyung-gun Chi, Hyerin Lim, Karthik Ramani, Jinkyu Kim, and Sangpil Kim. Higher-order relational reasoning for pedestrian trajectory prediction. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15251–15260. IEEE, 2024.

Parth Kothari, Brian Sifringer, and Alexandre Alahi. Interpretable social anchors for human trajectory forecasting in crowds. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15551–15561. IEEE, 2021.

Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. *Computer Graphics Forum*, 26:655–664, 2007.

Rongqin Liang, Yuanman Li, Xia Li, Yi Tang, Jiantao Zhou, and Wenbin Zou. Temporal pyramid network for pedestrian trajectory prediction with multi-supervision. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 2029–2037, 2021.

Xiaotong Lin, Tianming Liang, Jianhuang Lai, and Jian-Fang Hu. Progressive pretext task learning for human trajectory prediction. In *European Conference on Computer Vision*, pp. 197–214. Springer, 2024.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *International Conference on Learning Representations*, 2023.

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, pp. 18469–18477, 2026.

Matthias Luber, Johannes A Stork, Gian Diego Tipaldi, and Kai O Arras. People tracking with human motion predictions from social forces. In *2010 IEEE international conference on robotics and automation*, pp. 464–469. IEEE, 2010.

Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In *European conference on computer vision*, pp. 759–776. Springer, 2020.

Karttikeya Mangalam, Yang An, Harshayu Girase, and Jitendra Malik. From goals, waypoints & paths to long term human trajectory forecasting. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pp. 15213–15222. IEEE, 2021.

Weibo Mao, Chenxin Xu, Qi Zhu, Siheng Chen, and Yanfeng Wang. Leapfrog diffusion model for stochastic trajectory prediction. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5517–5526. IEEE, 2023.

Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-STGCNN: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 14412–14420. IEEE, 2020.

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*, pp. 261–268. IEEE, 2009.

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

Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. SoPhie: An attentive gan for predicting paths compliant to social and physical constraints. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 1349–1358. IEEE, 2019.

Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *European conference on computer vision*, pp. 683–700. Springer, 2020.

Zhaoen Su, Chao Wang, Henggang Cui, Nemanja Djuric, Carlos Vallespi-Gonzalez, and David Bradley. Temporally-continuous probabilistic prediction using polynomial trajectory parameterization. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3837–3843. IEEE, 2021.

Conghao Wong, Beihao Xia, Ziming Hong, Qinmu Peng, Wei Yuan, Qiong Cao, Yibo Yang, and Xinge You. View vertically: A hierarchical network for trajectory prediction via fourier spectrums. In *European Conference on Computer Vision*, pp. 682–700. Springer, 2022.

Conghao Wong, Ziqian Zou, and Beihao Xia. Resonance: Learning to predict social-aware pedestrian trajectories as co-vibrations. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 25788–25799. IEEE, 2025.

Chenxin Xu, Maosen Li, Zhenyang Ni, Ya Zhang, and Siheng Chen. GroupNet: Multiscale hypergraph neural networks for trajectory prediction with relational reasoning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6488–6497. IEEE, 2022a.

Chenxin Xu, Weibo Mao, Wenjun Zhang, and Siheng Chen. Remember intentions: Retrospective-memory-based trajectory prediction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6478–6487. IEEE, 2022b.

Kota Yamaguchi, Alexander C Berg, Luis E Ortiz, and Tamara L Berg. Who are you with and where are you going? In *CVPR 2011*, pp. 1345–1352. IEEE, 2011.

Cunjun Yu, Xiao Ma, Jiawei Ren, Haiyu Zhao, and Shuai Yi. Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In *European conference on computer vision*, pp. 507–523. Springer, 2020.

Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. AgentFormer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9813–9823, 2021.

Jianjing Zhang, Hongyi Liu, Qing Chang, Lihui Wang, and Robert X Gao. Recurrent neural network for motion trajectory prediction in human-robot collaborative assembly. *CIRP annals*, 69:9–12, 2020.

Pu Zhang, Wanli Ouyang, Pengfei Zhang, Jianru Xue, and Nanning Zheng. SR-LSTM: State refinement for lstm towards pedestrian trajectory prediction. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12077–12086. IEEE, 2019.

