TRAJCFM: OBSERVATION-CONDITIONED FLOW MATCHING FOR UNIFIED TRAJECTORY MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Real-world trajectories combine irregular sampling, partial observability, and heterogeneous motion dynamics. Modeling them across tasks requires capturing local motion structure under different observation conditions. We present TrajCFM, an observation-conditioned flow matching framework for unified trajectory modeling. A DiT-style Transformer, conditioned on flow time and observed trajectory context, learns a velocity field that transports noise at unobserved positions toward target coordinates while preserving observed locations. This formulation brings completion and prediction under a shared objective and backbone. To capture motion at varying temporal scales, observation-aware adaptive patching builds variable-length segment representations from local motion cues and observation availability. Sparse mixture-of-experts routing selectively activates experts to model heterogeneous trajectory patterns. Pretraining with diverse observation masks learns representations that can also be adapted to trajectory classification and road-conditioned generation. Experiments on five real-world datasets demonstrate that TrajCFM achieves state-of-the-art performance in both trajectory completion and prediction using only three flow integration steps. Downstream experiments on transport-mode classification across GeoLife and Grab-Posisi, together with road-conditioned generation, demonstrate the transferability of the learned representations.

1 INTRODUCTION

Trajectory data support transportation analysis, location-based services, and urban computing, with tasks ranging from completion and prediction to generation and classification (Zheng, 2015; Chen et al., 2024b). These tasks draw on common motion structure, making a shared model attractive for learning reusable representations rather than separate task-specific pipelines. Yet the available evidence differs: completion can use observations on either side of a gap, whereas prediction must extrapolate from a prefix. Irregular sampling and alternating stretches of steady motion, turns, and stops further change the context needed at each location. Unified trajectory modeling must accommodate both the observation pattern and this local heterogeneity.

Recent models have made substantial progress toward this goal. UniTraj learns a transferable backbone through large-scale masked pretraining, GenMove uses masked conditional diffusion to accommodate multiple mobility tasks, and UniMove shares spatial representations and expert-based mobility modeling across cities (Zhu et al., 2025; Long et al., 2025; Han et al., 2025b). These studies establish the value of sharing models across tasks or regions. We focus on how a shared coordinate model can reconstruct different missing patterns while adapting its local context and computation to the observations that remain.

We treat completion and prediction as the same conditional reconstruction problem with different observation masks. Building on flow matching (Lipman et al., 2023; Liu et al., 2023), we learn a velocity field that transports noise toward target coordinates at missing positions while keeping observed coordinates fixed. Trajectory flow models have addressed human forecasting and pseudo-GPS generation (Fu et al., 2025; Li et al., 2026). For shared reconstruction, the central design question is how the velocity network represents the remaining observations. TrajCFM combines point-level states with observation-dependent segment context in one network, trained for both dispersed gaps and an unobserved suffix.

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

We build local structure into this velocity network with two complementary design choices (Figure 1). Fixed-length patches impose the same granularity throughout a trajectory: a boundary may split a turn or speed transition, while a long steady stretch may receive unnecessarily fine segmentation. This motivates observation-aware adaptive patching, which organizes local observation context in response to motion cues and observation availability. Point tokens remain in the network to preserve coordinate-level resolution. Sparse mixture-of-experts (MoE) layers then route point and patch tokens through input-dependent feed-forward transformations. The two designs address which neighboring observations to summarize together and which computation to apply to their representations, within the same observation-conditioned Transformer.

On WorldTrace, four independently trained models give completion MAE of 5.75 ± 0.05 m and prediction MAE of 31.75 ± 0.25 m. A four-variant ablation examines the contribution of each component, while motion-group analyses and routing visualizations characterize model behavior. Beyond reconstruction, the shared representation supports road-conditioned generation through a ControlTraj adapter and trajectory classification through a lightweight task head.

