DiffVec: Diffusion Model for Trajectory Vector Recovery

Jiaqi Duan[1] , **Shengwei Tian***[1] , **Long Yu***[1] and **Xiangfu Meng**[2] and **Ya Zhang**[3]
[1]Xin Jiang University
[2]Liao Ning Technology University
[3]Tarim University
d_jq@foxmail.com, tianshengwei@163.com, yul_xju@163.com, marxi@126.com, zhya99@foxmail.com

Abstract

The increasing availability of trajectory data is often hampered by sparsity and noise. Existing trajectory recovery methods are further limited by either information loss from coordinate discretization into location IDs, or the inefficiency of conventional diffusion models that require a lengthy denoising process from pure noise. To address these challenges, we propose DiffVec, a novel and efficient diffusion framework for free-space trajectory recovery that operates directly on continuous coordinate data. The backbone of our framework is MVformer, a Transformer-based architecture that models motion vectors—the differences between consecutive coordinates—to better capture the local motion patterns of movement. This model is trained within our ResTraj diffusion paradigm, which commences the denoising process from a structured, interpolated prior to significantly reduce sampling steps. Extensive experiments on the Geolife and Porto datasets demonstrate that DiffVec significantly outperforms state-of-the-art baselines across MAE, MSE, and NDTW.

1 Introduction

The proliferation of mobile devices has generated massive trajectory data, enabling applications like next-location recommendation [Duan *et al.*, 2024] and urban planning. However, raw data often suffers from sparsity and noise due to device constraints and privacy concerns, limiting its direct utility. This necessitates *trajectory recovery*. Distinct from map-matching approaches restricted to road networks, we focus on the general *free-space* setting. This is vital for multi-modal scenarios (e.g., pedestrians, vessels) where underlying graph constraints are unavailable or inapplicable.

With the rapid advancement of deep learning, trajectory recovery methodologies have evolved significantly. Initially, rule-based approaches like linear interpolation [Li *et al.*, 2000] and polygon interpolation were widely used. However, these methods often fail to capture complex, non-linear mobility patterns. To address this, deep learning-based methods emerged. Early works employed Recurrent Neural Networks to model sequential dependencies. Subsequently, attention-based models and Transformers, such as AttnMove [Xia *et al.*, 2021] and TrajBERT [Si *et al.*, 2023], became the dominant paradigm. Most recently, Generative AI has brought Diffusion Probabilistic Models into this domain. By learning the underlying data distribution, models like DiffTraj [Zhu *et al.*, 2023] and ControlTraj [Zhu *et al.*, 2024] have demonstrated a superior capability to generate high-quality trajectory samples.

However, despite these advancements, current research on trajectory recovery still faces several key challenges:

- **Inefficiency in Trajectory Diffusion Models:** The standard "noise-to-data" diffusion paradigm fails to utilize observed segments of sparse trajectories as structural priors, ignoring local temporal correlations and causing severe computational inefficiency.
- **Suboptimal Utilization of Trajectory Coordinates:** Existing representations fail to fully utilize continuous trajectory coordinates, as raw absolute coordinates introduce non-stationary distributions that hinder the learning of consistent local geometries.

To address the first challenge, we adopt a diffusion model-based approach but depart from the standard "noise-to-data" paradigm. We propose the **ResTraj** strategy. **The core intuition** is that trajectory recovery should be treated as a "refinement" process rather than a "creation" process. Simple heuristic methods, like linear interpolation, can essentially recover the rough "skeleton" of a trajectory. Therefore, starting the diffusion process from pure noise is counter-intuitive. Instead, ResTraj bridges the gap between this coarse structural prior and the ground truth. By learning to predict the *residuals*—the non-linear deviations needed to refine the linear skeleton into the true path—our model simplifies the learning objective significantly, allowing for high-fidelity generation with a fraction of the sampling steps.

To address the second challenge, we introduce the **MVformer** (Motion Vector Transformer) backbone. **The intuition** behind this design stems from our observation that motion vectors possess more desirable properties compared to raw coordinates, particularly for Transformer architectures. As illustrated in Figure 1, raw coordinate vectors originate from the origin, leading to large absolute magnitudes and extremely limited angular variation (Figure 1(a, c)). In contrast,

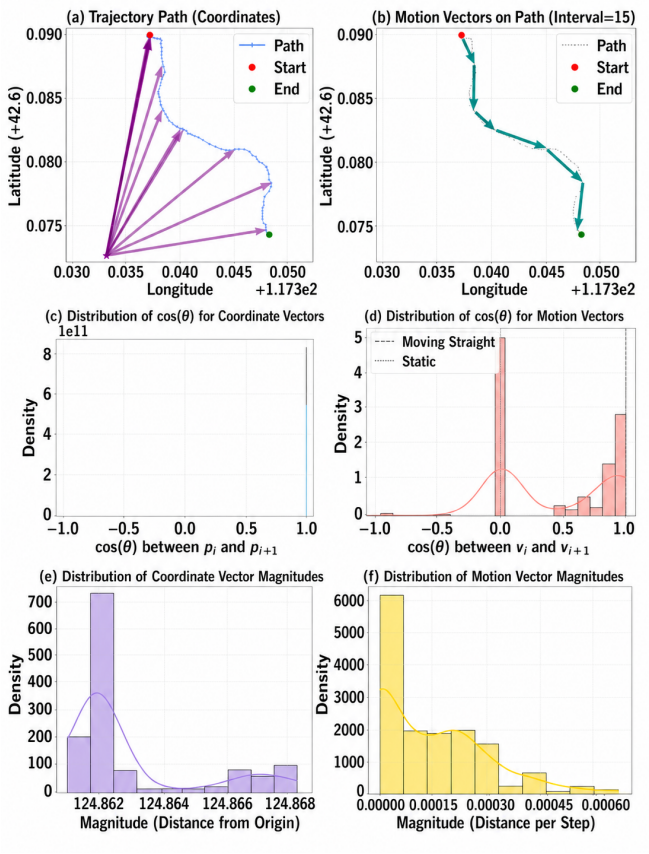

Figure 1: Comparison of geometric properties between Raw Coordinates and Motion Vectors. (a) and (c) show that raw coordinates suffer from non-stationary distributions and limited angular variation. In contrast, (b) and (d) demonstrate that motion vectors (differences between points) follow a regular distribution, making them more suitable for modeling local motion patterns.

motion vectors (Figure 1(b)) are **inherently suited for the dot-product self-attention mechanism**. Since the attention score is derived from the dot product $\mathbf{q} \cdot \mathbf{k} = \|\mathbf{q}\|\|\mathbf{k}\| \cos(\theta)$, motion vectors allow the model to explicitly leverage *vector magnitudes* (representing instantaneous speed) and *cosine angles* (representing directional changes) to capture local motion patterns. This geometric richness, combined with a stable Gaussian distribution (Figure 1(f)), enables MVformer to learn geometrically coherent movement patterns rather than merely memorizing absolute positions.