Zhanwei Zhang, Zishuo Hua, Minghao Chen, Wei Lu, Binbin Lin, Deng Cai, and Wenxiao Wang. G2LTraj: A global-to-local generation approach for trajectory prediction. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 2579–2587, 2024.

Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Ben Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, et al. TNT: Target-driven trajectory prediction. In *Conference on robot learning*, pp. 895–904. PMLR, 2021.

A Additional Method Details

A.1 Piecewise-Linear Goal Interpolation

We select M goal times over the prediction horizon, $\tau_m = (m+1)T_{\text{pred}}/M$ for $m = 0, \dots, M-1$, where $\tau_{M-1} = T_{\text{pred}}$. The corresponding goal offsets are $\mathbf{G}_i[m,:] = \mathbf{R}_i[\tau_m,:]$. For the first interval, the goal path linearly connects the zero residual at the beginning of prediction to the first goal offset:

$$(\mathbf{AG}_i)[\tau,:] = \frac{\tau}{\tau_0}\mathbf{G}_i[0,:], \qquad 1 \le \tau \le \tau_0. \tag{1}$$

For each subsequent interval $\tau_{m-1} < \tau \leq \tau_m$, where $m = 1, \ldots, M-1$, the path is obtained by interpolating between two adjacent goal offsets:

$$(\mathbf{AG}_i)[\tau, :] = \frac{\tau_m - \tau}{\tau_m - \tau_{m-1}} \mathbf{G}_i[m-1, :] + \frac{\tau - \tau_{m-1}}{\tau_m - \tau_{m-1}} \mathbf{G}_i[m, :]. \tag{2}$$

At each goal time τ_m, the interpolation satisfies $(\mathbf{AG}_i)[\tau_m, :] = \mathbf{G}_i[m, :]$. The interpolated path therefore passes through all intermediate and final goal offsets.

A.2 Construction of the Goal-Preserving Motion Basis

We construct the motion basis from the complete orthonormal DCT-II basis $\mathbf{C} \in \mathbb{R}^{T_{\text{pred}} \times T_{\text{pred}}}$, whose elements are

$$C_{\tau,k} = \alpha_k \cos\left(\frac{\pi k(2\tau - 1)}{2T_{\text{pred}}}\right), \qquad \tau = 1, \ldots, T_{\text{pred}}, \quad k = 0, \ldots, T_{\text{pred}} - 1, \tag{3}$$

where $\alpha_0 = 1/\sqrt{T_{\text{pred}}}$ and $\alpha_k = \sqrt{2/T_{\text{pred}}}$ for $k > 0$. The columns are ordered from low to high frequency. However, a standard DCT basis generally has nonzero values at the selected goal times. Directly using this basis would therefore allow the motion component to change the positions specified by the goal latent.

Let $\mathcal{T}_g = \{\tau_0, \ldots, \tau_{M-1}\}$ and $\mathcal{T}_r = \{1, \ldots, T_{\text{pred}}\} \setminus \mathcal{T}_g$ denote the goal and non-goal time sets. We retain the rows corresponding to the non-goal times and perform a reduced QR decomposition in the original frequency order: $\mathbf{C}_r = \mathbf{C}[\mathcal{T}_r, :] = \mathbf{QR}$. We then retain the first q columns of $\mathbf{Q}$, place them back at the non-goal time steps, and set the goal rows to zero:

$$\mathbf{D}[\mathcal{T}_r, :] = \mathbf{Q}_{:,1:q}, \qquad \mathbf{D}[\mathcal{T}_g, :] = \mathbf{0}, \qquad q \leq T_{\text{pred}} - M. \tag{4}$$

Because the retained columns of $\mathbf{Q}$ are orthonormal and zero-padding does not change their inner products, $\mathbf{D}^\top \mathbf{D} = \mathbf{I}_q$. In addition, $\mathbf{D}[\tau_m, :] = \mathbf{0}$ at every goal time. Therefore, for any motion coefficient matrix $\mathbf{U}_i$, $(\mathbf{DU}_i)[\tau_m, :] = \mathbf{0}$. This construction ensures that the motion component cannot change any position specified by the goal latent.