Our main contributions are as follows:

- An observation-conditioned trajectory transport model that shares its velocity network and objective across completion and prediction, preserves observed coordinates during inference, and supports generation and classification through representation adaptation.
- Observation-aware adaptive patching that constructs variable-length segments from local motion cues and available observations, introducing segment-level context into the velocity field while retaining point-level resolution.
- Sparse expert routing within the conditional Transformer that selects feed-forward transformations from point and patch context, providing token-dependent computation for heterogeneous trajectory patterns.
- An evaluation of completion and prediction on five datasets, classification on two datasets, and road-conditioned generation. On WorldTrace, the combined design reduces completion and prediction MAE by 14.46% and 5.26% relative to Plain DiT; four-seed evaluation and targeted controls assess stability and the two architectural choices.

2 Related Work

Unified trajectory models. Shared trajectory models pursue both task reuse and regional transfer. UniTraj learns a reusable backbone through large-scale resampling and masked pretraining on WorldTrace (Zhu et al., 2025). TrajFM combines spatial, temporal, and POI modalities with a masking-and-recovery scheme for region and task transfer (Lin et al., 2024). GenMove brings multiple mobility tasks into mask-conditioned diffusion with historical context (Long et al., 2025), while UniMove uses trajectory and location towers for multi-city mobility prediction (Han et al., 2025b). Our focus is continuous-coordinate reconstruction from the currently observed trajectory, with an observation-dependent representation of local context.

Generative trajectory modeling. Diffusion models support both conditional reconstruction and generation. CSDI reconstructs missing time-series values from observed entries (Tashiro et al., 2021), whereas ControlTraj generates trajectories conditioned on road information (Zhu et al., 2024). MoFlow applies conditional flow matching to multimodal human forecasting and uses distillation for one-step sampling (Fu et al., 2025); TrajFlow studies pseudo-GPS generation at urban, metropolitan, and nationwide scales (Li et al., 2026). These applications establish flow matching for forecasting and generation. TrajCFM extends the reconstruction interface to both dispersed observations and prefixes, and couples the velocity field to variable-length observation context.

Adaptive context and conditional computation. PatchTST uses fixed temporal patches (Nie et al., 2023); Pathformer routes inputs through multiple patch scales (Chen et al., 2024a); PatchTraj combines dynamic patch partitioning with time–frequency representations for pedestrian prediction (Liu et al., 2025). These designs vary the granularity of context. Sparse experts vary the transformation applied to a representation: Time-MoE studies sparse expert models for time series, while MoveGPT and UniMove apply expert-based modeling to mobility (Shi et al., 2025; Han et al., 2025a;b).

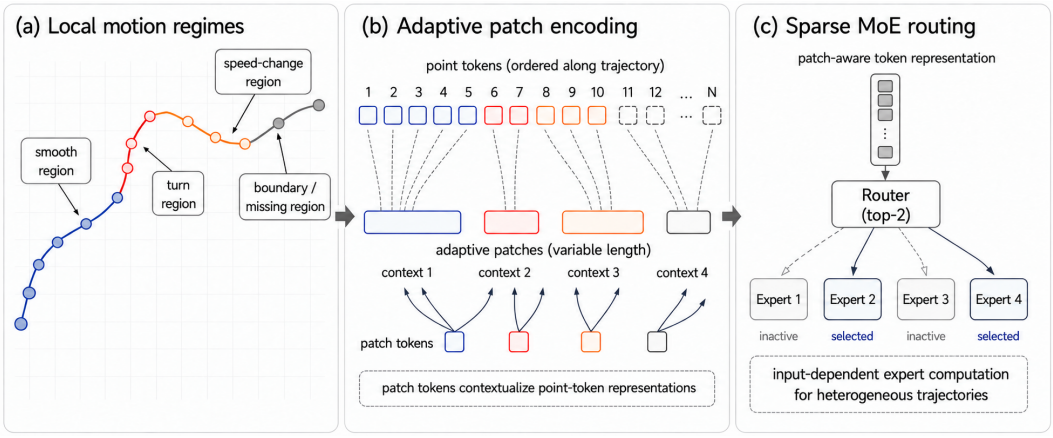


Figure 1: Adaptive patch encoding and sparse expert routing in TrajCFM. (a) A schematic trajectory with varying motion and observation patterns. (b) Variable-length patch tokens organize observation context alongside point tokens; colors distinguish schematic local contexts. (c) A top-2 router provides input-dependent expert computation for heterogeneous trajectories.