In summary, our main contributions can be summarized as follows:

- We propose **ResTraj**, a residual diffusion paradigm that initiates generation from an interpolated prior. This strategy aligns with the intuition of "refinement over creation," significantly accelerating convergence.
- We design **MVformer**, a Transformer backbone that models motion vectors instead of coordinates. This representation effectively captures the local motion patterns and improves geometric stability.
- We conduct extensive experiments on the Geolife and Porto datasets. The results demonstrate that DiffVec outperforms state-of-the-art baselines in recovery accuracy.

2 Related Work

2.1 Deep Learning for Trajectory Recovery

Trajectory recovery has evolved from heuristic interpolation [Li *et al.*, 2000] to data-driven deep learning. Early methods employed Recurrent Neural Networks to capture sequential dependencies [Cao *et al.*, 2018], while recent attention-based approaches have become dominant. Notable examples include **AttnMove** [Xia *et al.*, 2021] for history-enhanced recovery and **SAITS** [Du *et al.*, 2023], which utilizes diagonally-masked self-attention for precise time-series imputation. Similarly, **TrajBERT** [Si *et al.*, 2023; Musleh and Mokbel, 2023] adapts the BERT architecture to model bidirectional mobility contexts, and **RNTrajRec** [Chen *et al.*, 2023] further integrates road network constraints. A defining characteristic of these state-of-the-art methods is their reliance on spatial discretization, where continuous GPS coordinates are mapped into discrete location IDs (e.g., grid cells) to simplify the modeling vocabulary. Beyond standard architectures, recent works have also explored Large Language Models [Wang *et al.*, 2024; Lan *et al.*, 2024] for mobility modeling. However, we exclude LLM-based approaches from this study, as their inherent tokenization mechanisms struggle with the high-precision continuous coordinates required for fine-grained trajectory recovery [Musleh and Mokbel, 2024].

Consequently, the prevailing discretization strategy inevitably discards fine-grained geometric information, such as precise velocity vectors and distances, resulting in recovered paths that may lack geometric smoothness. To address this limitation, our DiffVec operates directly on continuous *motion vectors*, thereby preserving the local geometric structure of movement without suffering from quantization errors.

2.2 Diffusion Models in Spatio-Temporal Domains

Diffusion Probabilistic Models (DDPM) [Ho *et al.*, 2020] have recently outperformed GANs [Goodfellow *et al.*, 2014] in generative tasks. In the general spatio-temporal domain, score-based frameworks such as **CSDI** [Tashiro *et al.*, 2021], **SSSD** [Alcaraz and Strodthoff, 2022], **Diff-TS** [Yuan and Qiao, 2024], and the recent **PriSTI** [Liu *et al.*, 2023] have demonstrated superior performance in time-series imputation. Specific to trajectories, methods such as **Diff-Traj** [Zhu *et al.*, 2023], **ControlTraj** [Zhu *et al.*, 2024], and the prototype-guided **ProDiff** [Bu *et al.*, 2025] typically adhere to the standard "Noise-to-Data" paradigm. These models are designed to reconstruct the entire trajectory structure from scratch by progressively denoising from a pure Gaussian noise distribution.

Nevertheless, this standard generative process entails a significant efficiency bottleneck. Even with general acceleration samplers like **DDIM** [Song *et al.*, 2021], reconstructing structure from pure randomness ignores the strong local linearity of movement data and necessitates hundreds of sampling steps. In contrast, our **ResTraj** paradigm reformulates

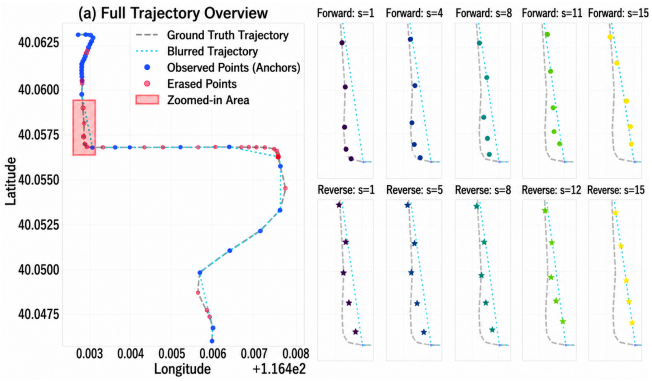

Figure 2: The forward and reverse propagation processes of ResTraj.

the task as *residual refinement* from an informative linear interpolation prior, which significantly accelerates convergence by focusing solely on learning the non-linear deviations.

3 Preliminaries

In this section, we formally define the trajectory data structures and formulate the recovery problem.

3.1 Problem Formulation

Trajectory. A trajectory is defined as a sequence of time-stamped coordinates $\mathbf{T} = \{p_1, p_2, ..., p_N\}$, where each point p_i at time t_i consists of a latitude and longitude coordinate, i.e., $p_i = (lat_i, lon_i, t_i)$. We assume the timestamps are strictly ordered such that $t_1 < t_2 < ... < t_N$.

