XTraj: A Coarse-to-Fine Autoregressive Framework for Transferable Trajectory Generation

Chao Wang[1,*], **Zhiquan Lai**[1,*], **Xinwei Fang**[3],
Yuanshao Zhu[2,†], **Xun Zhou**[1], **James Jianqiao Yu**[1,†]
[1]School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China
[2]City University of Hong Kong, Hong Kong SAR, China
[3]Department of Computer Science, University of York, York, United Kingdom
chaow@stu.hit.edu.cn, zhiquanlai@stu.hit.edu.cn, yuanshao@ieee.org
zhouxun2023@hit.edu.cn, jqyu@ieee.org

Abstract

Generating realistic human trajectories is essential for mobility simulation but remains challenging when they must generalize across cities and respect road network constraints. In practice, existing methods often suffer from three key limitations: geographic overfitting, length-rigid generation, and off-road drift when generating GPS coordinates directly. To address these issues, we propose **XTraj**, a transferable coarse-to-fine autoregressive framework for road-consistent trajectory generation. Specifically, XTraj first learns transferable road-segment representations by integrating road geometry, POI context, historical traffic intensity, and graph topology. Based on these representations, it then autoregressively generates variable-length road-segment routes under road-connectivity constraints. Finally, instead of directly generating coordinates, XTraj refines each generated route in a route-progress space by predicting monotonic progress increments along valid road geometry, from which GPS points are recovered by interpolation. This route-aligned design naturally supports variable-length GPS generation and guarantees road-geometry consistency by construction. Experiments on two real-world trajectory datasets show that XTraj improves spatial fidelity over competitive baselines, transfers effectively to unseen cities in a training-free setting, and ensures road-geometry consistency by construction without any post-hoc map matching.

1 Introduction

Trajectory data provide fine-grained evidence of route choice, travel demand, and spatial interaction across road networks, supporting applications such as traffic planning, travel-time estimation, and location-based services [36, 35, 29, 26]. However, large-scale trajectory records remain difficult to access because of privacy concerns, collection costs, and proprietary restrictions [23, 12, 2, 27, 14]. These barriers limit the availability of data that capture real travel behavior. In this context, trajectory generation offers a practical alternative by synthesizing realistic mobility records [2, 27]. Such records can support model development and evaluation without directly releasing individual travel histories, helping extend the use of mobility data when access to real records is restricted.

Recent advances in generative modeling, particularly diffusion-based approaches, have substantially improved trajectory synthesis by learning complex spatiotemporal distributions [39, 40, 9]. Nevertheless, three obstacles still limit their practical deployment. (i) Geographic overfitting: many methods learn trajectory distributions in a city-specific space [5]. This dependency on the training city's spatial

[*]Equal contribution
[†]Corresponding authors

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

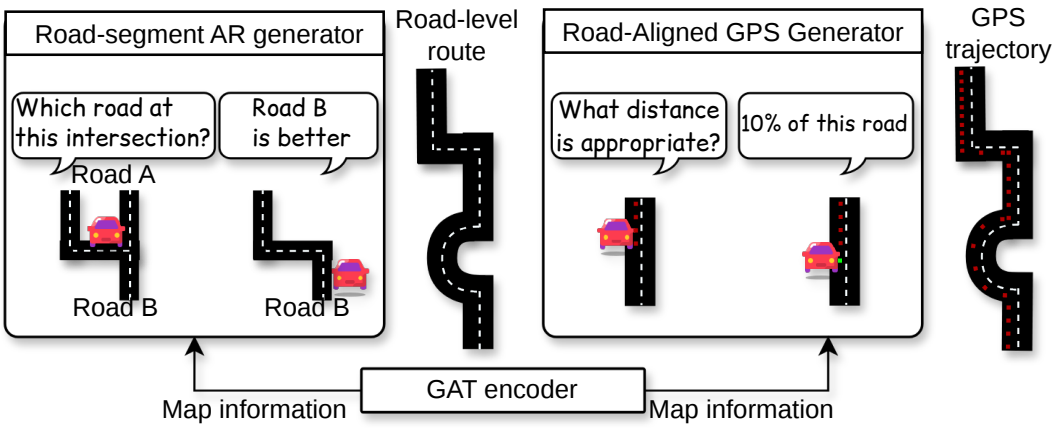


Figure 1: the overall workflow of XTraj. A pretrained GAT encodes road segments, the Road-Segment-Level Autoregressive Generator produces a road-segment route, and the Road-Aligned GPS Generator decodes that route into a GPS trajectory.

extent limits their reuse on unfamiliar networks. (ii) Length-rigid generation: many methods resample trajectories to a fixed number of points before training [39, 40]. This preprocessing obscures natural variation in sequence length and can distort temporal information associated with trip and sampling density. (iii) Off-road drift: GPS trajectory generators can produce points outside valid roads when generating in an unconstrained continuous space. These deviations require post-hoc correction, which can introduce additional errors [3]. Together, these limitations leave a gap in trajectory generation that jointly supports cross-city transfer, variable-length output, and road-geometry consistency.

These barriers motivate three requirements for practical trajectory generation. **Transferability** requires representations and generation rules that can operate on a new road network without additional parameter training. **Length adaptability** requires support for variable-length sequences rather than imposing a fixed number of observations on every trip. **Road consistency** requires both legally connected road-segment sequences and GPS points that lie on the corresponding road geometry. Together, these requirements call for realistic, variable-length trajectories that remain topologically and geometrically valid when generated in new cities without retraining.

To meet these requirements, we introduce **XTraj**, a **cross**-city transferable coarse-to-fine autoregressive framework that decomposes **traj**ectory generation into three components: road representation learning, road-segment route generation, and GPS trajectory refinement. Firstly, the road-segment encoder addresses geographic overfitting by learning transferable road representations through masked trajectory pretraining with general road attributes and local graph topology. Secondly, the Road-Segment-Level Autoregressive Generator addresses length-rigid generation by generating road-segment routes under topology and destination-directed constraints. Finally, given a generated road-segment route, the Road-Aligned GPS Generator eliminates off-road drift through variable-length progress prediction and coordinate interpolation according to the predicted progress. Together, XTraj achieves transferable, length-adaptive, and road-consistent generation across different urban networks.

In summary, our work makes the following key contributions:

- We propose XTraj, a coarse-to-fine framework for autoregressive trajectory generation. It first learns transferable road-segment representations, then uses these representations to generate connected routes under road-network constraints, and finally refines each route into GPS coordinates.
- We introduce a route-aligned GPS generation formulation that predicts monotonic progress along generated routes and recovers coordinates through interpolation, supporting variable-length generation while preserving road-geometry consistency.
- We evaluate XTraj on real-world vehicle trajectory datasets from Xi'an and Chengdu. Quantitative and qualitative results demonstrate improved spatial fidelity, effective training-free cross-city transfer, and road-consistent generation without post-hoc map matching.

2 Related Work

Trajectory generation. Trajectory generation aims to synthesize realistic mobility traces while reducing privacy, regulatory, and data-collection barriers [38, 16, 1]. Existing methods fall into two broad categories. Knowledge-driven models, including gravity, intervening-opportunity, and exploration–preferential-return models, rely on behavioral assumptions and offer interpretability, but typically operate at coarse granularity and do not capture fine-grained route geometry [42, 24, 21, 22]. Data-driven methods instead learn mobility distributions from historical trajectories. Early approaches build on autoregressive sequence models, VAEs, GANs, and imitation-learning frameworks, exemplified by SeqGAN [33], TrajVAE [5], MoveSim [8], TS-TrajGen [11], DP-TrajGAN [34], and TrajGAIL [6]. More recent diffusion-based methods substantially improve spatial fidelity by denoising in continuous coordinate space, including DiffTraj [39], ControlTraj [40], CarDiff [9], and Diff-RNTraj [30]. Among these, ControlTraj and Diff-RNTraj further incorporate road-network conditioning to improve geometric validity. However, ControlTraj assumes that the road-segment route is provided as input, sidestepping the route-generation problem itself, while Diff-RNTraj remains tied to city-specific road structures and still operates within a fixed-length diffusion formulation. Despite their progress, diffusion-based methods generate coordinates in continuous space and therefore cannot guarantee road-geometry consistency by construction; their reliance on fixed-length denoising further limits variable-length generation.

Transferable trajectory modeling. Beyond generation, a complementary line of work studies how to transfer trajectory knowledge across cities, tasks, and data modalities when target-city data are sparse [20, 28, 32, 13]. Transferability is challenging because mobility patterns depend jointly on road topology, spatial scale, POI distribution, traffic demand, and sampling frequency. One major direction learns general trajectory representations through self-supervised, contrastive, or masked objectives, such as t2vec [17], CTLE [18], TrajCL [4], START [10], and RED [37]. These methods improve downstream transfer for tasks such as classification, similarity search, and trajectory recovery, but they are not designed for trajectory generation. A more recent line investigates cross-city or general-purpose trajectory models that integrate road-network structure or foundation-model-style pretraining, including TransferTraj [31], TrajMamba [19], and UniTraj [41]. While these works demonstrate the value of transferable trajectory representations, they primarily target representation learning, recovery, or prediction rather than fine-grained synthetic generation. XTraj bridges these two directions by learning transferable road-segment representations from general road attributes and graph topology, and then directly using them to generate variable-length, road-consistent trajectories without retraining on target cities.

3 Preliminaries

Road network and features. We represent a city road network as a directed graph $\mathcal{G} = (\mathcal{V},\ \mathcal{E})$, where each node $v \in \mathcal{V}$ corresponds to a road segment and each edge $(u,\ v) \in \mathcal{E}$ denotes a legal transition between connected segments. Each segment v is associated with a feature vector $\mathbf{x}_v = [\mathbf{x}_v^{\text{geo}}\,;\,\mathbf{x}_v^{\text{poi}}\,;\,\mathbf{x}_v^{\text{heat}}]$ that aggregates road geometry, surrounding point-of-interest (POI) context, and historical traffic heat.

Trajectory. A GPS trajectory is a variable-length sequence of timestamped locations $\boldsymbol{\tau}_{\text{p}} = [(\text{lon}_i,\ \text{lat}_i,\ t_i)]_{i=0}^{n-1}$, where $(\text{lon}_i,\ \text{lat}_i)$ denotes the coordinate at timestamp t_i. After map-matching $\boldsymbol{\tau}_{\text{p}}$ to the road network and merging consecutive points that fall on the same segment, the trajectory can be equivalently represented at the road-segment level as $\boldsymbol{\tau}_{\text{r}} = [r_1,\ r_2,\ \ldots,\ r_m]$ with $r_i \in \mathcal{V}$, where $m \leq n$ since multiple GPS points may share the same road segment. The two representations describe the same trip at different granularities.

Problem formulation. Given a road network $\mathcal{G}$, road-segment features $\{\mathbf{x}_v\}_{v\in\mathcal{V}}$, and a trip condition $\boldsymbol{c}$ specifying origin, destination, and departure time, our goal is to generate a realistic GPS trajectory $\hat{\boldsymbol{\tau}}_{\text{p}}$. We formulate this as a coarse-to-fine generation process:

$$F_{\text{r}} :\ (\mathcal{G}, \{\mathbf{x}_v\},\ \boldsymbol{c}) \ \longrightarrow\ \hat{\boldsymbol{\tau}}_{\text{r}}, \qquad F_{\text{p}} :\ (\hat{\boldsymbol{\tau}}_{\text{r}},\ \mathcal{G},\ \boldsymbol{c}) \ \longrightarrow\ \hat{\boldsymbol{\tau}}_{\text{p}}. \tag{1}$$

Here, F_{r} generates a topology-valid road-segment route $\hat{\boldsymbol{\tau}}_{\text{r}}$, and F_{p} refines it into a fine-grained GPS trajectory $\hat{\boldsymbol{\tau}}_{\text{p}}$. The generated trajectory is required to respect road-network topology, support variable-length output, and generalize across cities with different road networks.