TrajCFM combines observation-conditioned patch construction with token-dependent expert routing inside a coordinate velocity network.

3 METHOD

Overview. TrajCFM represents a partially observed trajectory with point tokens and adaptive patch tokens (Figure 1). Point tokens preserve coordinate-level resolution; patch tokens summarize variable-length segments of the observation condition. A DiT-style Transformer processes both through self-attention and sparse expert layers. Its point-token head predicts velocities for updating the missing coordinates.

3.1 OBSERVATION-CONDITIONED FLOW MATCHING

Let $X = (x_1, \ldots, x_T) \in \mathbb{R}^{T\times 2}$ be a normalized coordinate sequence with zero padding, and $\Delta = (\Delta_1, \ldots, \Delta_T)$ its sampling intervals. A binary validity mask a_i identifies nonpadding positions, and an observation mask $m_i \le a_i$ identifies the available coordinates. The target mask is $u_i = a_i - m_i$. The fixed condition is $c = (C, \Delta, m, a)$, where $C = m \odot X$ contains zeros at unobserved positions. Completion masks positions within the trajectory; prediction masks a suffix.

For Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$ and flow time $\tau \sim \mathcal{U}[0, 1]$, construct

$$X_0 = C + u \odot \epsilon, \qquad X_\tau = (1-\tau)X_0 + \tau X, \qquad U = X - X_0. \tag{1}$$

Masks broadcast over the two coordinate dimensions. The path preserves observed coordinates and changes only valid target positions. A conditional network $v_\theta(X_\tau, \tau; c)$ predicts its velocity; completion and prediction use the same network and objective with different observation masks. We train with target-position L1 regression,

$$\mathcal{L}_{\text{vel}} = \mathbb{E}\left[\frac{\sum_{b,i} u_{bi} \| v_\theta(X_{\tau_b}, \tau_b; c_b)_i - U_{bi} \|_1}{\sum_{b,i} u_{bi}}\right], \tag{2}$$

where b indexes trajectories in a training batch and valid target points receive equal weight. We adopt L1 velocity regression because it gave the best reconstruction accuracy among the objectives evaluated in preliminary experiments. Appendix B discusses its relationship to the squared-error FM objective.

3.2 Observation-Aware Adaptive Patching

Point conditioning. The observation condition supplies a point embedding c_i, which is combined with the current flow state to form a point token:

$$c_i = W_c C_i + W_\Delta \Delta_i + W_m[m_i, a_i], \qquad h_i^{(0)} = W_x(X_\tau)_i + c_i + e_i, \tag{3}$$

where affine maps embed coordinates, intervals, and masks, and e_i is a learned position embedding. A global condition modulates the Transformer through adaptive layer normalization (Peebles & Xie, 2023):

$$q = \mathrm{TimeEmbed}(\tau) + g\left(\frac{\sum_i a_i c_i}{\max(1, \sum_i a_i)}\right). \tag{4}$$

Pooling includes every valid position, with mask embeddings distinguishing observed and hidden coordinates. The condition embeddings are fixed for a given input, whereas the state embeddings and flow-time component change during integration.

Point and patch tokens play different roles. Each point token tracks a coordinate being reconstructed; a patch token summarizes several positions in the observation condition. Keeping both lets attention combine precise point locations with a coarser local context. In completion, this context can contain observations on both sides of a gap. In prediction, it summarizes the available prefix while the point states evolve at the unobserved suffix.