Sparse Trajectory. A sparse trajectory, denoted as $\hat{\mathbf{T}}$, has the same sequence length N as the corresponding ground truth trajectory. We represent $\hat{\mathbf{T}}$ as $\{\tilde{p}_1, \tilde{p}_2, ..., \tilde{p}_N\}$, where $\tilde{p}_i$ is either the observed coordinate p_i or a masked value. To indicate the observation status, we introduce a binary mask vector $\mathbf{M} = \{m_1, m_2, ..., m_N\}$. The elements of the sparse trajectory are defined based on this mask:

$$m_i = \begin{cases} 1 & \text{if } p_i \text{ is observed in } \hat{\mathbf{T}} \\ 0 & \text{if } p_i \text{ is masked (missing)} \end{cases} \tag{1}$$

Problem Statement. Given a sparse trajectory $\hat{\mathbf{T}}$ and its corresponding mask $\mathbf{M}$, the objective is to recover the ground truth trajectory $\mathbf{T}$. Specifically, we aim to infer the missing locations p_i where $m_i = 0$ (i.e., $p_i \in \mathbf{T}$ and $p_i \notin \hat{\mathbf{T}}$), utilizing the observed points and the learned spatio-temporal patterns. We consider the free-space recovery setting, where no external road network topology is provided or assumed during the inference process.

3.2 ResTraj Diffusion Process

The objective of ResTraj is to enable trajectory recovery to depart from the conventional paradigm of denoising from pure noise. We first employ linear interpolation to construct a coarse "blurred trajectory" as a structural prior. As visualized in Figure 2, the forward process (top row) gradually perturbs the ground truth towards this interpolated prior, while the reverse process (bottom row) starts directly from this coarse initialization to iteratively recover the precise geometry. This formulation allows the model to focus on refining local details rather than reconstructing global structure from scratch.

Fill Sparse Trajectories

To provide an initial coarse imputation for our model, we first employ linear interpolation. This technique serves to fill missing segments in a trajectory by assuming uniform motion between known anchor points.

Formally, for each missing point $\mathbf{p}_i$ in the trajectory $\mathbf{T}$, where $m_i = 0$ indicates it is a missing point that needs to be imputed, the linear interpolation method estimates its coordinates $\hat{\mathbf{p}}_i$ at time t_i. We first identify its nearest preceding observed anchor point $\mathbf{p}_{\text{left}}$ and succeeding observed anchor point $\mathbf{p}_{\text{right}}$. These anchors are defined by their timestamps, t_{left} and t_{right}, as follows:

$$t_{\text{left}} = \max(\{t_k \mid t_k \leq t_i \text{ and } m_k = 1\}) \tag{2}$$

$$t_{\text{right}} = \min(\{t_k \mid t_k \geq t_i \text{ and } m_k = 1\}) \tag{3}$$

We then compute an interpolation ratio $\alpha_i \in [0, 1]$ based on the timestamp t_i's relative position between t_{left} and t_{right}. The calculation for α_i is given by:

$$\alpha_i = \begin{cases} 0 & \text{if } t_{\text{right}} = t_{\text{left}} \text{ (i.e., } t_i = t_{\text{left}}) \\ \frac{t_i - t_{\text{left}}}{t_{\text{right}} - t_{\text{left}}} & \text{otherwise} \end{cases} \tag{4}$$

The final imputed coordinate $\hat{\mathbf{p}}_i$ is then computed as a weighted average of the two anchors, as follows:

$$\hat{\mathbf{p}}_i = (1 - \alpha_i) \cdot \mathbf{p}_{\text{left}} + \alpha_i \cdot \mathbf{p}_{\text{right}} \tag{5}$$

This process is applied to all missing points. The resulting interpolated trajectory, which serves as a preliminary estimation, is referred to as the blurred trajectory ($\mathbf{B}_0$), formed by combining the observed points and the imputed points as follows:

$$\mathbf{B_0} = (\hat{\mathbf{p}}_0, \hat{\mathbf{p}}_1, \ldots, \hat{\mathbf{p}}_{N-1}) \tag{6}$$

$$\text{where } \hat{\mathbf{p}}_i = \begin{cases} \mathbf{p}_i & \text{if } m_i = 1, \\ \hat{\mathbf{p}}_i \quad \text{(calculated by (5))} & \text{if } m_i = 0. \end{cases} \tag{7}$$

Forward Process

In the forward diffusion process, we progressively add noise to the ground truth trajectory $\mathbf{T}_0$ conditioned on the established blurred trajectory $\mathbf{B}_0$. Unlike conventional diffusion models that add noise to every point, our approach focuses solely on the masked (missing) regions. Inspired by the work of ResShift, we design the following transition distribution of the forward process that focuses solely on the masked sites:

$$q(\mathbf{T}_s \mid \mathbf{T}_{s-1}, \mathbf{B}_0, \mathbf{M}) = \mathcal{N}(\mathbf{T}_s; \mu_s, \text{diag}(\sigma_s)) \tag{8}$$

To distinguish the timestamp t in the trajectory sequence, we use $s = 1, 2, \cdots, S$ to denote the time step in the diffusion process.

$$\mu_s = (\alpha_s \mathbf{E}_0 + \mathbf{T}_{s-1}) \odot (1 - \mathbf{M}) + (\eta_s \mathbf{E}_0 + \mathbf{T}_0) \odot \mathbf{M} \tag{9}$$

$$\sigma_s = \kappa^2 \alpha_s (1 - \mathbf{M}) \tag{10}$$

Here, $\mathbf{E}_0 = \mathbf{B}_0 - \mathbf{T}_0$ represents the initial offset. The noise schedule is controlled by coefficients $\{\eta_s\}_{s=1}^{S}$. We adopt

the exponential schedule from [Yue *et al.*, 2023], defined as $\sqrt{\eta_s} = \sqrt{\eta_1} \cdot b^{\beta_s}$, where b and β_s are determined by a curvature hyperparameter p:

$$\beta_s = (S-1)\left(\frac{s-1}{S-1}\right)^p, \quad b = \exp\left[\frac{1}{2(S-1)}\log\frac{\eta_S}{\eta_1}\right] \tag{11}$$

In this work, we use a **fixed** p for all samples. We define $\alpha_s = \eta_s - \eta_{s-1}$ as the accumulated offset proportion. κ is a scale hyperparameter. After transformation, we obtain the conditional distribution:

$$q(\mathbf{T}_s \mid \mathbf{T}_0, \mathbf{B}_0, \mathbf{M}) = \mathcal{N}\big(\mathbf{T}_s; \eta_s \mathbf{E}_0 + \mathbf{T}_0, \operatorname{diag}(\kappa^2 \eta_s (1 - \mathbf{M}))\big) \tag{12}$$