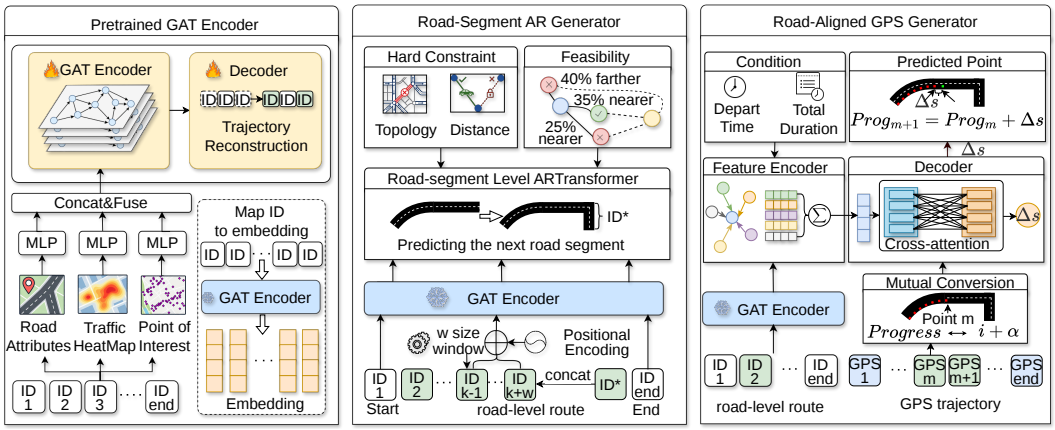


Figure 2: The detailed architecture of XTraj. A GAT encoder is first pretrained through masked trajectory reconstruction with general road attributes and local graph topology. The Road-Segment-Level Autoregressive Generator then uses the learned road representations to generate a road-segment route. Finally, the Road-Aligned GPS Generator decodes the generated route into a GPS trajectory.

4 Methodology

As stated in Figure 2, XTraj generates trajectories through three sequential modules that together implement the coarse-to-fine framework. A road-segment encoder first learns transferable representations of road segments from general road attributes and graph topology, sidestepping reliance on city-specific identifiers. The Road-Segment-Level Autoregressive Generator then produces variable-length road-segment routes by predicting connected segments within the road network. Finally, the Road-Aligned GPS Generator refines each generated road-segment route into a fine-grained GPS trajectory by predicting monotonic progress values along the route and projecting them onto the road geometry. This decomposition addresses the three limitations identified. The encoder counters geographic overfitting, the Road-Segment-Level Autoregressive Generator removes the need for length-rigid modeling, and the Road-Aligned GPS Generator eliminates off-road drift by construction.

4.1 Transferable Road-Segment Encoder

The first module of XTraj counters geographic overfitting by learning road-segment representations from attributes that are available in any city, rather than from absolute coordinates or city-specific identifiers. Each road segment r_i is described by three types of features: geometric features $\mathbf{x}_i^{\text{geo}}$, surrounding POI features $\mathbf{x}_i^{\text{poi}}$, and traffic-heat features $\mathbf{x}_i^{\text{heat}}$. All three feature types are computable from public road and POI sources, ensuring that the encoder remains applicable across cities.

Feature fusion and graph encoding. Each feature type is first projected into a shared latent space through a type-specific MLP, and the projected vectors are concatenated and fused into a unified hidden representation:

$$\mathbf{h}_i = \text{MLP}_{\text{fuse}}\Big(\text{MLP}_{\text{geo}}(\mathbf{x}_i^{\text{geo}})\,;\,\text{MLP}_{\text{poi}}(\mathbf{x}_i^{\text{poi}})\,;\,\text{MLP}_{\text{heat}}(\mathbf{x}_i^{\text{heat}})\Big)\,, \tag{2}$$

where ; denotes concatenation. The fused vector $\mathbf{h}_i$ summarizes the local attributes of segment r_i before any graph-level interaction. To incorporate road-network topology, we propagate $\{\mathbf{h}_i\}$ over the directed road graph $\mathcal{G}$ using L stacked Graph Attention Network (GAT) layers with multi-head attention [25]. At each layer, segment r_i aggregates information from its incoming neighbors $\mathcal{N}_{\text{in}}(i)$ via attention weights computed from feature similarity, followed by residual connections and layer normalization. After L layers, a linear projection produces the final road embedding $\mathbf{z}_i \in \mathbb{R}^{d_{\text{emb}}}$, which encodes both the segment's own attributes and its L-hop topological context. We collect all segment embeddings into a matrix $\mathbf{Z} = [\mathbf{z}_1^\top, \ldots, \mathbf{z}_{|\mathcal{V}|}^\top]$.

Masked trajectory pretraining. Static graph encoding alone may not capture how segments are actually connected through real travel behavior. To inject sequential mobility priors into the

embeddings, we pretrain the encoder with a masked trajectory reconstruction objective inspired by masked language modeling [7]. Given a road-segment trajectory $\boldsymbol{\tau}_{\mathrm{r}} = [r_1, \ldots, r_m]$, we retrieve its embedding sequence $[\mathbf{z}_{r_1}, \ldots, \mathbf{z}_{r_m}]$ from $\mathbf{Z}$, randomly mask a fraction ρ of internal positions $\mathcal{M} \subset \{2, \ldots, m-1\}$, and feed the masked sequence into a lightweight bidirectional Transformer reconstruction head. For each masked position $t \in \mathcal{M}$, the head outputs a contextualized hidden state $\tilde{\mathbf{h}}_t$, and the model is trained to identify the true segment r_t from all candidates in $\mathcal{V}$ via a contrastive softmax objective:

$$\mathcal{L}_{\mathrm{enc}} = -\sum_{t \in \mathcal{M}} \log \frac{\exp\left(\tilde{\mathbf{h}}_t^\top \mathbf{z}_{r_t}\right)}{\sum_{j=1}^{|\mathcal{V}|} \exp\left(\tilde{\mathbf{h}}_t^\top \mathbf{z}_j\right)}. \tag{3}$$

By matching contextualized hidden states against the global embedding pool, this objective shapes the road embeddings $\mathbf{z}_i$ to reflect not only static attributes and graph topology but also the sequential roles segments play in real trajectories.

Cross-city applicability. After pretraining, the encoder produces road embeddings solely from features provided by any city's road network and POI database. Therefore, applying it to a new city requires no further training: we only need to construct the local road graph and recompute the three feature types, and the resulting embeddings can be directly consumed by the downstream generators.

4.2 Road-Segment-Level Autoregressive Generator

The second module replaces length-rigid generation with an autoregressive process that produces routes of adaptive length. Built on the frozen embeddings from Section 4.1, the generator predicts road segments one at a time and terminates once it reaches the destination, supporting road-segment routes of arbitrary length without requiring resampling. Two structural constraints (a topology constraint and a destination-directed distance constraint) ensure that the generated route remains graph-valid and reaches the specified destination.

Autoregressive next-segment prediction. Let $\hat{\boldsymbol{\tau}}_{\mathrm{r},\,1:t} = [r_1, \ldots, r_t]$ denote the partial route generated so far, and let $\boldsymbol{c}$ encode the trip condition (origin, destination r_{dst}, and departure time). At each step t, the input to a causal Transformer decoder is the sequence of pretrained embeddings $[\mathbf{z}_{r_1}, \ldots, \mathbf{z}_{r_t}]$ enriched with positional encodings, restricted to the most recent W tokens for efficiency. The trip condition $\boldsymbol{c}$ is injected as a prefix token. The decoder produces a hidden state $\mathbf{h}_t$ at the current position, and the next-segment distribution is obtained by projecting $\mathbf{h}_t$ onto all road embeddings:

$$\mathbf{h}_t = \mathrm{TransformerDec}(\mathbf{z}_{r_1}, \ldots, \mathbf{z}_{r_t};\ \boldsymbol{c}), \qquad P(r_{t+1} \mid \hat{\boldsymbol{\tau}}_{\mathrm{r},1:t},\ \boldsymbol{c}) = \mathrm{Softmax}(\mathbf{Z}\mathbf{h}_t), \tag{4}$$

where $\mathbf{Z} \in \mathbb{R}^{|\mathcal{V}| \times d_{\mathrm{emb}}}$ is the road-embedding matrix. Because a locally probable next segment may be unreachable from r_t in the road graph, we further mask out all candidates outside the legal outgoing neighborhood $\mathcal{N}_{\mathrm{out}}(r_t) \subset \mathcal{V}$ before the softmax, so that probability mass is distributed only over connected segments. This topology mask is applied during both training and inference, ensuring that the model assigns probability only to graph-valid transitions.

Distance-monotone decoding. At inference time, autoregressive decoding can still drift away from the destination or fall into local cycles. To prevent this, we impose a destination-directed distance constraint based on precomputed shortest-path distances $d(\cdot, r_{\mathrm{dst}})$. Among the candidates passing the topology mask, we accept the highest-scoring segment r_{t+1} that does not increase the distance to the destination:

$$r_{t+1} = \arg \max_{r \in \mathcal{N}_{\mathrm{out}}(r_t) \cap \mathcal{S}_t} P(r \mid \hat{\boldsymbol{\tau}}_{\mathrm{r},\,1:t},\ \boldsymbol{c}), \qquad \mathcal{S}_t = \{r :\ d(r,\ r_{\mathrm{dst}}) \le d(r_t,\ r_{\mathrm{dst}})\}, \tag{5}$$

and decoding terminates when $r_{t+1} = r_{\mathrm{dst}}$. The constraint is non-strict ($\le$ rather than $<$) so that detours of equal distance, common in real road networks with parallel paths, are not excluded. By keeping each step at least as close to the destination as the previous one, this rule prevents long-range drift while still allowing the model to follow learned local route preferences.

Training objective. During training, the topology mask $\mathcal{N}_{\text{out}}(r_t)$ is applied to ensure that the cross-entropy is computed over the same restricted candidate set used at inference, while the distance constraint is omitted to expose the model to the full distribution of real routes. The training loss is

$$\mathcal{L}_{\text{road}} = -\sum_{t=1}^{m-1} \log P(r_{t+1} \mid \boldsymbol{\tau}_{\text{r},\,1:t}, \boldsymbol{c}) \,, \tag{6}$$

where P is the softmax over $\mathcal{N}_{\text{out}}(r_t)$. This objective directly optimizes the likelihood of real road-segment sequences under graph-valid transitions.

Variable-length output. Because decoding terminates when the destination is reached rather than after a fixed number of steps, the generator naturally produces routes whose length adapts to the underlying trip distance. The resulting generated road-segment route $\hat{\boldsymbol{\tau}}_{\text{r}} = [r_1,\, \ldots,\, r_M]$ can serve either as a coarse output by itself or as input to the Road-Aligned GPS Generator.

4.3 Road-Aligned GPS Generator

The third module converts the generated road-segment route into fine-grained GPS trajectories while ensuring that every output point lies on a valid road segment. To this end, we represent each GPS point as continuous progress along the route rather than as an unconstrained 2D coordinate, and let the model autoregressively predict monotonic progress increments. Coordinates are then recovered by projecting each progress value onto the corresponding road geometry, so that road-geometry consistency holds by construction.