Adaptive segmentation. We form variable-length segments from the fixed observation condition. A convolutional scorer produces $\ell_i \in [0, 1]$, while a rule score r_i summarizes local motion and mask-transition cues derived from C, Δ, and m. Their fusion is

$$s_i = a_i \,\mathrm{clip}\big(\ell_i + \lambda_r(r_i - \bar{r}), 0, 1\big), \qquad \bar{r} = \frac{\sum_i a_i r_i}{\max(1, \sum_i a_i)}. \tag{5}$$

Here λ_r controls the rule contribution. Starting a segment at j, we select the first eligible boundary i satisfying

$$i - j + 1 \geq L_{\min}, \qquad s_i \geq \theta, \qquad s_i \geq \frac{1}{i - j + 1} \sum_{k=j}^{i} s_k + \delta. \tag{6}$$

The threshold θ selects high-scoring positions, and the margin δ requires a rise above the segment's average score. Otherwise, the segment closes at $L_{\max}$ or the sequence end. Scores use only the masked observation condition and are reused across Euler evaluations. Appendix A provides feature normalization, smoothing, terminal-segment handling, and the complete algorithm.

Patch representation. A separate MLP embeds $[C_i, \Delta_i, m_i]$ into f_i. For patch P_j, initialize

$$p_j^{(0)} = \frac{\sum_{i \in P_j} a_i s_i f_i}{\max(10^{-6}, \sum_{i \in P_j} a_i s_i)} + \mathrm{LengthEmbed}(|P_j|). \tag{7}$$

Two cross-attention blocks refine each token using the point features within its patch, with residual feed-forward updates. The resulting patch tokens summarize the fixed observation condition. Gradients reach the scorer through the pooling weights and score regularizer; the discrete boundary decisions are not differentiated.

3.3 Sparse-Expert Trajectory Transformer

The backbone receives $[p_1, \ldots, p_J; h_1^{(0)}, \ldots, h_T^{(0)}]$ with a combined validity mask. Its six DiT-style blocks use q for adaptive layer normalization. Within each block, self-attention first exchanges information across point and patch tokens; a sparse MoE (Shazeer et al., 2017) then replaces the dense feed-forward transformation. For a contextualized token h_i, a linear router selects two of four experts:

$$\mathcal{K}_i = \mathrm{Top2}(W_r h_i), \quad \alpha_{i,e} = \frac{\exp((W_r h_i)_e)}{\sum_{e' \in \mathcal{K}_i} \exp((W_r h_i)_{e'})}, \quad \mathrm{MoE}(h_i) = \sum_{e \in \mathcal{K}_i} \alpha_{i,e} E_e(h_i). \tag{8}$$

The selected experts' logits are normalized within the active pair. Each expert is a two-layer feed-forward network, and each block has its own router and experts. After the final block, a condition-modulated linear head maps only the point tokens to two-dimensional velocities. For the Plain DiT comparison, we omit the patch branch and use dense feed-forward layers, retaining the point conditioning, self-attention, and output head.

3.4 Training and Inference

Training. We sample completion and prediction conditions with probabilities 0.7 and 0.3. Completion uses random, shape-aware RDP (Ramer, 1972; Douglas & Peucker, 1973), and block-plus-random masks with probabilities 0.6, 0.2, and 0.2; prediction hides the last five points. Combining velocity regression with a patch-score diversity term gives the complete loss:

$$\mathcal{L} = \mathcal{L}_{\text{vel}} - \frac{0.01}{B} \sum_b \frac{\sum_i a_{bi}(s_{bi} - \bar{s}_b)^2}{\max(1, \sum_i a_{bi})}, \qquad \bar{s}_b = \frac{\sum_i a_{bi} s_{bi}}{\max(1, \sum_i a_{bi})}. \tag{9}$$

Here B is the batch size. The second term encourages variation in patch scores.

Sampling. Inference starts from $X^{(0)} = C + u \odot \epsilon$. With $S = 3$ Euler evaluations and $\tau_k = k/S$, update

$$X^{(k+1)} = C + u \odot \left[X^{(k)} + \frac{1}{S} v_\theta(X^{(k)}, \tau_k; c) \right]. \tag{10}$$

Each update restores observed coordinates and zeros padding. Completion and prediction use this same sampler; the observation mask determines the target positions. The three-step budget is selected on validation data (Section 4.4).

Downstream adaptation. For classification, we fine-tune the backbone with a supervised task head. For road-conditioned generation, an adapter maps frozen trajectory features to the condition interface of ControlTraj, which performs diffusion sampling.