Reverse Process

During the reverse process, the model progressively recovers the ground truth trajectory by iteratively approximating the posterior distribution:

$$p_\theta(\mathbf{T}_0 \mid \mathbf{B}_0, \mathbf{M}) = \int p(\mathbf{T}_S \mid \mathbf{B}_0, \mathbf{M}) \prod_{s=1}^{S} p_\theta(\mathbf{T}_{s-1} \mid \mathbf{T}_s, \mathbf{B}_0, \mathbf{M}) \, \mathrm{d}\mathbf{T}_{1:S} \tag{13}$$

where $p(\mathbf{T}_S \mid \mathbf{B}_0, \mathbf{M}) \approx \mathcal{N}(\mathbf{T}_S; \mathbf{B}_0, diag(\kappa^2 \mathbf{M}))$ and $p_\theta(\mathbf{T}_{s-1} | \mathbf{T}_s, \mathbf{B}_0, \mathbf{M})$ is the reverse transition probability. After reparameterization sampling, we obtain the following formula:

$$p_\theta(\mathbf{T}_{s-1} | \mathbf{T}_s, \mathbf{B}_0, \mathbf{M}) = \mathcal{N}\left(\mathbf{T}_{s-1}; \hat{\mu}, diag(\hat{\sigma})\right) \tag{14}$$

$$\hat{\mu} = \frac{\eta_{s-1}}{\eta_s}(1 - \mathbf{M}) \odot \mathbf{T}_s + \frac{\alpha_s}{\eta_s}(1 - \mathbf{M}) \odot f_\theta\left(\mathbf{T}_s, \mathbf{B}_0, s\right) \tag{15}$$

$$\hat{\sigma} = \kappa^2 \frac{\eta_{s-1}}{\eta_s} \alpha_s (1 - \mathbf{M}) \tag{16}$$

Here, f_θ is a neural network parameterized by θ, which aims to predict the ground truth trajectory $\mathbf{T}_0$ from $\mathbf{T}_s$. In the reverse denoising process, we specifically focus on denoising the masked regions, ensuring that the observed parts of the trajectory remain unchanged to preserve ground-truth information. To achieve precise coordinate estimation, we employ the Mean Absolute Error (MAE) as our optimization objective:

$$\min_\theta \mathcal{L}_\theta = \mathbb{E}_{\mathbf{B}_0, \mathbf{T}_0, s} \| f_\theta(\mathbf{T}_s, \mathbf{B}_0, s) - \mathbf{T}_0 \| \tag{17}$$

3.3 MVformer Architecture

To parameterize the denoising network f_θ within our ResTraj diffusion process, we design a Transformer-based architecture named **MVformer** (Motion Vector Transformer), as illustrated in Figure 3. The core principle of MVformer is to model the local motion patterns via **motion vectors** rather than absolute coordinates.

Motion Vector Representation

Standard trajectory recovery models typically take absolute coordinates (e.g., latitude and longitude) as input. However, absolute coordinates suffer from arbitrary origins and non-stationary distributions, making it difficult for neural networks to capture translation-invariant patterns. To address this, we transform the input coordinate trajectory $\mathbf{T}_s \in \mathbb{R}^{N\times 2}$ into a sequence of motion vectors $\mathbf{V}_s \in \mathbb{R}^{N\times 2}$.

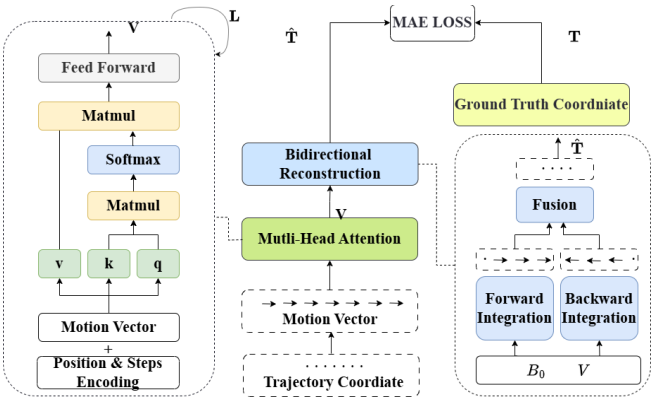

Figure 3: Overall structure diagram of MVformer

Formally, for a trajectory sequence at diffusion step s, the motion vector $\mathbf{v}_{s,i}$ for the i-th point is defined as the spatial displacement relative to its predecessor:

$$\mathbf{v}_{s,i} = \begin{cases} \mathbf{p}_{s,1} & \text{if } i = 1 \\ \mathbf{p}_{s,i} - \mathbf{p}_{s,i-1} & \text{if } i > 1 \end{cases} \tag{18}$$

This differential representation explicitly encodes the velocity and heading of the moving object. Unlike coordinates that vary significantly across different spatial regions, motion vectors follow a stable, zero-centered distribution (as visualized in the Figure 1), enabling the model to focus on learning the local geometric structure and sequential dependencies of human mobility.

Embedding and Encoder Backbone

The vector sequence $\mathbf{V}_s$ is first projected into a high-dimensional hidden space $\mathbb{R}^{N\times D_{\text{model}}}$ via a linear layer. To inform the model of the current noise level, we inject the diffusion step s using a sinusoidal time embedding $\mathbf{e}_s$:

$$\mathbf{H}_0 = \text{Linear}(\mathbf{V}_s) + \mathbf{e}_s \tag{19}$$

To capture the sequential order without disrupting the translation invariance of motion vectors, we employ **Rotary Positional Encoding (RoPE)**. RoPE encodes relative positions by rotating the query and key vectors in the attention mechanism, which is mathematically well-suited for vector-based sequence modeling.

The backbone consists of L stacked Transformer encoder blocks. Each block comprises a Multi-Head Self-Attention (MSA) layer and a Feed-Forward Network (FFN), wrapped with residual connections and Layer Normalization:

$$\mathbf{H}'_l = \text{LayerNorm}(\mathbf{H}_{l-1} + \text{MSA}(\mathbf{H}_{l-1})) \tag{20}$$

$$\mathbf{H}_l = \text{LayerNorm}(\mathbf{H}'_l + \text{FFN}(\mathbf{H}'_l)) \tag{21}$$

Here, the MSA mechanism allows each motion vector to attend to all other vectors in the sequence, capturing long-range spatio-temporal correlations essential for recovering missing segments.