Route-aligned progress representation. Given the generated road-segment route $\hat{\boldsymbol{\tau}}_{\text{r}} = [r_1,\, \ldots,\, r_M]$, we represent the t-th GPS point along the route as a scalar progress value

$$s_t = i_t + \alpha_t, \qquad i_t \in \{0,\, 1,\, \ldots,\, M-1\},\ \alpha_t \in [0,\, 1), \tag{7}$$

where i_t indexes the road segment containing the point (the $(i_t{+}1)$-th segment in the route) and α_t is the relative arc-length position within that segment. By construction, s_t is monotonically non-decreasing, with $s_0 = 0$ at the route start and $s_{\text{end}} = M$ at the destination. Predicting non-negative progress increments $\Delta s_t = s_t - s_{t-1}$ instead of absolute coordinates carries two structural advantages: Every progress value can be deterministically projected onto the road geometry, so generated points always lie on a valid segment. Progress is defined relative to the generated route rather than to any city-wide coordinate system, decoupling the generator from city-specific spatial distributions and enabling cross-city reuse.

Path-conditioned decoder. We adopt an encoder–decoder architecture. The encoder takes the route embedding sequence $[\mathbf{z}_{r_1},\, \ldots,\, \mathbf{z}_{r_M}]$ enriched with positional encodings and trip-level conditions $\boldsymbol{c}_{\text{trip}}$ (departure time, total trip duration, and target end progress s_{end}), and produces a path memory $\mathbf{M} \in \mathbb{R}^{M \times d_{\text{model}}}$ via a bidirectional Transformer. Conditioned on $\mathbf{M}$, a causal Transformer decoder predicts progress increments step by step: at step t, it attends to the most recent W progress values $[s_{\max(0,\, t-W)},\, \ldots,\, s_{t-1}]$ via self-attention and to $\mathbf{M}$ via cross-attention, producing a hidden state $\mathbf{g}_t$. To help the decoder pace its prediction throughout the trip, we additionally concatenate the remaining trip ratio $\eta_t = 1 - t/(n-1)$ to the decoder input at each step, where n is the target number of GPS points; this informs the model how much of the trip is left so that the increments reach the destination at the appropriate pace rather than overshooting or stalling. The increment is produced through an MLP head with a softplus activation,

$$\Delta\hat{s}_t = \text{Softplus}(\text{MLP}_{\Delta}(\mathbf{g}_t)) \,, \qquad \hat{s}_t = \hat{s}_{t-1} + \Delta\hat{s}_t, \tag{8}$$

which guarantees that the predicted progress is monotonically non-decreasing.

Training and inference. For each training trajectory, we precompute the ground-truth progress sequence $[s_t]_{t=0}^{n-1}$ by map-matching every GPS point to the route and computing its arc-length ratio within the matched segment, and we train the decoder with a SmoothL1 loss on the increments.

$$\mathcal{L}_{\text{GPS}} = \frac{1}{n-1} \sum_{t=1}^{n-1} \text{SmoothL1}(\Delta\hat{s}_t,\, \Delta s_t) \,, \quad \Delta s_t = s_t - s_{t-1}. \tag{9}$$



Real DiffTraj ControlTraj CarDiff XTraj

Figure 3: Geographic visualization of the generated trajectory in Xi'an.

At inference, each predicted $\hat{s}_t = \hat{i}_t + \hat{\alpha}_t$ is mapped back to a GPS coordinate by walking $\hat{\alpha}_t$ of the arc length along segment $r_{\hat{i}_t+1}$, $(\widehat{\mathrm{lon}}_t, \widehat{\mathrm{lat}}_t) = \mathrm{Proj}(\hat{s}_t, \hat{\boldsymbol{\tau}}_\mathrm{r})$, where $\mathrm{Proj}(\cdot)$ denotes deterministic positioning on the road geometry. Since Proj always returns a point on the route, every generated GPS coordinate satisfies road-geometry consistency by construction, removing the need for post-hoc map matching.

5 Experiments

We evaluate XTraj on real-world trajectory datasets through baseline comparisons, training-free cross-city transfer, variable-length analysis, spatial-resolution analysis, and component ablations. XTraj achieves stronger distributional fidelity than the evaluated baselines while preserving road alignment in both in-city and transfer settings. We further examine its performance across trajectory-length groups and spatial resolutions to assess its robustness under different evaluation conditions. Finally, ablation studies analyze how multimodal road representations, destination-directed routing constraints and the Road-Segment-Level Autoregressive Generator contribute to spatial fidelity and route completion.

5.1 Experimental Setup

Datasets. We use trajectory datasets from DIDI Chuxing covering Xi'an and Chengdu, which differ in road layouts and mobility patterns. GPS trajectories are map-matched to directed road graphs extracted from OpenStreetMap. Road-segment features describe geometry, surrounding POI context, and historical traffic intensity. For GPS trajectory modeling, matched points are converted into progress values along their corresponding road-segment routes. Further preprocessing details are provided in Appendix B.

Baselines. We compare XTraj with sequence-based models, including VAE [15] and GAN [2], as well as diffusion-based generators, including DiffTraj [39], Diff-RNTraj [30], ControlTraj [40], and CarDiff [9]. ControlTraj receives the ground-truth segment route as an additional input, giving it access to route information that is not available to XTraj.

Evaluation Metrics. Following common practice [39, 40], we organize evaluation into four groups. Spatial fidelity is measured by Density, the Jensen–Shannon divergence (JSD) between generated and real GPS-point distributions on a 256×256 grid. Trip properties are measured by Trip, the JSD between origin-destination distributions, and Length, the JSD between trajectory-length histograms. Physical consistency is measured by Map-Matching Rate (MMR), the proportion of generated points requiring post-hoc map matching, and Average Shift, the mean distance from generated points to the nearest valid road segment. Routing robustness is measured by Success Rate, the proportion of routes reaching the destination, and Intervention Rate, the frequency of path-constraint activation during decoding. Lower values are better for Density, Trip, Length, MMR, Average Shift, and Intervention Rate; higher values are better for Success Rate.

Transfer Setting. Training-free cross-city transfer applies a source-city model to the other city's road network without additional parameter training or fine-tuning. The transferred model, denoted XTraj-Transfer, uses road embeddings computed for the target network. We distinguish this setting from in-city evaluation, where the model is trained on trajectories from the evaluation city.

Table 1: Generation performance, including training-free transfer. Lower values indicate better performance.

Model	Xi'an					Chengdu				
	Density	Length	Trip	MMR	Shift	Density	Length	Trip	MMR	Shift
VAE	0.3892	0.0945	0.1052	0.9832	31.83m	0.3254	0.0342	0.1384	0.9867	29.85m
GAN	0.3542	0.0618	0.0921	0.9786	31.76m	0.2987	0.0319	0.1147	0.9766	28.98m
Diff-RNTraj	0.0939	0.3168	0.0548	**0.0000**	**0.00m**	0.0457	0.2220	0.0457	**0.0000**	**0.00m**
DiffTraj	0.2324	0.0078	0.0420	0.9757	30.03m	0.2361	0.0164	0.0340	0.9748	28.26m
ControlTraj	0.1545	0.0047	0.0180	0.9645	17.69m	0.1172	0.0134	0.0103	0.9523	14.18m
CarDiff	0.1801	0.0051	0.0178	0.9497	13.58m	0.1664	0.0142	0.0316	0.9571	16.73m
XTraj-Transfer	**0.0747**	**0.0163**	**0.0079**	**0.0000**	**0.00m**	**0.0780**	**0.0173**	**0.0048**	**0.0000**	**0.00m**
XTraj (Ours)	**0.0592**	**0.0023**	**0.0073**	**0.0000**	**0.00m**	**0.0412**	**0.0035**	**0.0061**	**0.0000**	**0.00m**

5.2 In-City Generation Performance

Realistic trajectory generation requires both fidelity to observed mobility distributions and adherence to road geometry. Table 1 compares XTraj with existing methods in Xi'an and Chengdu using Density, Length, Trip, Map-Matching Rate, and Average Shift. XTraj achieves the lowest distributional values among the in-city models in both cities while matching Diff-RNTraj in road alignment with zero Average Shift. Although Diff-RNTraj also achieves perfect road alignment, its Length value is substantially higher, indicating that geometric validity alone does not guarantee accurate trip-level distributional fidelity. XTraj instead reproduces spatial density, trip structure, and trajectory-length distributions while preserving explicit road consistency. Figure 3 provides a visual comparison of the generated trajectories in Xi'an and shows that XTraj preserves the visible street structure more clearly than coordinate-generating baselines, whose outputs are more spatially diffuse around local roads and intersections. Together, these visual analysis and quantitative results demonstrate that XTraj captures both trip-level distributions and fine-grained road-network patterns.

5.3 Training-Free Cross-City Transfer

Table 2: Quantitative results of training-free transfer performance. This table compares XTraj and base models with their training-free transfer across Xi'an and Chengdu. Lower values indicate better performance.

Model	Xi'an			Chengdu		
	Density	Length	Trip	Density	Length	Trip
ControlTraj	0.1545	0.0047	0.0180	0.1172	0.0134	0.0103
ControlTraj-Transfer	0.6296	0.6021	0.5601	0.5505	0.6112	0.5611
CarDiff	0.1801	0.0051	0.0178	0.1664	0.0142	0.0316
CarDiff-Transfer	0.5382	0.5283	0.5523	0.5593	0.5322	0.5598
XTraj (Ours)	**0.0592**	**0.0023**	**0.0073**	**0.0412**	**0.0035**	**0.0061**
XTraj-Transfer (Ours)	**0.0747**	**0.0163**	**0.0079**	**0.0780**	**0.0173**	**0.0048**

Table 2 compares XTraj-Transfer with transferred baselines on cities not used for model training. Across the evaluated cities, XTraj-Transfer achieves the lowest Density, Trip, and Length values among the transferred models while preserving road alignment. Coordinate-based baselines show substantial degradation on unfamiliar networks, illustrating the difficulty of transferring coordinate-dependent generation across cities. In contrast, XTraj enables training-free cross-city transfer through reusable road representations and route-relative progress prediction.

5.4 Density Performance Across Spatial Resolutions

A coarse density grid can conceal differences in the placement of generated points around individual roads. Figure 4 therefore compares Density across eight spatial resolutions ranging from 64×64 to 8192×8192 in Xi'an and Chengdu. XTraj achieves the lowest Density values at every resolution in both cities, while diffusion-based baselines such as ControlTraj and DiffTraj remain substantially higher. Its advantage is particularly pronounced at coarse-to-medium resolutions, indicating closer

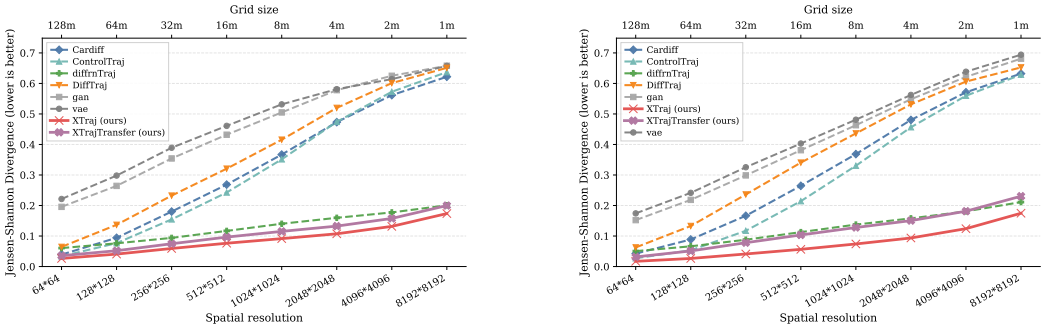


Figure 4: Multi-scale Density analysis on the Xi'an (left) and Chengdu (right) dataset. Performance is evaluated across eight grid resolutions ranging from 64×64 to 8192×8192, comparing 8 models that include XTraj (ours) and its transfer variant.

agreement with broad mobility distributions. As the grid becomes finer, Density increases for all methods, but XTraj retains its advantage at street-level granularity. XTraj-Transfer also closely follows the in-city curve across resolutions, supporting training-free transfer across spatial scales. Together, these results are consistent with transferable road representations and route-aligned GPS generation supporting density fidelity across spatial resolutions and cities.