4 Experiments

We evaluate reconstruction accuracy, architectural contributions, and computational cost, followed by transfer to generation and classification.

4.1 Experimental Setup

Data and tasks. We study completion and prediction on WorldTrace (Zhu et al., 2025), Chengdu and Xi'an from the JCLRNT data release (Mao et al., 2022), GeoLife (Zheng et al., 2010b), and Porto (UCI Machine Learning Repository, 2015). Completion reconstructs 50% of valid points hidden at random; prediction reconstructs the final five points from the observed prefix. Within each dataset, both tasks use the same checkpoint, differing only in the observation mask. WorldTrace uses a deterministic 90/10 training–validation split and the original test set of 245,129 trajectories. Checkpoints and integration steps are selected on validation data. Reconstruction protocol details are given in Appendix C; downstream task settings are introduced with their results.

Baselines and metrics. The completion and prediction comparisons include Linear, DHTR (Wang et al., 2021), Transformer (Vaswani et al., 2017), DeepMove (Feng et al., 2018), TrajBERT (Si et al., 2024), TrajFM (Lin et al., 2024), and UniTraj (Zhu et al., 2025), spanning interpolation, task-specific models, and pretrained trajectory models. All methods use the same input observations, reconstruction targets, and evaluation pipeline. Each trainable method is trained or fine-tuned on the target training split, with hyperparameters and checkpoints selected using validation data only. DeepMove is adapted to continuous-coordinate reconstruction rather than discrete location prediction. We measure MAE and RMSE in meters using geodesic point errors. Both metrics are computed within each trajectory and then averaged across trajectories, giving each eligible trajectory equal weight. The same error definitions apply to prediction.

Implementation details. TrajCFM is implemented in PyTorch and trained on NVIDIA RTX 4090 GPUs with CUDA. WorldTrace architectural comparisons use two GPUs with a global batch size of 384 (192 per GPU) and seed 42. TrajCFM results in Tables 1 and 2 are reported as means over four independently trained seeds on each dataset, with sample standard deviations shown in subscripts. WorldTrace uses seeds 0, 1, 2, and 42. All TrajCFM results in Tables 1 and 2 use three Euler evaluations, selected by the joint validation criterion in Section 4.4. The architecture and training objective follow Section 3, with adaptive-patch settings detailed in Appendix A.

Processed trajectories contain at most 200 points. Coordinates are centered at the first observed point and transformed using the checkpoint's fixed normalization; sampling intervals and validity masks accompany the coordinates. WorldTrace training uses AdamW with initial learning rate 10^{-4}, weight decay 10^{-4}, and gradient clipping at 1.0. A validation-loss plateau reduces the learning rate, and the lowest-loss checkpoint is retained. Cross-dataset fine-tuning starts from the WorldTrace backbone and updates the model on the target training split. Appendix Table 6 summarizes the shared architecture and WorldTrace training configuration.

4.2 Trajectory Completion and Prediction

Completion. TrajCFM achieves the lowest MAE and RMSE across all five datasets (Table 1). Against each dataset's strongest baseline, completion MAE falls by 43.76% on WorldTrace, 2.05% on Chengdu, 8.03% on Xi'an, 38.46% on GeoLife, and 1.52% on Porto.

Prediction. The same model predicts an unobserved suffix by changing the observation mask, without a task-specific head. TrajCFM achieves the lowest MAE and RMSE on all five datasets (Table 2), including 31.75 m MAE on WorldTrace and 189.36 m on Porto.

Table 1: Trajectory completion on five datasets with 50% of valid points hidden at random. MAE and RMSE are in meters; lower is better. Bold and underline mark the best and second-best results. TrajCFM values are four-seed means with sample standard deviations in subscripts.