Bidirectional Reconstruction

The MVformer outputs a predicted sequence of motion vectors $\hat{\mathbf{V}}_0 = \{\hat{\mathbf{v}}_{0,i}\}_{i=1}^{N}$. To reconstruct the spatial trajectory $\hat{\mathbf{T}}_0$, we propose a bidirectional scheme that leverages observed anchors as hard constraints.

1) Forward Integration: We derive the forward candidate trajectory $\hat{\mathbf{T}}_{\text{fwd}} = \{\hat{\mathbf{p}}_{\text{fwd},i}\}_{i=1}^{N}$. If the first point p_1 is missing ($m_1 = 0$), we use $\hat{\mathbf{v}}_{0,1}$ as the direct coordinate prediction. For subsequent points, we accumulate motion vectors. Observed anchors ($m_i = 1$) reset the accumulation to prevent error propagation:

$$\hat{\mathbf{p}}_{\text{fwd},i} = \begin{cases} \mathbf{p}_i & \text{if } m_i = 1 \\ \hat{\mathbf{v}}_{0,1} & \text{if } m_i = 0, i = 1 \\ \hat{\mathbf{p}}_{\text{fwd},i-1} + \hat{\mathbf{v}}_{0,i} & \text{if } m_i = 0, i > 1 \end{cases} \tag{22}$$

2) Backward Integration: We generate the backward trajectory $\hat{\mathbf{T}}_{\text{bwd}}$ by integrating from the end ($i = N \dots 1$). If the last point is missing, we adopt the forward estimate $\hat{\mathbf{p}}_{\text{fwd},N}$ to initialize the backward pass:

$$\hat{\mathbf{p}}_{\text{bwd},i} = \begin{cases} \mathbf{p}_i & \text{if } m_i = 1 \\ \hat{\mathbf{p}}_{\text{fwd},i} & \text{if } m_i = 0, i = N \\ \hat{\mathbf{p}}_{\text{bwd},i+1} - \hat{\mathbf{v}}_{0,i+1} & \text{if } m_i = 0, i < N \end{cases} \tag{23}$$

3) Anchor-Constrained Fusion: Let $\tau = \min\{k \mid m_k = 1\}$ be the first observed anchor. For the initial missing segment ($i < \tau$), forward integration relies on a predicted start point and may be unstable; we therefore use only the backward pass. For all other missing points, we average both passes to mitigate errors:

$$\hat{\mathbf{p}}_{0,i} = \begin{cases} \mathbf{p}_i & \text{if } m_i = 1 \\ \hat{\mathbf{p}}_{\text{bwd},i} & \text{if } i < \tau \\ (\hat{\mathbf{p}}_{\text{fwd},i} + \hat{\mathbf{p}}_{\text{bwd},i})/2 & \text{otherwise} \end{cases} \tag{24}$$

Optimization Objective

Recall that in Eq. (17), we defined the general diffusion training objective. In our framework, the denoising function $f_\theta(\mathbf{T}_s, \mathbf{B}_0, s)$ is parameterized by the MVformer backbone followed by the bidirectional reconstruction module.

Let $\hat{\mathbf{T}}_0$ denote the final reconstructed coordinates output by this process. We instantiate the loss function to specifically minimize the Mean Absolute Error (MAE) on the missing regions. The final optimization objective is formulated as:

$$\mathcal{L}_\theta = \mathbb{E}_{\mathbf{B}_0, \mathbf{T}_0, s}\left[\|(\hat{\mathbf{T}}_0 - \mathbf{T}_0) \odot (1 - \mathbf{M})\|_1\right] \tag{25}$$

By calculating the loss exclusively on the unobserved segments (where $m_i = 0$), we enforce the model to learn the intrinsic geometric correlations required to infer the missing trajectory from the observed anchors.

4 Experiments

4.1 Datasets

We evaluate our model on two real-world trajectory datasets. The **Geolife** dataset [Zheng *et al.*, 2010], collected by Microsoft Research Asia, records the GPS trajectories of 182 users over a five-year period in Beijing. The data is sampled at **5-second intervals**, covering diverse transportation modes such as walking, driving, and cycling. The **Porto** dataset [Moreira *et al.*, 2013] consists of approximately 1.7 million taxi trajectories from the city of Porto, Portugal, sampled at a fixed **15-second interval**, representing a high-density, vehicle-only scenario. Following standard preprocessing [Si *et al.*, 2023], we segment trajectories into fixed-length sequences ($N = 64$) and filter out static or short segments to ensure data quality. The basic statistics are summarized in Table 1.

Dataset	#Duration	#Trajectory	#Average Distance
Geolife	5 years	17,621	73.37 km
Porto	1 year	1,710,670	3.26 km

Table 1: Dataset statistics.

4.2 Experimental Settings

Task Setup. To simulate sparse trajectories, we employ a normal distribution-based masking strategy to remove continuous sub-sequences of points in each trajectory. The missing rate is set to 50% during training. We partition the datasets into training, validation, and test sets with a ratio of 7:2:1.

Implementation Details. Our model is implemented in PyTorch. The **MVformer** backbone consists of $L = 4$ layers with a hidden dimension of $d = 128$ and 4 attention heads. For the **ResTraj** process, we set the total diffusion steps $S = 15$ and the noise scale $kappa = 0.001$. The noise schedule follows a fixed exponential curve ($p = 0.3$) as recommended in [Yue *et al.*, 2023]. We use the AdamW optimizer with a learning rate of $1e^{-4}$ and a batch size of 2048. Training runs for 200 epochs with early stopping.

Evaluation Metrics. To evaluate performance from both microscopic (point-level) and macroscopic (trajectory-level) perspectives, we employ three complementary metrics: **Mean Absolute Error (MAE)**, which calculates the average L2 distance between recovered and ground-truth coordinates; **Mean Squared Error (MSE)**, which computes the average squared L2 distance to strictly penalize large outliers; and **Normalized Dynamic Time Warping (NDTW)**, which assesses the shape similarity and temporal alignment between the recovered and ground-truth trajectories.