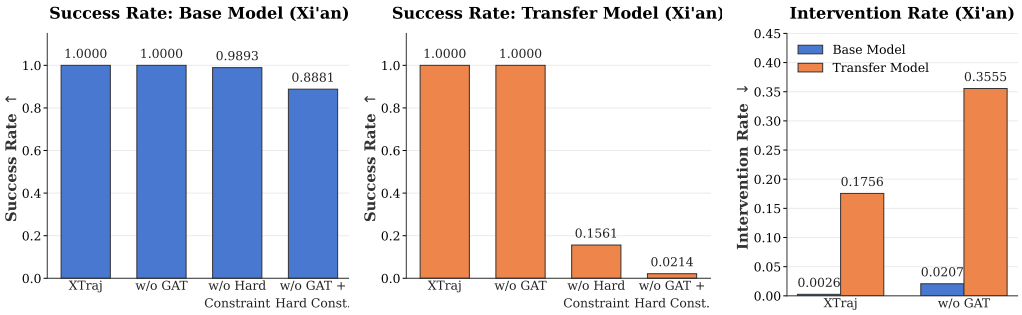
5.5 Ablation Study

Table 3: Ablation study on architectural components and routing constraints.

City	Variant	Density ↓	Success Rate ↑	Intervention Rate ↓
Xi'an	**XTraj**	**0.0592**	1.0000	0.0026
	w/o GAT	0.0886	1.0000	0.0207
	w/o Hard Constraint	0.0703	0.9893	–
	w/o GAT + Hard Constraint	–	0.8881	–
	w/o Road-Segment-Level Autoregressive Generator	0.0862	1.0000	–
	XTraj-Transfer	**0.0747**	1.0000	**0.1756**
	w/o GAT transfer	0.1261	1.0000	0.3555
	w/o Hard Constraint transfer	–	**0.1561**	–
	w/o GAT + Hard Constraint transfer	–	0.0214	–
Chengdu	**XTraj**	**0.0412**	1.0000	0.0076
	w/o GAT	0.0741	1.0000	0.0175
	w/o Hard Constraint	0.0454	0.9650	–
	w/o GAT + Hard Constraint	–	0.9062	–
	w/o Road-Segment-Level Autoregressive Generator	0.0966	1.0000	–
	XTraj-Transfer	**0.0780**	1.0000	**0.1921**
	w/o GAT transfer	0.1133	1.0000	0.3992
	w/o Hard Constraint transfer	–	**0.2008**	–
	w/o GAT + Hard Constraint transfer	–	0.0196	–

Table 3 and Figure 5 examine the contributions of the multimodal GAT encoder, destination-directed hard constraint, and the Road-Segment-Level Autoregressive Generator. We compare spatial fidelity and routing robustness in both in-city and training-free transfer settings. Success Rate measures the proportion of routes reaching the target destination, while Intervention Rate measures how often the constraint overrides the decoder's top prediction.

Cross-city Generalization via Multimodal GAT. Removing the GAT encoder increases Density and Intervention Rate in both cities, with larger deterioration under training-free transfer. Although destination completion is maintained when the hard constraint remains enabled, the increased intervention frequency indicates that the decoder's preferred transitions become less consistent with destination-directed routing. These results support the contribution of graph-based road representations to spatial fidelity and routing reliability beyond the validity enforced by explicit constraints.



(a) (b) (c)

Figure 5: Ablation study results in Xi'an. (a) success rate (b) success rate under training-free transfer; (c) Intervention Rate

Preventing Error Accumulation via Hard Constraints. Removing the hard constraint causes a comparatively modest reduction in in-city Success Rate but substantially reduces destination completion under training-free transfer. This contrast suggests that learned routing preferences become less reliable on unfamiliar road networks. Explicit destination-directed guidance therefore provides important support for route completion when the model operates outside its training city.

Route Planning via Road Segment Generator. To isolate the contribution of the Road-Segment-Level Autoregressive Generator, we replace it with shortest-path routing while retaining the same Road-Aligned GPS Generator. Table 3 shows that removing the Road-Segment-Level Autoregressive Generator increases Density values. These results indicate that learned route selection better reproduces observed spatial distributions than deterministic shortest-path routing alone, even when destination completion is maintained.

6 Conclusion

We present XTraj, a transferable coarse-to-fine framework for realistic vehicle trajectory generation. Rather than generating fixed-length GPS sequences, XTraj decomposes the task into road-segment route generation and GPS trajectory refinement. A multimodal GAT encoder learns transferable road representations from geometry, POI context, traffic heat, and graph topology. Using these embeddings, the Road-Segment-Level Autoregressive Generator produces variable-length, topology-valid road-segment routes, while the Road-Aligned GPS Generator predicts monotonic route-aligned progress and recovers GPS coordinates through road-geometry interpolation. Experiments on Xi'an and Chengdu datasets show that XTraj improves spatial fidelity over diffusion-based baselines in both in-city and training-free cross-city scenarios, while guaranteeing perfect road-geometry consistency. Ablation studies confirm that graph-based road representation learning is crucial for both generation quality and transferability. Overall, XTraj demonstrates that combining multimodal road embeddings with coarse-to-fine autoregressive generation provides an effective approach for scalable, transferable, and road-consistent trajectory synthesis.

Acknowledgement

This work is supported by the National Natural Science Foundation of China under Grant No. 62506097, U25A20530, U25B2045, by the Shenzhen Basic Research Program Natural Science Foundation under Grant No. JCYJ20250604145542055, and by the Guangdong Basic and Applied Basic Research Foundation under Grant No. 2026A1515010223.

References

[1] Hugo Barbosa, Marc Barthelemy, Gourab Ghoshal, Charlotte R James, Maxime Lenormand, Thomas Louail, Ronaldo Menezes, José J Ramasco, Filippo Simini, and Marcello Tomasini. Human mobility: Models and applications. *Physics Reports*, 734:1–74, 2018.

[2] Chu Cao and Mo Li. Generating mobility trajectories with retained data utility. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 2610–2620, 2021.

[3] Ji Cao, Tongya Zheng, Qinghong Guo, Yu Wang, Junshu Dai, Shunyu Liu, Jie Yang, Jie Song, and Mingli Song. Holistic semantic representation for navigational trajectory generation. *arXiv preprint arXiv:2501.02737*, 2025.

[4] Yanchuan Chang, Jianzhong Qi, Yuxuan Liang, and Egemen Tanin. Contrastive trajectory similarity learning with dual-feature attention. In *2023 IEEE 39th International conference on data engineering (ICDE)*, pages 2933–2945. IEEE, 2023.

[5] Xinyu Chen, Jiajie Xu, Rui Zhou, Wei Chen, Junhua Fang, and Chengfei Liu. Trajvae: A variational autoencoder model for trajectory generation. *Neurocomputing*, 428:332–339, 2021.

[6] Seongjin Choi, Jiwon Kim, and Hwasoo Yeo. Trajgail: Generating urban vehicle trajectories using generative adversarial imitation learning. *Transportation Research Part C: Emerging Technologies*, 128:103091, 2021.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[8] Jie Feng, Zeyu Yang, Fengli Xu, Haisu Yu, Mudan Wang, and Yong Li. Learning to simulate human mobility. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3426–3433, 2020.

[9] Baoshen Guo, Zhiqing Hong, Junyi Li, Shenhao Wang, and Jinhua Zhao. Leveraging the spatial hierarchy: Coarse-to-fine trajectory generation via cascaded hybrid diffusion. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 359–370, 2026.

[10] Jiawei Jiang, Dayan Pan, Houxing Ren, Xiaohan Jiang, Chao Li, and Jingyuan Wang. Self-supervised trajectory representation learning with temporal regularities and travel semantics. In *2023 IEEE 39th international conference on data engineering (ICDE)*, pages 843–855. IEEE, 2023.

[11] Wenjun Jiang, Wayne Xin Zhao, Jingyuan Wang, and Jiawei Jiang. Continuous trajectory generation based on two-stage gan. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 4374–4382, 2023.

[12] Fengmei Jin, Wen Hua, Matteo Francia, Pingfu Chao, Maria E Orlowska, and Xiaofang Zhou. A survey and experimental study on privacy-preserving trajectory data publishing. *IEEE Transactions on Knowledge and Data Engineering*, 35(6):5577–5596, 2022.

[13] Guangyin Jin, Yuxuan Liang, Yuchen Fang, Zezhi Shao, Jincai Huang, Junbo Zhang, and Yu Zheng. Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. *IEEE transactions on knowledge and data engineering*, 36(10):5388–5408, 2023.

[14] Jong Wook Kim and Beakcheol Jang. Privacy-preserving generation and publication of synthetic trajectory microdata: A comprehensive survey. *Journal of Network and Computer Applications*, 230:103951, 2024.

[15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[16] Xiangjie Kong, Qiao Chen, Mingliang Hou, Hui Wang, and Feng Xia. Mobility trajectory generation: a survey. *Artificial Intelligence Review*, 56(Suppl 3):3057–3098, 2023.

[17] Xiucheng Li, Kaiqi Zhao, Gao Cong, Christian S Jensen, and Wei Wei. Deep representation learning for trajectory similarity computation. In *2018 IEEE 34th international conference on data engineering (ICDE)*, pages 617–628. IEEE, 2018.

[18] Yan Lin, Huaiyu Wan, Shengnan Guo, and Youfang Lin. Pre-training context and time aware location embeddings from spatial-temporal trajectories for user next location prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4241–4248, 2021.

[19] Yichen Liu, Yan Lin, Shengnan Guo, Zeyu Zhou, Youfang Lin, and Huaiyu Wan. Trajmamba: An efficient and semantic-rich vehicle trajectory pre-training model. *arXiv preprint arXiv:2510.17545*, 2025.

[20] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[21] Luca Pappalardo, Filippo Simini, Salvatore Rinzivillo, Dino Pedreschi, Fosca Giannotti, and Albert-László Barabási. Returners and explorers dichotomy in human mobility. *Nature communications*, 6(1):8166, 2015.

[22] Luca Pappalardo, Salvatore Rinzivillo, and Filippo Simini. Human mobility modelling: exploration and preferential return meet the gravity model. *Procedia Computer Science*, 83:934–939, 2016.

[23] Vincent Primault, Antoine Boutet, Sonia Ben Mokhtar, and Lionel Brunie. The long road to computational location privacy: A survey. *IEEE Communications Surveys & Tutorials*, 21(3): 2772–2793, 2018.