A.3 Low-Dimensional Motion Reconstruction

The local residual not explained by the goal path is $\mathbf{E}_i = \mathbf{R}_i - \mathbf{AG}_i$. Since the goal path passes through all selected goal offsets, $\mathbf{E}_i[\tau_m, :] = \mathbf{0}$. The motion coefficients are obtained by projecting the local residual onto the motion basis: $\mathbf{U}_i = \mathbf{D}^\top \mathbf{E}_i$. The reconstructed motion component is $\tilde{\mathbf{E}}_i = \mathbf{DU}_i = \mathbf{DD}^\top \mathbf{E}_i$. When $q = T_{\text{pred}} - M$, the basis covers the complete space of trajectories that are zero at all goal times and exactly reconstructs $\mathbf{E}_i$. When $q < T_{\text{pred}} - M$, it gives a low-dimensional orthogonal projection.

The residual trajectory is therefore represented as $\mathbf{R}_i \approx \mathbf{AG}_i + \mathbf{DU}_i$, and the complete future trajectory is reconstructed as $\hat{\mathbf{y}}_i = \mathbf{B}_i + \mathbf{A}\hat{\mathbf{G}}_i + \mathbf{D}\hat{\mathbf{U}}_i$. The goal-preserving property can equivalently be expressed as $\partial \hat{\mathbf{y}}_i[\tau_m, :] / \partial \mathbf{z}_{m,i} = \mathbf{0}$.

A.4 Information Exchange in the Flow Network

Suppressing the batch dimension, we stack the candidate states of all agents into $\mathbf{Z}_t \in \mathbb{R}^{N \times K \times d_z}$, where $d_z = 2M + 2q$ and $K = LS$. For candidate $k = (l, s)$, the goal and motion queries are fused into $\mathbf{q}_k = \phi_q([\mathbf{g}_l; \mathbf{m}_s]) \in \mathbb{R}^h$. For each agent–candidate pair, the network concatenates the current joint latent $\mathbf{Z}_t[i, k, :]$, a sinusoidal embedding $\gamma(t) \in \mathbb{R}^h$ of the flow time, the agent history feature $\mathbf{H}_i \in \mathbb{R}^h$, and the persistent mode feature $\mathbf{q}_k$. A linear input projection maps this concatenation to the initial hidden feature $\mathbf{S}_t^{(0)}[i, k, :] \in \mathbb{R}^h$.

The network first applies an Agent Transformer independently to each candidate. In implementation, $\mathbf{S}_t^{(0)}$ is reshaped to $[BK, N, h]$, so self-attention operates over the N agents while the candidate index remains fixed. An agent mask prevents padded agents from participating in the interaction. This produces agent-aware features $\mathbf{S}_t^{(a)}$ for every candidate.

The features are then rearranged from $[BK, N, h]$ to $[BN, K, h]$ and passed through a Mode Transformer. Self-attention now operates over the K candidates while the agent index remains fixed, allowing the different goal–motion combinations of the same agent to exchange information and reduce redundant predictions. No additional positional or identity embedding is used on the candidate axis because the fused mode features already distinguish the candidates. The output is finally restored to the agent–candidate layout and mapped by an MLP to $\hat{\mathbf{V}}_t \in \mathbb{R}^{N\times K\times d_z}$. The per-agent tensors $\mathbf{Z}_{i,t}$ and $\hat{\mathbf{V}}_{i,t}$ in Section 3.3 denote slices of these global tensors.

We use $h = 128$ in all experiments. Both the Agent Transformer and Mode Transformer contain two layers with four attention heads. Each layer uses a feed-forward dimension of $4h$, GELU activation, pre-normalization, and dropout of 0.1. At inference time, all K candidate latents are generated in parallel by a single network forward pass.