Baselines

To demonstrate the effectiveness of our proposed DiffVec framework, we compare it against a comprehensive set of baseline methods spanning attention-based and diffusion-based categories. For attention and Transformer-based approaches, we evaluate **AttnMove** [Xia *et al.*, 2021] (specifically its AttnMove-H variant), which employs an intra-trajectory attention mechanism to dynamically weigh observed anchor points for inferring missing locations. We also include **TrajBERT** [Si *et al.*, 2023], a BERT-based model that leverages bidirectional context and a Masked Trajectory Model objective to learn coordinate representations. Additionally, we adapt **SAITS** [Du *et al.*, 2023], a state-of-the-art imputation method utilizing Diagonally-Masked Self-

Method	Geolife Dataset									Porto Dataset								
	Missing 10%			Missing 30%			Missing 50%			Missing 10%			Missing 30%			Missing 50%		
	MAE	MSE	NDTW	MAE	MSE	NDTW	MAE	MSE	NDTW	MAE	MSE	NDTW	MAE	MSE	NDTW	MAE	MSE	NDTW
AttnMove	4.25	110.5	4.25	6.45	149.2	6.85	8.25	182.3	8.55	7.85	3.25	5.15	10.55	4.85	6.85	14.50	6.55	9.25
TrajBERT	3.85	82.5	3.85	5.92	105.4	5.65	7.15	132.6	7.25	6.85	1.15	4.20	8.95	1.65	5.95	12.85	2.25	7.85
SAITS	5.55	115.2	4.85	7.55	162.5	7.15	9.25	205.8	9.55	9.55	4.55	5.85	12.25	6.25	7.85	16.85	8.55	10.25
Diff-TS	4.55	135.5	4.15	6.85	165.2	6.25	8.55	198.5	8.15	7.55	2.55	4.95	9.85	3.95	6.65	13.95	5.65	8.95
ControlTraj	3.45	75.5	3.25	5.65	95.8	4.85	6.85	125.6	6.15	6.15	0.85	3.85	8.25	1.25	5.55	11.95	1.85	7.15
ProDiff	2.92	28.4	1.25	5.35	56.8	2.45	6.15	68.2	3.85	5.25	0.45	3.45	7.15	0.65	4.95	11.45	1.55	5.85
DiffVec	**2.50**	**21.5**	**1.00**	**4.80**	**47.1**	**2.15**	**5.52**	**55.5**	**3.45**	**4.55**	**0.25**	**3.12**	**6.56**	**0.45**	**4.45**	**10.81**	**1.25**	**5.13**

Table 2: Performance comparison on Geolife and Porto datasets (Missing 10%, 30%, and 50%). Results are scaled by 10^{-3}. Best results are highlighted in bold, and second-best are underlined.

Attention (DMSA), to capture temporal dependencies for spatial inference. In the realm of generative diffusion models, we compare against **Diff-TS** [Yuan and Qiao, 2024], a probabilistic framework utilizing a Transformer encoder-decoder adapted for continuous coordinate reconstruction; **ControlTraj** [Zhu *et al.*, 2024], a conditional model featuring a U-Net backbone with geographic attention, adapted to treat observed anchors as topological constraints; and **ProDiff** [Bu *et al.*, 2025], a prototype-guided diffusion model that integrates macro-level pattern learning, using observed anchors as conditional inputs for fine-grained reconstruction.

4.3 Performance Comparison

Table 2 presents the quantitative comparison on the Geolife and Porto datasets. We analyze the performance differences based on the modeling mechanisms of the baselines compared to our proposed DiffVec.

Comparison with Traditional Baselines. Discrete-adapted models, such as AttnMove and TrajBERT, effectively capture basic spatio-temporal dependencies; however, their design relies on discretizing continuous locations into grid tokens, which inherently limits their precision in recovering fine-grained GPS coordinates. Conversely, while SAITS is specifically designed for continuous time-series imputation, it treats trajectories as generic numerical sequences. It fails to explicitly model the complex spatial correlations and geographic constraints unique to mobility data, resulting in sub-optimal performance compared to spatially-aware methods.

Comparison among Diffusion Baselines. While diffusion models generally surpass traditional baselines, existing methods face distinct limitations. ControlTraj, despite representing the state-of-the-art in generation, lacks conditional constraints for partial observations, rendering it inferior to the prototype-guided ProDiff in recovery tasks. Furthermore, standard diffusion paradigms are typically hindered by the inefficiency of denoising from pure noise and the neglect of local motion patterns, often yielding results that are plausible but geometrically inconsistent.

DiffVec employs **ResTraj** to reformulate the task as residual refinement from an interpolated prior, effectively overcoming the instability of pure-noise generation. Complementarily, by explicitly modeling **local motion patterns**, DiffVec captures trajectory flow patterns to facilitate more precise recovery.

4.4 Ablation Study

To rigorously assess the contribution of each component within the DiffVec framework, we conducted an ablation study by comparing the full model (**Ours**) against three specific variants: (1) **-ddpm**: Replaces our ResTraj paradigm with a standard DDPM that denoises from isotropic Gaussian noise; (2) **-bi_restruct**: Removes the *Bidirectional Reconstruction* pass from the MVformer, relying solely on forward motion accumulation; and (3) **-vec**: Removes the motion vector formulation, operating directly on absolute coordinates. The convergence dynamics of MAE, MSE, and NDTW are visualized in Figure 4.

Efficacy of Residual Diffusion (ResTraj). As observed in Figure 4, the **-ddpm** baseline (orange curve) exhibits an inefficient convergence rate and significantly higher final error compared to our method. Standard diffusion models struggle to reconstruct the complex spatio-temporal structure of trajectories from pure noise within limited sampling steps. In contrast, by conditioning on an interpolated prior, our **ResTraj** strategy reformulates the generative task as residual refinement. This significantly simplifies the optimization landscape, leading to faster convergence and superior stability.

Impact of Bidirectional Reconstruction. The **-bi_restruct** variant (green curve), which utilizes only forward integration, consistently underperforms the full model. This performance degradation is attributed to *error propagation* inherent in sequential motion accumulation. Without the constraints provided by the backward pass, minor deviations in early steps amplify over time. The full model effectively mitigates this by fusing forward and backward estimates, utilizing future anchors to calibrate the trajectory and ensure endpoint consistency.