[24] Samuel A Stouffer. Intervening opportunities: a theory relating mobility and distance. *American sociological review*, 5(6):845–867, 1940.

[25] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[26] Di Wang, Tomio Miwa, and Takayuki Morikawa. Big trajectory data mining: A survey of methods, applications, and services. *Sensors*, 20(16):4571, 2020.

[27] Huandong Wang, Changzheng Gao, Yuchen Wu, Depeng Jin, Lina Yao, and Yong Li. Pategail: a privacy-preserving mobility trajectory generator with imitation learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 14539–14547, 2023.

[28] Leye Wang, Bin Guo, and Qiang Yang. Smart city development with urban transfer learning. *Computer*, 51(12):32–41, 2018.

[29] Sheng Wang, Zhifeng Bao, J. Shane Culpepper, and Gao Cong. A survey on trajectory data management, analytics, and learning. *ACM Comput. Surv.*, 54(2), March 2021. ISSN 0360-0300. doi: 10.1145/3440207. URL https://doi.org/10.1145/3440207.

[30] Tonglong Wei, Youfang Lin, Shengnan Guo, Yan Lin, Yiheng Huang, Chenyang Xiang, Yuqing Bai, and Huaiyu Wan. Diff-rntraj: A structure-aware diffusion model for road network-constrained trajectory generation. *IEEE Transactions on Knowledge and Data Engineering*, 36 (12):7940–7953, 2024.

[31] Tonglong Wei, Yan Lin, Zeyu Zhou, Haomin Wen, Jilin Hu, Shengnan Guo, Youfang Lin, Gao Cong, and Huaiyu Wan. Transfertraj: A vehicle trajectory learning model for region and task transferability. *arXiv preprint arXiv:2505.12672*, 2025.

[32] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5668–5675, 2019.

[33] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[34] Jing Zhang, Qihan Huang, Yirui Huang, Qian Ding, and Pei-Wei Tsai. Dp-trajgan: A privacy-aware trajectory generation model with differential privacy. *Future Generation Computer Systems*, 142:25–40, 2023.

[35] Yu Zheng. Trajectory data mining: An overview. *ACM Trans. Intell. Syst. Technol.*, 6(3), May 2015. ISSN 2157-6904. doi: 10.1145/2743025. URL https://doi.org/10.1145/2743025.

[36] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. Urban computing: Concepts, methodologies, and applications. *ACM Trans. Intell. Syst. Technol.*, 5(3), September 2014. ISSN 2157-6904. doi: 10.1145/2629592. URL https://doi.org/10.1145/2629592.

[37] Silin Zhou, Shuo Shang, Lisi Chen, Christian S Jensen, and Panos Kalnis. Red: Effective trajectory representation learning with comprehensive information. *arXiv preprint arXiv:2411.15096*, 2024.

[38] Weiping Zhu, Renhe Jiang, Dongyuan Li, Xiangjie Kong, and Xuan Song. Trajectory generative models: a survey from unconditional and conditional perspectives. *GeoInformatica*, 29(4): 1033–1065, 2025.

[39] Yuanshao Zhu, Yongchao Ye, Shiyao Zhang, Xiangyu Zhao, and James Yu. Difftraj: Generating gps trajectory with diffusion probabilistic model. *Advances in Neural Information Processing Systems*, 36:65168–65188, 2023.

[40] 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.

[41] Yuanshao Zhu, James Jianqiao Yu, Xiangyu Zhao, Xun Zhou, Liang Han, Xuetao Wei, and Yuxuan Liang. Unitraj: Learning a universal trajectory foundation model from billion-scale worldwide traces. *arXiv preprint arXiv:2411.03859*, 2024.

[42] George Kingsley Zipf. The p 1 p 2/d hypothesis: on the intercity movement of persons. *American sociological review*, 11(6):677–686, 1946.

A XTraj Implementation Details

A.1 Three-Stage Generation Pipeline

XTraj synthesizes trajectories progressively through three sequential stages. Each stage leverages the outputs and representations learned from the previous one, transitioning from coarse-grained macro-routing to fine-grained micro-progress mapping.

Stage 1 → Stage 2 → Stage 3:
Transferable Road-Segment Encoder (GATv2) → Road-Segment-Level Autoregressive Generator → Road-Aligned GPS Generator

A.2 Stage 1: Transferable Road-Segment Encoder

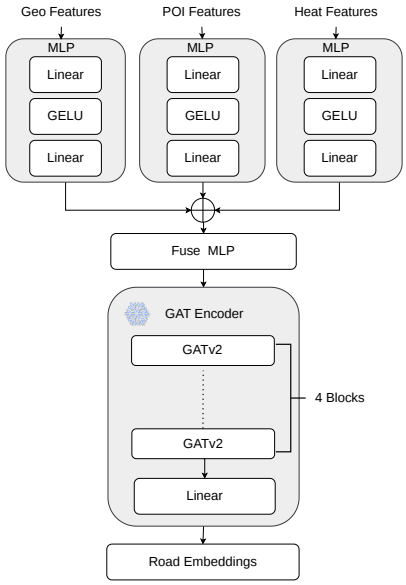


Figure 6: Model Architecture of Transferable Road-Segment Encoder

Task: Masked trajectory reconstruction. Road segments are encoded via a multi-modal GATv2 graph encoder. A lightweight Transformer decoder reconstructs masked road tokens from the encoder's node embeddings.

Masking strategy: 30% of interior tokens per trajectory segment are randomly masked. The decoder predicts masked road IDs via cross-entropy against the encoder's embedding matrix:

$$\mathcal{L}_{\text{enc}} = \sum_{t \in \mathcal{M}} \text{CrossEntropy}\left(\tilde{\mathbf{h}}_t^\top \mathbf{Z}^\top,\ r_t\right) \tag{10}$$

After pre-training, the robust road embeddings $\mathbf{Z} \in \mathbb{R}^{|\mathcal{V}| \times 64}$ are exported via a single forward pass and frozen for use in Stages 2 and 3.

Table 4: Architectural Hyperparameters for Stage 1: Transferable Road-Segment Encoder.

Hyperparameter	**Value**
Graph Encoder: MultiModalGATv2Encoder	
Architecture	GATv2 with multi-modal node features
Node features	Geographic (geo), POI (147 categories), traffic heat
Hidden dimension	128
Attention heads	4
Number of GATv2 layers	4
Output embedding dim	64
Dropout	0.1
Reconstruction Decoder	
Architecture	Masked Trajectory Reconstruction Transformer
d_{model}	64
Attention heads	4
Number of layers	2
Dropout	0.1
Segment length	8 road segments

A.3 Stage 2: Road-Segment-Level Autoregressive Generator

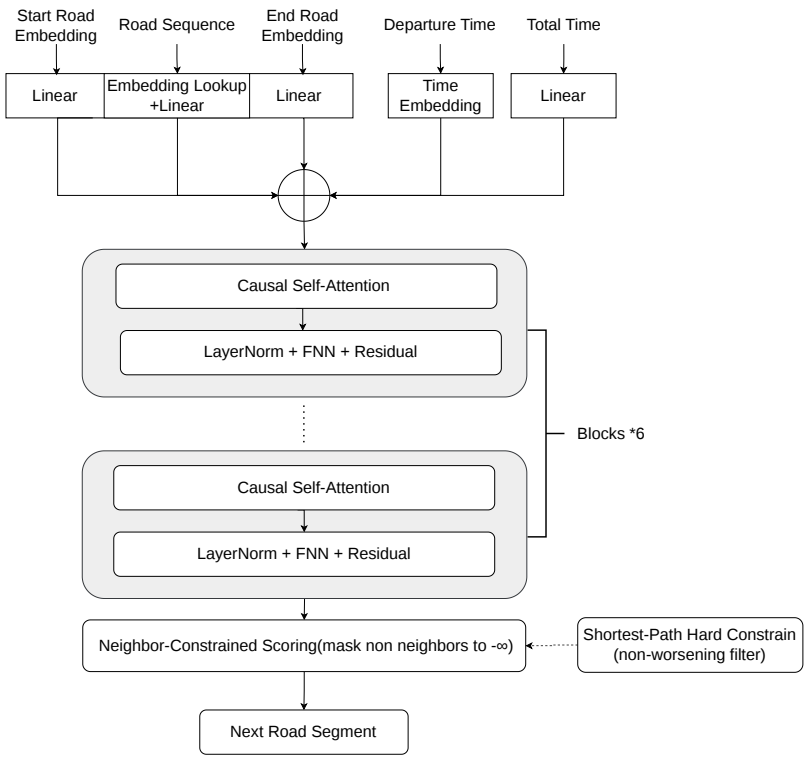


Figure 7: Model Architecture of Road-Segment-Level Autoregressive Generator

Task: Autoregressive next-hop prediction on the road graph. At each step, the model predicts which topological neighbor of the current road is the subsequent segment in the trajectory.

Neighbor constraint: To guarantee strict topological validity, only roads present in the precomputed neighbor table of the current road are scored. All other candidates are explicitly masked to $-\infty$ prior to the softmax operation.

Table 5: Architectural Hyperparameters for Stage 2: Road-Segment-Level Autoregressive Generator.

Hyperparameter	**Value**
Architecture	Pre-LayerNorm Transformer Encoder (causal)
Input projection	$\mathrm{Linear}(d_{\mathrm{emb}} \to d_{\mathrm{model}})$
d_{model}	256
Attention heads	8
Number of layers	6
Feed-forward dim	1024 ($4 \times d_{\mathrm{model}}$)
Activation	GELU
Dropout	0.1
Max sequence length	64 road segments
Causal window	6
Road embedding dim	64 (frozen from Stage 1)
Conditioning	Departure time, start/end road embedding, total duration

A.4 Stage 3: Road-Aligned GPS Generator

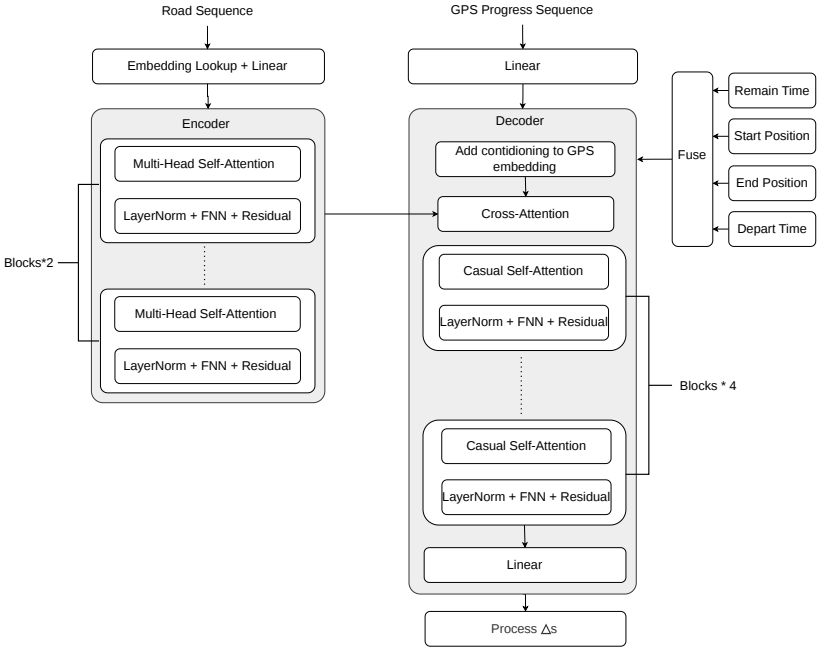


Figure 8: Model Architecture of Road-Aligned GPS Generator

Task: Given the generated road-segment route, the model autoregressively predicts the fine-grained, within-segment progress $\alpha_t \in [0, 1)$ at each GPS timestamp. The model conditions on the full generated road-segment route via a path encoder and subsequently decodes the GPS progress values step-by-step.