Methods	Trajectory Completion									
	WorldTrace		Chengdu		Xi'an		GeoLife		Porto	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Linear	427.68	516.15	205.74	258.52	176.49	220.87	196.85	249.76	396.61	482.39
DHTR	220.35	302.47	75.19	98.68	62.85	83.43	80.04	168.25	194.37	232.59
Transformer	130.82	147.62	55.23	62.85	45.85	51.96	94.68	113.77	104.36	126.96
DeepMove	51.16	62.29	29.32	39.02	27.31	35.67	86.38	107.78	136.66	174.96
TrajBERT	58.13	70.14	26.48	33.83	19.45	25.13	<u>34.53</u>	<u>43.24</u>	78.77	99.23
TrajFM	47.64	58.92	19.10	25.09	18.86	24.13	59.34	64.24	<u>71.15</u>	<u>92.96</u>
UniTraj	<u>10.22</u>	<u>13.56</u>	<u>11.98</u>	<u>20.94</u>	<u>8.93</u>	<u>13.83</u>	37.21	63.89	78.28	100.14
TrajCFM	**5.75**$_{\pm 0.05}$	**8.69**$_{\pm 0.07}$	**11.73**$_{\pm 0.11}$	**20.30**$_{\pm 0.17}$	**8.21**$_{\pm 0.07}$	**13.21**$_{\pm 0.10}$	**21.25**$_{\pm 0.19}$	**29.83**$_{\pm 0.23}$	**70.07**$_{\pm 0.58}$	**91.25**$_{\pm 0.76}$

Table 2: Trajectory prediction on five datasets, reconstructing the final five points from the observed prefix. Errors are in meters; lower is better. Bold and underline mark the best and second-best results. TrajCFM values are four-seed means with sample standard deviations in subscripts.

Methods	Trajectory Prediction									
	WorldTrace		Chengdu		Xi'an		GeoLife		Porto	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Linear	153.12	159.65	156.85	164.58	286.94	346.28	189.02	201.34	432.91	530.04
DHTR	146.48	151.63	123.47	129.73	228.16	270.83	180.32	187.59	346.28	420.32
Transformer	114.25	117.07	67.38	70.86	122.91	149.62	165.02	170.84	232.87	285.56
DeepMove	55.69	58.67	<u>36.31</u>	<u>39.10</u>	79.14	95.63	116.46	123.20	236.12	298.24
TrajBERT	80.57	86.36	64.73	68.92	119.38	143.87	113.68	<u>121.18</u>	307.46	393.08
TrajFM	75.45	81.32	77.82	80.48	141.76	169.42	121.94	128.16	<u>197.83</u>	<u>253.19</u>
UniTraj	<u>49.85</u>	<u>55.02</u>	42.75	45.93	<u>76.57</u>	<u>93.23</u>	<u>108.35</u>	133.60	206.54	266.37
TrajCFM	**31.75**$_{\pm 0.25}$	**37.49**$_{\pm 0.29}$	**35.33**$_{\pm 0.31}$	**38.67**$_{\pm 0.29}$	**66.36**$_{\pm 0.58}$	**83.71**$_{\pm 0.65}$	**56.93**$_{\pm 0.47}$	**63.44**$_{\pm 0.44}$	**189.36**$_{\pm 1.66}$	**242.88**$_{\pm 1.85}$

Stability. Across four independent WorldTrace training seeds, completion and prediction MAE have standard deviations of 0.05 m and 0.25 m, respectively, both below 1% of their means. The seed-42 model used for the architectural comparisons is close to the four-seed mean on all four metrics; individual runs are reported in Appendix Table 8.

4.3 Ablation and Routing Analysis

We compare four full-data WorldTrace variants under the same test protocol (Table 3). Plain DiT uses point tokens and dense feed-forward layers; AP-only adds the patch branch; MoE-only replaces the dense layers with sparse experts; Full combines both designs. Implementation details for the dense variants are given in Appendix D.

Table 3: Full-data WorldTrace ablation of adaptive patching (AP) and sparse experts (MoE), using seed 42 and three Euler evaluations. Checkmarks and crosses indicate enabled and disabled components. Errors are in meters; bold marks the best results.