Necessity of Motion Vector Modeling. The **-vec** variant (red line) exhibits a slow and linear descent, contrasting with the sharp convergence of vector-equipped models. This demonstrates that integrating motion vectors effectively captures the local motion patterns of trajectories, thereby significantly accelerating model convergence and facilitating the efficient recovery of geometric structures. The persistent gap in NDTW further confirms the superiority of this design in maintaining structural coherence.

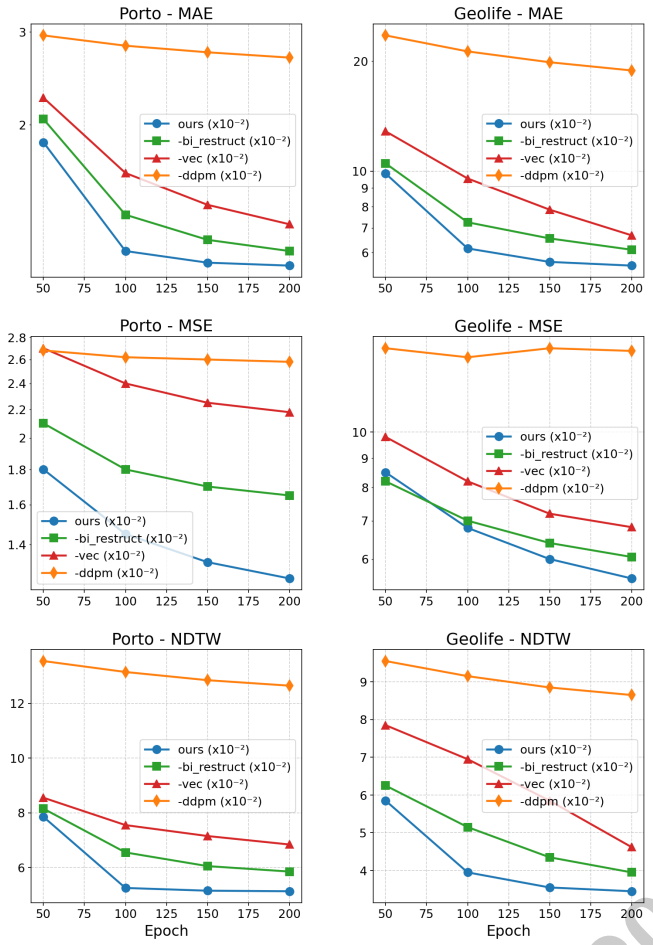

Figure 4: Ablation study demonstrating the convergence behavior of different model variants on the Porto and Geolife datasets.

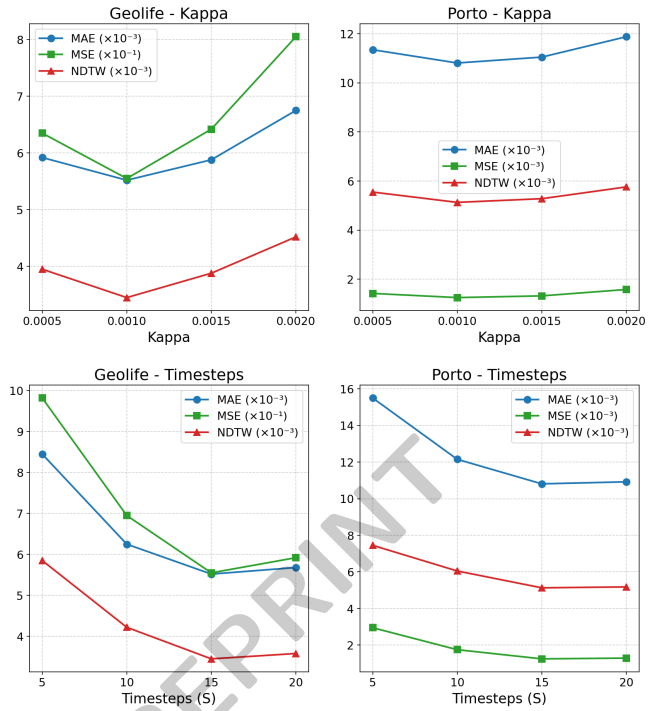

Figure 5: Sensitivity analysis of $kappa$ and S. Metrics are scaled for visualization.

Method	Params(M)	FLOPs(G)	Steps	Latency
ControlTraj	8.20	0.1781	300	∼2859ms
Diffusion-TS	2.30	0.1200	300	∼7620ms
ProDiff	23.21	1.6100	300	∼4815ms
Ours	**0.86**	**0.0517**	**15**	**∼57ms**

Table 3: Efficiency comparison with diffusion-based baselines.

4.5 Parameter Sensitivity and Efficiency Analysis

We investigate the impact of noise schedule κ and diffusion steps S to determine the optimal configuration. The results are shown in Figure 5.

Impact of Noise Schedule ($kappa$). Both datasets achieve the global optimum at $\kappa = 0.001$ but exhibit distinct sensitivity profiles. Geolife displays a pronounced convex pattern, where performance degrades significantly when $\kappa > 0.0015$, indicating low tolerance for structural disruption. In contrast, Porto demonstrates greater robustness with flatter curvature, suggesting that dense road constraints mitigate high noise levels. We select $\kappa = 0.001$ as the balanced optimum.

Impact of Diffusion Steps (S). DiffVec converges rapidly due to the ResTraj paradigm. For Geolife, the error decreases sharply to an inflection point at $S = 15$ before a slight rebound at $S = 20$, implying a risk of over-refinement. Porto shows more gradual improvement that saturates after $S = 15$. Consequently, we fix $S = 15$ to maximize efficiency without compromising accuracy.

Efficiency Comparison. We further compare DiffVec against diffusion-based baselines in Table 3. Thanks to the ResTraj paradigm, DiffVec requires only 15 denoising steps—20× fewer than the 300 steps used by others. This leads to substantially lower latency (∼57 ms vs. ∼2859–7620 ms) and the smallest model footprint (0.86M parameters), demonstrating clear computational advantages.