Monotone convergence guarantee: The output head applies a Softplus activation to predict non-negative increments $\Delta\hat{s}_t \geq 0$, mechanically ensuring that the overall progress $\hat{s}_t$ is strictly non-decreasing. Furthermore, an absolute upper bound s_{end} is enforced during decoding via a deterministic truncation: $\hat{s}_t = \min(\hat{s}_{t-1} + \Delta\hat{s}_t,\ s_{\mathrm{end}})$.

Table 6: Architectural Hyperparameters for Stage 3: Road-Aligned GPS Generator.

Hyperparameter	Value
Architecture	
Path Encoder	2-layer Transformer Encoder over the generated road-segment route
AR Decoder	4-layer Transformer Decoder over GPS steps
Local attention window	128 steps
Output head	$\text{Linear} \rightarrow \text{GELU} \rightarrow \text{Linear} \rightarrow \text{Softplus}$
d_{model}	256
Attention heads	8
Dropout	0.1
Max road segments	256
Max GPS steps	4096
Max Δs per step	2.0
Conditioning	Departure time, total duration, remaining time, start/end point

B Dataset and Preprocessing Details

To ensure the physical validity and high quality of the trajectory data used for training and evaluating XTraj, we implemented a rigorous, multi-stage preprocessing pipeline for the raw GPS trajectories. The pipeline transforms raw GPS traces into a clean, 1D road-progress representation suitable for trajectory modeling through three main steps.

B.1 Trajectory Repair and Topological Validation

Raw map-matched trajectories often contain topological gaps due to GPS signal loss or map-matching artifacts. To address this, we perform a systematic trajectory repair process. **Connectivity Bridging:** We validate road segment connections by checking if the endpoint of segment A exactly matches the start point of segment B. For disconnected segments, we utilize a Breadth-First Search (BFS) algorithm to find the shortest connecting path, restricted to inserting bridge paths of ≤ 3 segments. After this step, trajectories are separated into kept, repaired, and dropped subsets.

B.2 Quality Filtering and Semantic Enrichment

We further filter trajectories and road segments based on strict quality criteria and enrich the road graph with semantic context. **Connectivity Filtering:** We remove any remaining disconnected trajectories to ensure all sequences form valid paths. **POI Enrichment:** We filter Points of Interest (POIs) within the map boundaries and compute POI statistics within a $0.5\,\text{km}$ radius of each road segment's midpoint. These POIs are categorized based on the first 4 digits of their official typecodes.

B.3 GPS to Road-Progress Conversion

The final step converts 2D GPS coordinates into a 1D continuous road-progress representation:

- **Progress Calculation:** Each GPS point is mapped to a continuous scalar value $s_k = i_k + \alpha_k$. Here, i_k denotes the 0-based index of the road segment within the unique sequence of visited roads, and $\alpha_k \in [0, 1)$ represents the relative progress ratio along the polyline of that specific segment.
- **Zero-Coordinate Handling:** Invalid GPS points recorded at origin $(0, 0)$ contribute to the topological road-segment sequence but are excluded from the continuous s sequence.
- **Time Alignment and Sampling:** Time deltas (Δt) are strictly preserved. The accumulated time of any skipped points is added to the next valid point. Finally, from a 1-second baseline dataset, we perform a two-stage sampling to create a 3-second interval dataset (keeping indices 0, 3, 6, ..., plus the final destination point).

Table 7: Key hyperparameters for GPS trajectory preprocessing.

Hyperparameter	Value
Maximum inserted bridge segments	3
POI contextual search radius	0.5 km
Baseline temporal resolution	1 second
Subsampled temporal resolution	3 seconds

C Details of the Evaluation Metrics

To comprehensively evaluate the performance of XTraj and the baselines, we employ a suite of metrics covering distributional fidelity, physical consistency, and routing robustness.

C.1 Distributional Fidelity Metrics

We use Jensen-Shannon Divergence (JSD) to measure the alignment between the distributions of generated trajectories (Q) and real trajectories (P). JSD is a symmetric and smoothed version of the Kullback-Leibler (KL) divergence, defined as:

$$\mathrm{JSD}(P \parallel Q) = \frac{1}{2}\mathrm{KL}(P \parallel M) + \frac{1}{2}\mathrm{KL}(Q \parallel M)$$

where $M = \frac{1}{2}(P + Q)$.

- **Density (JSD-density):** We discretize the city map into a 256×256 grid and compute the 2D spatial distribution of all GPS points. JSD-density measures how well the generated traffic heatmap matches the real-world traffic density.
- **Length (JSD-length):** We compute the distribution of trajectory lengths (in terms of travel distance or the number of valid GPS points) and measure the JSD. This evaluates the model's ability to capture variable-length patterns.
- **Trip (JSD-trip):** We extract the Origin-Destination (OD) pairs for all trajectories, map them to discrete regional grids, and compute the JSD between the real and generated OD distributions.

C.2 Physical Consistency Metrics

These metrics assess whether the generated continuous GPS coordinates comply with the real-world physical road infrastructure.

- **Post-hoc Map Matching Rate (MMR):** This metric represents the proportion of generated trajectory points that require post-hoc map-matching or other means for trajectory repair. A lower MMR indicates better generated trajectory details and a reduced need for subsequent correction methods.
- **Average Shift:** The mean geographical distance (in meters) from each generated GPS point to its nearest valid road segment. While coordinate-based models typically exhibit significant spatial shifts requiring calibration, XTraj naturally yields a shift of $0.0\,\mathrm{m}$ due to its route-aligned design.

C.3 Routing Robustness Metrics

To evaluate the stability and effectiveness of the autoregressive routing mechanism, we introduce the following metrics:

- **Success Rate:** The proportion of generated road-segment routes that successfully navigate to the exact target destination node within the predefined maximum step limit ($L_{\max} - 1$). A failure is recorded if the sequence falls into an unresolvable dead-end or exceeds the length limit before arrival.
- **Intervention Rate:** The frequency at which the deterministic shortest-path hard constraint explicitly overrides the top-1 prediction of the AR Transformer. It is calculated as the ratio of decoding steps where the raw highest-probability token is invalid (either topologically

or directionally), forcing the model to select a lower-probability but valid candidate from the constrained eligible set. A lower intervention rate implies that the neural network has inherently learned superior physical routing priors without relying heavily on rule-based corrections.

D Additional Qualitative Results and Visualizations

To complement the quantitative evaluations presented in Section 5, this appendix provides comprehensive qualitative visualizations. These visualizations intuitively demonstrate XTraj's superior spatial fidelity, strict adherence to road topologies, and remarkable training-free transferability across both micro and macro spatial scales.

D.1 Single Trajectory Generation (Micro-level Fidelity)

Figures 9 to 12 illustrate the point-by-point generation process of XTraj compared to the reconstructed ground-truth trajectories. Given specific start (green circle) and end (red square) points, XTraj accurately generates the complete route along with the detailed GPS progress.

Crucially, the color gradients (representing sequential progression) demonstrate that XTraj successfully learns the kinematic pacing of vehicles. Furthermore, the generated trajectories achieve Jaccard similarity scores at the road-segment level ranging from 0.50 to 1.00. It is important to highlight that while high scores demonstrate the model's ability to precisely fit human routing preferences, lower Jaccard similarity scores—coupled with the successful completion of Origin-Destination (OD) tasks—reflect the diversity of the generated trajectories. This indicates that the model has learned the underlying road network topology well enough to explore valid alternative paths rather than merely memorizing training samples. Regardless of the specific route chosen, the coarse-to-fine autoregressive mechanism effectively captures realistic routing decisions while perfectly confining the generated points within the valid road network boundaries.

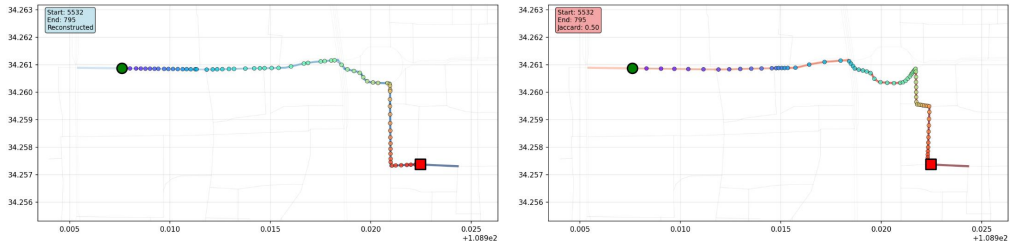


Figure 9: Single trajectory generation (Example 1). Left: Ground-truth reconstructed trajectory. Right: XTraj generated trajectory. The model successfully navigates the complex road network to complete the OD task. The road-segment Jaccard similarity of 0.50 reflects the model's capability to generate diverse and plausible alternative routes while strictly adhering to the underlying road geometry.

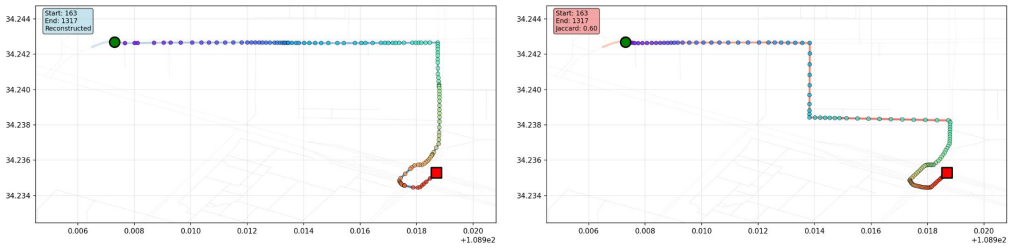


Figure 10: Single trajectory generation (Example 2). XTraj demonstrates the ability to generate variable-length, long-horizon trajectories (15 roads, 141 GPS points). The Jaccard similarity of 0.60 further confirms that the model can provide diverse yet effective path options for OD tasks while perfectly matching the kinematic pacing and topological constraints of the real trajectory.

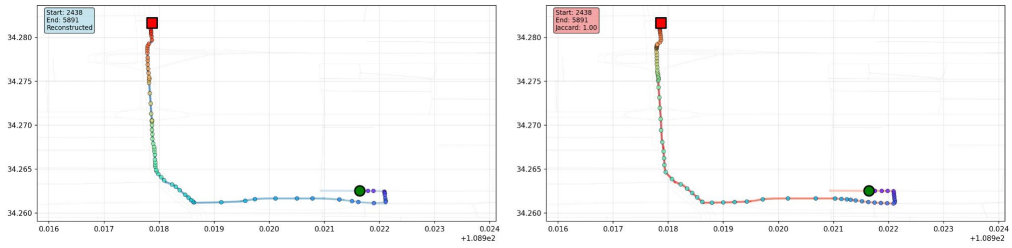


Figure 11: Single trajectory generation (Example 3). In this scenario, XTraj achieves a perfect Jaccard similarity of 1.00, meaning the generated route exactly matches the ground-truth human routing choice, with realistic fine-grained GPS progress variations.