A.5 Implementation Details

Candidate assignment during flow training. For each training example, we construct the K candidate source states and evaluate the prediction network at $t = 0$ to obtain one-step candidate predictions. The resulting predicted latents are inverse-normalized and decoded into trajectories $\widetilde{\mathbf{y}}_i^{(k)}$. Candidate errors are evaluated in the same coordinate system as the displacement metrics:

$$\begin{aligned} a_{i,k} &= \frac{1}{T_{\text{pred}}} \sum_{\tau=1}^{T_{\text{pred}}} \left\| \widetilde{\mathbf{y}}_i^{(k)}[\tau,:] - \mathbf{y}_i[\tau,:] \right\|_2, \\ e_{i,k} &= \left\| \widetilde{\mathbf{y}}_i^{(k)}[T_{\text{pred}},:] - \mathbf{y}_i[T_{\text{pred}},:] \right\|_2. \end{aligned} \tag{5}$$

These errors are computed from the source-state predictions, rather than from predictions extrapolated from the ground-truth-interpolated state $\mathbf{Z}_{i,t}$. Ground truth is used to score the assignment predictions; it is not an input to this source-state prediction branch.

For each agent, the candidate with the lowest ADE receives motion supervision, and the r candidates with the lowest FDE receive normalized goal weights. The candidate weights are detached before weighting the matching errors evaluated at the sampled time t. Thus, gradients do not propagate through the assignment weights. The additional source-state evaluation is used during training; inference still requires only one prediction-network evaluation for all K candidates.

Hyperparameters. Table 9 summarizes the implementation settings shared across NBA, SDD, and ETH–UCY, unless otherwise specified.

Table 9: Architecture and training hyperparameters used across the three benchmarks.

Setting	Value
Architecture	
Hidden dimension h	128
History encoder blocks	2
Flow Agent Transformer layers	2
Flow Mode Transformer layers	2
Attention heads per layer	4
Feed-forward dimension	512
Transformer dropout	0.1
Representation and candidate assignment	
Goal timestamps M	2
Motion basis dimension q	8
Goal latent dimension	4
Motion latent dimension	16
Goal queries L	10
Motion queries S	2

Continued on next page

Table 9 continued

Setting	Value
Prediction candidates K	20
Goal assignment top-r	2
Assignment temperature η	0.1
Latent standard deviation floor	10^{-4}
Loss weights	
Initialization motion loss weight λ_m^{init}	0.25
Flow motion loss weight λ_m^{flow}	1.0
Training and inference	
Optimizer	AdamW
Learning rate (both stages)	2×10^{-4}
Weight decay	10^{-4}
Maximum gradient norm	1.0
Initialization training epochs	20
Initialization batch size	32
Flow training epochs	66
Flow batch size	16
Flow learning rate schedule	Constant
EMA decay (flow training)	0.999
Flow-network evaluations at inference	1

Unless otherwise specified, we use $M = 2$ goal timestamps, corresponding to one intermediate goal and the final goal. The goal-preserving motion basis has dimension $q = 8$, resulting in a four-dimensional goal latent and a 16-dimensional motion latent.

The conditional flow network has a hidden dimension of $h = 128$. Both the Agent Transformer and the Mode Transformer contain two layers with four attention heads. Each layer uses a feed-forward dimension of $4h$, GELU activation, pre-normalization, and a dropout rate of 0.1.

GMFlow is trained in two stages: we first optimize the latent initialization module and then freeze it while training the conditional flow network. At inference time, GMFlow performs a single network evaluation followed by one Euler update to generate all K candidate trajectories in parallel.

A.6 Direct Regression and Integration Steps

We compare flow matching with direct latent residual regression on NBA. The regression baseline retains the history encoder, latent initialization, mode queries, and prediction network architecture. Given the initial latent $Z_{i,0}$, it predicts the residual to the target latent at $t = 0$, and reconstructs the prediction by adding this residual to $Z_{i,0}$. The baseline is trained separately using the same candidate assignment strategy and training budget. All methods generate $K = 20$ candidate trajectories.

For the integration-step analysis, we use the same trained GMFlow checkpoint and candidate initializations and vary only the number of Euler steps. With J steps, the latent states are updated as

$$\mathbf{Z}_{i,t_{j+1}} = \mathbf{Z}_{i,t_j} + \frac{1}{J} F_\theta \left(\mathbf{Z}_{i,t_j}, t_j, \mathbf{H}, \mathbf{Q} \right), \qquad t_j = \frac{j}{J}, \quad j = 0, \ldots, J-1. \tag{6}$$

Table 7 in the main paper reports the direct-regression and integration-step comparisons on NBA at the four-second prediction horizon.