5 Conclusion

In this paper, we proposed **DiffVec**, a diffusion-based framework for recovering fine-grained continuous trajectories from sparse observations in free space. To address inefficiency, we designed the **ResTraj** paradigm, which reformulates generation as residual refinement from a structural prior, significantly accelerating convergence. To overcome coordinate instability and non-stationary distributions, we introduced **MVformer** to model local motion patterns via motion vectors, capturing instantaneous velocities and directional changes. A bidirectional reconstruction mechanism with anchor-constrained fusion ensures endpoint consistency. Experiments on Geolife and Porto datasets demonstrate DiffVec's superiority over state-of-the-art baselines. Future work includes incorporating road network constraints and exploring lightweight model distillation for edge applications.

Acknowledgments

This work is supported in part by Tianshan Talent Training Program (2023TSYCLJ0023), Xinjiang Uygur Autonomous Region Universities Fundamental Research Funds Scientific Research Project (XJEDU2022P018), Excellent Graduate Innovation Project of Xinjiang University (XJDX2025YJS089).

References

[Alcaraz and Strodthoff, 2022] Juan Miguel Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. In *arXiv*, 2022.

[Bu *et al.*, 2025] Tianci Bu, Le Zhou, Wenchuan Yang, Jianhong Mou, Kang Yang, Suoyi Tan, Feng Yao, Jingyuan Wang, and Xin Lu. Prodiff: Prototype-guided diffusion for minimal information trajectory imputation. *arXiv*, 2025.

[Cao *et al.*, 2018] Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. Brits: Bidirectional recurrent imputation for time series. volume 31, 2018.

[Chen *et al.*, 2023] Yuqi Chen, Hanyuan Zhang, Weiwei Sun, and Baihua Zheng. Rntrajrec: Road network enhanced trajectory recovery with spatial-temporal transformer. pages 829–842, 2023.

[Du *et al.*, 2023] Wenjie Du, David Côté, and Yan Liu. Saits: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219:119619, 2023.

[Duan *et al.*, 2024] Jiaqi Duan, Xiangfu Meng, and Guihong Liu. Where to go at the next timestamp. *Data Science and Engineering*, 9(1):88–101, 2024.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, volume 27, pages 2672–2680, 2014.

[Ho *et al.*, 2020] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. arXiv, 2020.

[Lan *et al.*, 2024] Zhengxing Lan, Lingshan Liu, Bo Fan, Yisheng Lv, Yilong Ren, and Zhiyong Cui. Traj-llm: A new exploration for empowering trajectory prediction with pre-trained large language models. *IEEE Transactions on Intelligent Vehicles*, 2024.

[Li *et al.*, 2000] Xin Li, Guodong Cheng, and Ling Lu. Comparison of spatial interpolation methods. volume 15, pages 260–265, 2000.

[Liu *et al.*, 2023] Mingzhe Liu, Han Huang, Hao Feng, Leilei Sun, Bowen Du, and Yanjie Fu. PriSTI: A conditional diffusion framework for spatiotemporal imputation. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 1927–1939. IEEE, 2023.

[Moreira *et al.*, 2013] João Moreira, Ricardo Gaetano, Mário Viana, Carlos Rocha, and Pedro Furtado. Predicting the next location: a recurrent model with spatial and temporal contexts. In *ECML PKDD*, pages 532–547. Springer, 2013.

[Musleh and Mokbel, 2023] Mashaal Musleh and Mohamed F Mokbel. Kamel: A scalable bert-based system for trajectory imputation. *Proceedings of the VLDB Endowment*, 17(3), 2023.

[Musleh and Mokbel, 2024] Mashaal Musleh and Mohamed F Mokbel. Let's speak trajectories: A vision to use nlp models for trajectory analysis tasks. *ACM Transactions on Spatial Algorithms and Systems*, 10(2):1–25, 2024.

[Si *et al.*, 2023] Junjun Si, Jin Yang, Yang Xiang, Hanqiu Wang, Li Li, Rongqing Zhang, Bo Tu, and Xiangqun Chen. Trajbert: Bert-based trajectory recovery with spatial-temporal refinement for implicit sparse trajectories. *IEEE Transactions on Mobile Computing*, 23(5):4849–4860, 2023.

[Song *et al.*, 2021] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[Tashiro *et al.*, 2021] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional score-based diffusion models for probabilistic time series imputation. arXiv, 2021.

[Wang *et al.*, 2024] Xinyu Wang, Meng Chen, Rho Liu, Niexiao Liu, Yixu Shen, Yuhong We, and Wen-Chih Peng. Llm-mob: Large language models for human mobility prediction. In *Proceedings of the 32nd ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL '24)*, pages 1–10. ACM, 2024.

[Xia *et al.*, 2021] Tong Xia, Yunhan Qi, Jie Feng, Fengli Xu, Funing Sun, Diansheng Guo, and Yong Li. Attnmove: History enhanced trajectory recovery via attentional network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4494–4502, 2021.

[Yuan and Qiao, 2024] Xinyu Yuan and Yan Qiao. Diffusion-ts: Interpretable diffusion for general time series generation. *arXiv*, 2024.

[Yue *et al.*, 2023] Zongsheng Yue, Jianyi Wang, and Chen Change Loy. ResShift: Efficient diffusion model for image super-resolution by residual shifting. arXiv, 2023.

[Zheng *et al.*, 2010] Yu Zheng, Xing Chen, Xing Xie, and Wei-Ying Ma. Geolife: a collaborative social networking service among a user group in geo-life. In *Proceedings of the 1st ACM SIGSPATIAL international workshop on location based social networks*, pages 31–40, 2010.

[Zhu *et al.*, 2023] Yuanshao Zhu, Yongchao Ye, Shiyao Zhang, Xiangyu Zhao, and James J. Q. Yu. DiffTraj: Generating GPS trajectory with diffusion probabilistic model. arXiv, 2023.

[Zhu *et al.*, 2024] Yuanshao Zhu, James Jianqiao Yu, Xiangyu Zhao, Qidong Liu, Yongchao Ye, Wei Chen, Zijian Zhang, Xuetao Wei, and Yuxuan Liang. Controltraj: Controllable trajectory generation with topology-constrained diffusion model. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4676–4687, 2024.