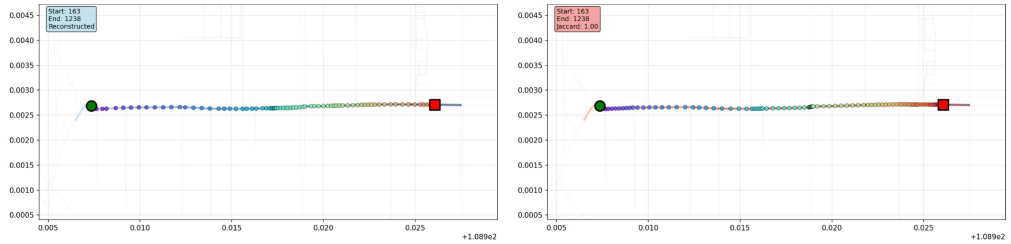


Figure 12: Single trajectory generation (Example 4). Another example demonstrating a perfect route match (Jaccard: 1.00) on a relatively straight path, highlighting XTraj's stability in standard urban navigation tasks.

D.2 Micro-level Analysis and Training-Free Generalization

To further substantiate the claimed topological robustness and precision, we conduct a micro-scale inspection of the generated trajectories. This fine-grained analysis serves to validate the model's adherence to underlying road networks at a localized level.

Micro-level Alignment and Topological Integrity. Figure 13 presents magnified views of representative intersections, offering empirical evidence of XTraj's generation fidelity. A persistent challenge in continuous-space generative models is the occurrence of *spatial hallucinations*—where generated coordinates drift into non-traversable regions. In contrast, both XTraj and its training-free variant (XTraj-Transfer) maintain strict topological validity. As visualized, the synthesized trajectory points align precisely with the road geometry, exhibiting zero "off-road" excursions.

This consistency underscores the efficacy of our **path hard constraint**. By embedding graph-based structural priors directly into the generative process, XTraj effectively eliminates error accumulation. Remarkably, this physical consistency is preserved even during training-free transfer to entirely unseen urban topologies (e.g., from Chengdu to Xi'an), demonstrating the superior generalization capabilities of our discrete-continuous hybrid approach.

D.3 Comprehensive Visual Comparison with Baselines

To provide a direct visual explanation for why diffusion-based baselines perform poorly on metrics like Post-hoc Map-Matching Rate (MMR) and Average Shift, we plot their generated outputs against XTraj.

Micro-level Comparison (Figure 14). When zoomed in, the structural flaws of coordinate-space diffusion models become blatantly apparent. **DiffTraj** (red) generates a massive amount of noise, with points scattered randomly across buildings and unnavigable spaces. While **ControlTraj** (orange) and **CarDiff** (blue) generate smoother curves, they frequently take "straight-line shortcuts" across blocks, violating real-world physical constraints. In stark contrast, **XTraj** (purple) strictly adheres to the road lanes, identical to the **Real** data (green).

Macro-level Comparison (Figure 15). At the city scale, **Diff-RNTraj** (brown) fails to capture the correct density, resulting in a chaotic, spider-web-like distribution. **DiffTraj** (red) exhibits severe grid-like artifacts, failing to understand the underlying urban topology. While **ControlTraj** and

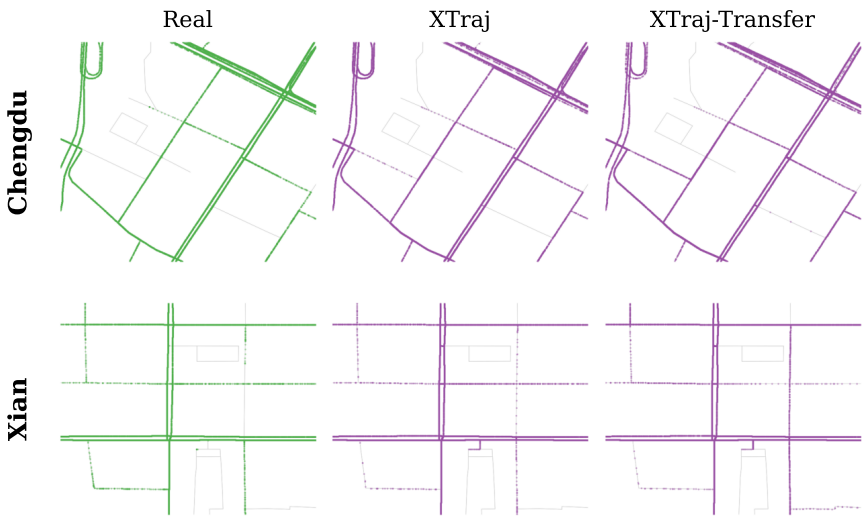


Figure 13: Micro-level trajectory alignment for XTraj and its training-free transfer variant. The visualizations highlight the model's ability to maintain perfect road-network adherence across different city scales and geometries.

CarDiff capture the rough outlines of major highways, they still produce fuzzy boundaries and noisy background artifacts. **XTraj** is the only model that consistently outputs clean, topologically accurate, and distributionally faithful traffic networks.

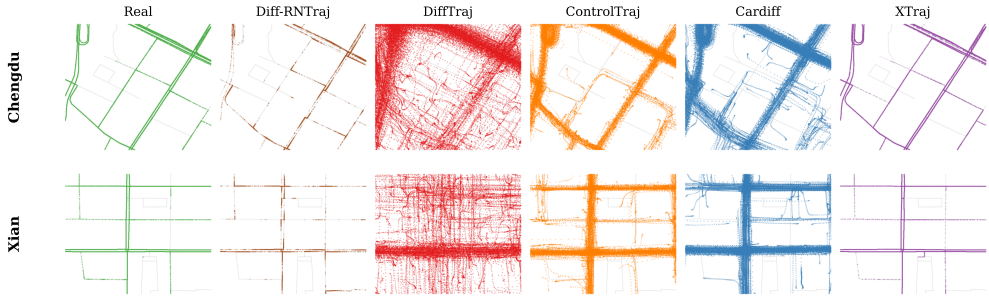


Figure 14: Micro-level qualitative comparison across all models. Coordinate-based baselines (DiffTraj, ControlTraj, CarDiff) suffer from severe spatial hallucinations, frequently generating trajectories that cross through buildings or empty spaces. XTraj (Ours) explicitly aligns generation with the road network, ensuring 100% physical validity.

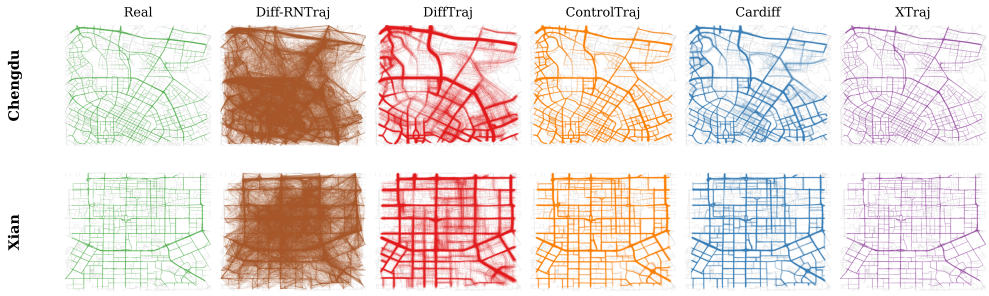


Figure 15: Macro-level city-wide distribution comparison. Baselines exhibit various failure modes: chaotic connectivity (Diff-RNTraj), grid artifacts and excessive noise (DiffTraj), and fuzzy boundaries (ControlTraj, CarDiff). XTraj (Ours) produces the cleanest and most accurate reproduction of the real traffic distribution.

E Quantitative Results of Training-free Transfer Performance

In this section, we present a quantitative analysis of the training-free transfer performance. As reported in Table 2, XTraj effectively captures invariant mobility patterns, and significantly outperforms all coordinate-based cross city transfer baselines.

To further qualitatively demonstrate the catastrophic performance drops of coordinate-based baselines under the training-free transfer setting, we visualize the generation outputs of ControlTraj and CarDiff when transferred to unseen cities. As illustrated in Figures 17 and 18, although these models generate plausible trajectories in their source cities, their performance degrades severely under transfer, producing trajectories that deviate substantially from the target road networks. In contrast, XTraj preserves macro-level traffic density distributions and road-network consistency across unseen cities, as shown in Figure 16.

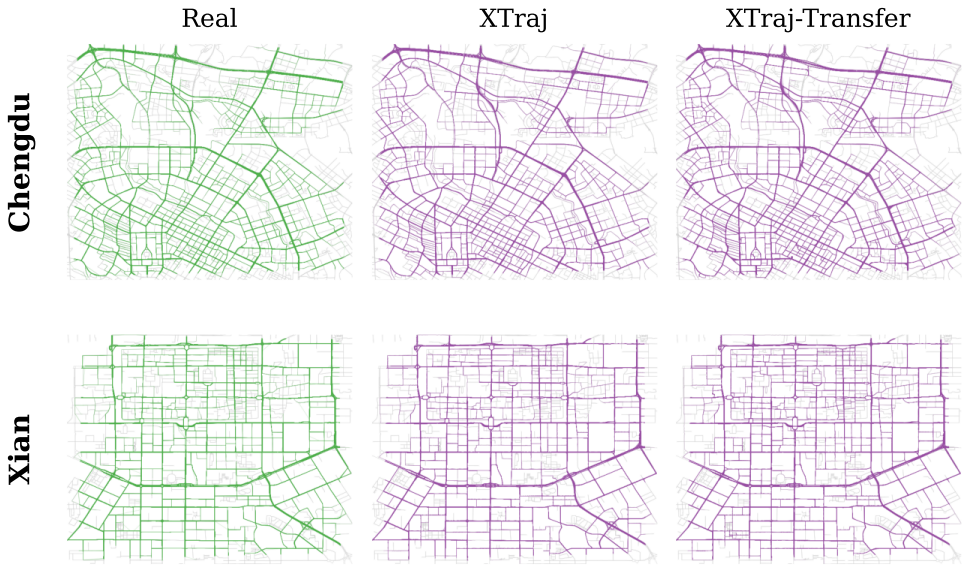


Figure 16: Visual comparison of training-free transfer performance for XTraj. From left to right: Real trajectories, XTraj (in-city training), and XTraj-Transfer (training-free cross-city transfer).

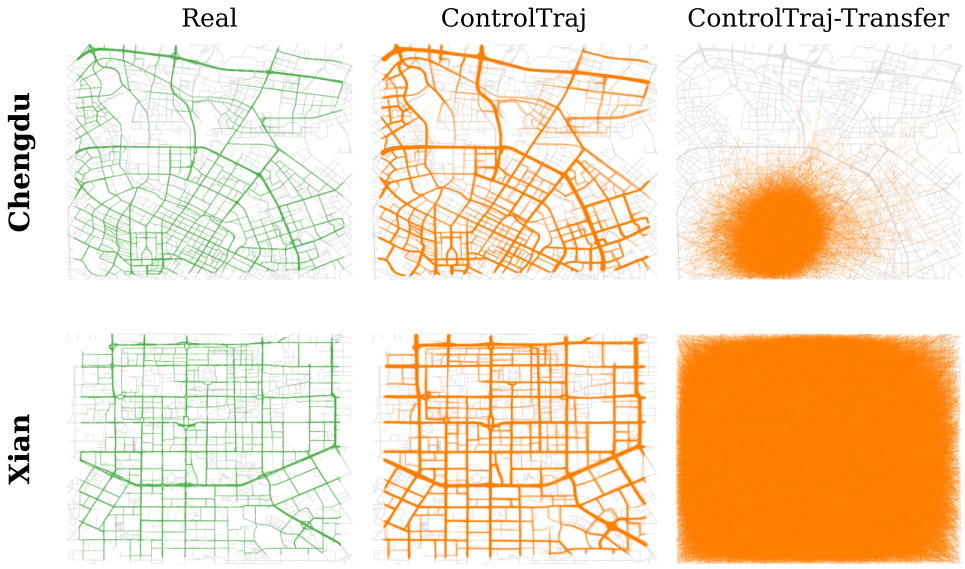


Figure 17: Visual comparison of training-free transfer performance for ControlTraj. The figure highlights the severe degradation and spatial hallucinations when the coordinate-based model is forced to generate trajectories in an unseen target city.

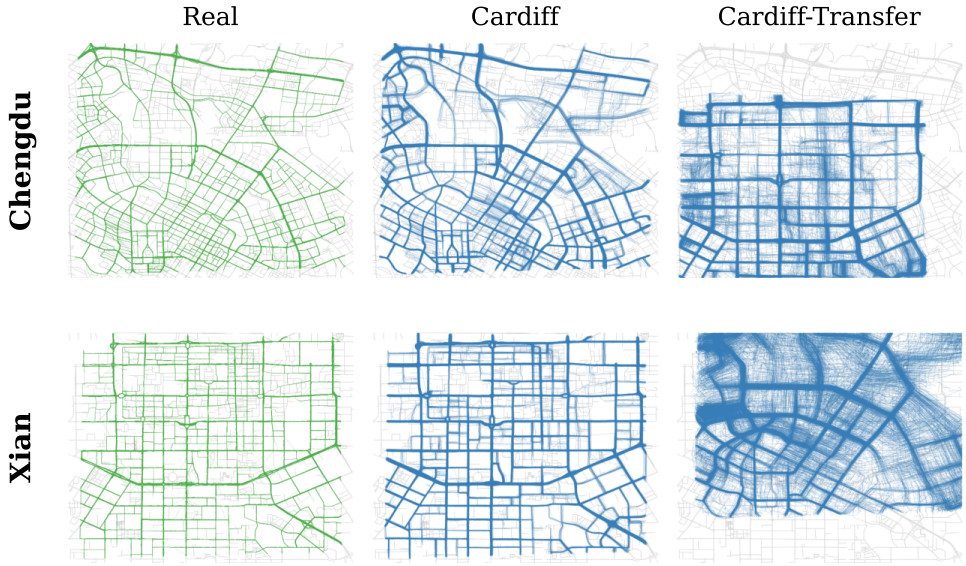


Figure 18: Visual comparison of training-free transfer performance for CarDiff. Similar to ControlTraj, CarDiff exhibits significant performance drops in the training-free transfer setting, generating off-road points and losing structural fidelity.

F Detailed Experiment Setups

In this section, we describe the experimental setup of this paper in detail, including the dataset, baseline implementation and evaluation metrics. All experiments are implemented with PyTorch 2.11 and Python 3.9 (conda environment). The three-stage training pipeline uses different hardware configurations: Stage 1 (Transferable Road-Segment Encoder) and Stage 2 (Road-Segment-Level Autoregressive Generator) are each conducted on a single NVIDIA RTX4090-24G GPU, while Stage 3 (Road-Aligned GPS Generator) is trained with Distributed Data Parallel (DDP) across 4 NVIDIA RTX4090-24G GPUs using the NCCL backend.

F.1 Porto Transfer

To evaluate transferability to an unseen city network, we apply XTraj models trained on Chengdu and Xi'an to Porto without additional parameter training or fine-tuning. Table 8 reports the results for spatial density, origin–destination, and travel-length distributions.

Table 8: Training-free transfer from Chengdu and Xi'an to Porto.

Source → target	Density	Trip	Length
Chengdu → Porto	0.0708	0.0031	0.0192
Xi'an → Porto	0.0714	0.0024	0.0407

XTraj achieves low divergence across all three metrics in both transfer settings. These results demonstrate that XTraj can transfer to the unseen Porto road network while preserving realistic spatial density, trip structure, and travel-length patterns, providing evidence that its generation capability extends beyond the cities used for training.

F.2 OD-Matched Route Diversity

To examine whether the destination-distance prior leads to shortest-path collapse, we compare XTraj with shortest-path and most-common-route baselines using an OD-matched route-choice analysis.

Table 9: OD-matched route-choice analysis.

Setting	Route gap	Routes/OD	Entropy	Dominance	Coverage	Edit distance
Ground truth	–	3.94	0.603	0.791	–	0.118
XTraj	0.064	3.64	0.497	0.821	0.951	0.089
w/o mask	0.076	8.10	0.630	0.805	0.949	0.118
Shortest path	0.393	1.00	0.000	1.000	0.381	0.000
Most-common route	0.083	1.00	0.000	1.000	0.791	0.000

XTraj achieves a smaller route-distribution gap than the shortest-path baseline, with values of 0.064 and 0.393, respectively. It remains close to ground truth in distinct routes per OD and route-choice entropy, and covers 95.1% of ground-truth route modes. Both deterministic baselines generate only one route per OD and have zero entropy.

XTraj generates diverse routes through stochastic decoding. For a fixed OD pair, it samples each next segment from the masked distribution over valid outgoing neighbors, allowing repeated generations to produce different plausible routes.

F.3 Temporal Realism

We evaluate temporal realism using Trip-time JSD and Speed JSD, which measure how closely generated trajectories reproduce real trip-time and speed distributions. Table 10 shows that XTraj achieves low divergence for both metrics in Chengdu and Xi'an, indicating that its generated trajectories preserve realistic travel durations and movement speeds.

Table 10: Temporal distribution similarity.

City	Trip-time JSD	Speed JSD
Chengdu	0.0185	0.0787
Xi'an	0.0154	0.0366

The baselines use fixed-length interpolation, which removes inter-point timing information and limits their ability to preserve the original temporal dynamics. In comparison, XTraj's low Trip-time JSD and Speed JSD provide evidence of temporal fidelity beyond spatial alignment.

F.4 Downstream Traffic-Flow Forecasting

We evaluate downstream utility through traffic-flow forecasting using STAEformer, STGformer, and STGCN trained with real or generated trajectories.

Traffic flow is computed at a five-minute temporal resolution. Seconds-within-day timestamps are converted into five-minute bins, and the number of real or generated trips passing through each road segment is counted within each bin.

The Chengdu data cover November 1–30, 2016, yielding 8,640 time steps. The forecasting task uses 12 historical steps to predict the next 12 steps, producing 8,617 supervised sliding-window samples. A chronological 60/20/20 split gives 5,170 training, 1,723 validation, and 1,724 test samples.

Table 11: Downstream traffic-flow forecasting using real and generated trajectories.

Model	Real RMSE	Generated RMSE	Real MAE	Generated MAE
STAEformer	1.77984	1.76453	1.24160	1.23371
STGformer	1.76075	1.77295	1.23136	1.23391
STGCN	1.81793	1.80152	1.32340	1.29401

Generated trajectories achieve comparable forecasting utility to real trajectories. STAEformer and STGCN improve with generated data, while STGformer shows a small degradation. These results suggest that XTraj preserves traffic-flow patterns useful for downstream forecasting in the evaluated Chengdu setting.

G Discussion

Model assumption. XTraj assumes that general road attributes and network topology are sufficient to generate realistic vehicle trajectories. Road segments are represented using geometry, POI distribution, and traffic heat, while GPS points are modeled as monotonic progress along connected segments. Trip-level conditions such as origin, destination, and departure time provide context for variable-length routes. Historical traffic patterns are assumed representative, enabling cross-city transfer without retraining. These assumptions allow the model to synthesize both coarse road-segment paths and fine-grained GPS trajectories consistently.

Distance-monotone assumption. The distance-monotone constraint is applied at the road-segment transition level: a candidate is allowed if its endpoint is closer to the destination. Curved roads and roundabouts remain valid when they make endpoint progress. However, the destination-distance prior may underrepresent temporary detours, including routes that use faster roads. Revisited segments, loops, or U-turns account for 3.13% of the uncleaned Chengdu data and 1.72% of the uncleaned Xi'an data. These trips are excluded from training and standard evaluation during preprocessing. We evaluate their filtered OD pairs separately.

Table 12: Evaluation on filtered non-monotone trips.

City	Length	Generated mean	Real mean
Chengdu	0.0189	3,808.1 m	3,937.8 m
Xi'an	0.0217	4,296.1 m	4,474.1 m

Generated routes are shorter than observed routes by 129.7 m in Chengdu and 178.0 m in Xi'an. This suggests that XTraj produces more destination-directed paths for these irregular trips and may underrepresent strong detours, loops, or U-turns.

Limitations. XTraj has several limitations. It requires complete and accurate road, POI, and traffic data, which may be unavailable in some regions. Autoregressive decoding, though constrained by topology and shortest-path rules, can propagate errors under unusual conditions. Transferability may degrade when road layouts or traffic distributions differ significantly from the training city. Broader evaluation across more cities remains future work.

Transfer and privacy scope. Training-free transfer performs no model training or fine-tuning on the target city but uses coarse target-city traffic statistics computed from one day of observations. XTraj does not provide formal differential-privacy guarantees. It samples neighbor-constrained routes from learned distributions rather than retrieving raw traces. However, synthetic trajectories could be misinterpreted as real data, and biases in POI or traffic inputs may propagate into outputs, affecting service equity. Overreliance on generated trajectories in autonomous systems or policy could lead to failures under unmodeled real-world scenarios.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly state that XTraj generates realistic, road-valid trajectories, supports variable-length outputs at both road- and GPS-levels, and enables cross-city transferability. These claims are empirically validated in the main experiments (road-level & GPS-level generation, cross-city transfer, and ablations)

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper discusses limitations about comprehesive data acquisition, unsupport cases.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: paper provides assumptions and proofs.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: All experimental procedures are described: datasets, preprocessing, training splits, hyperparameters, baselines, and evaluation metrics (road-level, GPS-level, cross-city transfer). Experimental Setup provides a more detailed information

Guidelines:

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

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

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

   Answer: [Yes]

   Justification: Code and anonymized data preparation scripts are provided in supplemental material; instructions enable reproducing generation results.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Training and test splits, optimizer, learning rates, epoch counts, batch sizes, and model architecture details are reported for each stage of HeatINCTraj. See Experimental Setup and Appendix.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Metrics include standard deviations for repeated runs; for distributions JSDs are reported to quantify variability

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Experiments report GPU type (NVIDIA RTX4090), number of GPUs, memory, training time per stage, and inference time per trajectory. See Appendix.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Research adheres to NeurIPS Code of Ethics; no unsafe data collection are involved.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The paper discusses positive societal impact (privacy-preserving synthetic trajectory generation, urban planning support) and potential negative impacts (misuse for surveillance), with mitigation strategies (controlled data release, anonymization).

Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: synthetic trajectory release is anonymized.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Trajectory Data are from DIDI Chuxing; Road networks from OpenStreetMap (ODbL license), POI features from AMap API; proper citation and license references included in Appendix.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: Newly released HeatINCTraj model code is documented with training details, license, and usage instructions.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: No human subjects or crowdsourced data were used.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: No human subjects research performed.

    Guidelines:

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

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

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: LLMs were not used for any core modeling. LLM is used for Mostly Drafting paper, Editing and Understanding technical concepts.

    Guidelines:

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