			Completion		Prediction	
Model	AP	MoE	MAE	RMSE	MAE	RMSE
Plain DiT	×	×	6.67	10.32	33.58	39.50
AP-only	✓	×	6.30	10.10	33.24	39.19
MoE-only	×	✓	6.08	9.89	32.81	38.62
Full	✓	✓	**5.71**	**8.64**	**31.81**	**37.56**

Adaptive patches. Adding patch context improves both MAE and RMSE with either dense or sparse feed-forward layers. In the sparse model, AP reduces completion and prediction MAE by 6.07% and 3.03%, respectively.

Sparse experts. MoE improves both tasks with and without the patch branch. Relative to AP-only, Full reduces completion and prediction MAE by 9.33% and 4.30%. Combining the two components gives the lowest error on all four metrics, with MAE reductions of 14.46% for completion and 5.26% for prediction over Plain DiT.

With adaptive patching and sparse experts retained, directly regressing coordinates yields higher completion and prediction errors than TrajCFM (Appendix Table 10), supporting the flow-matching formulation in this comparison.

Granularity and capacity controls. Two targeted controls examine patch granularity and parameter allocation (Appendix Table 9). Fixed16+MoE retains the scorer, weighted pooling, patch refinement, and sparse experts, but uses segments of 16 points. Adaptive segmentation lowers completion MAE from 6.10 to 5.71 m and prediction MAE from 32.64 to 31.81 m. The parameter-matched Dense model retains AP and replaces sparse experts with dense layers with total parameter counts matched within 0.06%. Its MAEs are 5.94 and 32.34 m, compared with 5.71 and 31.81 m for Full. Both controls also have higher RMSE.

Routing behavior. Figure 2 shows layer-dependent expert usage, with additional preferences across motion groups within a layer. In a separate 5,000-trajectory validation diagnostic, Full improves over the AP-only reference in every evaluated group for both tasks; the largest absolute prediction gain occurs in steady straight motion (Appendix Table 13). Appendix F defines the checkpoint pair, ground-truth motion descriptors, and training-derived group thresholds.

The groupwise results show that the benefit of sparse computation extends beyond turns. Prediction MAE falls by 2.41 m for steady straight motion, 1.26 m for low-speed points, 1.02 m for larger turns, and 1.35 m for speed changes. Completion gains span 0.29–0.41 m across the same groups. Together with the layer-dependent routing patterns, this suggests a distributed role for the experts: their contribution depends on the current representation and network depth, rather than a fixed assignment of one expert to each motion category.

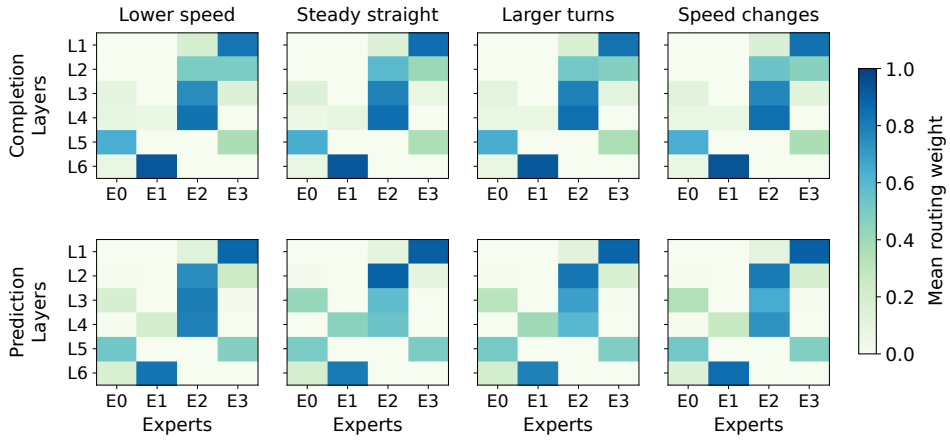


Figure 2: Expert routing at the third Euler evaluation on 2,000 validation trajectories, for completion (top) and prediction (bottom). Colors show mean normalized top-2 weights over unobserved point tokens, including zeros for unselected experts. Rows indicate layers and columns indicate layer-local experts; all panels share a color scale.

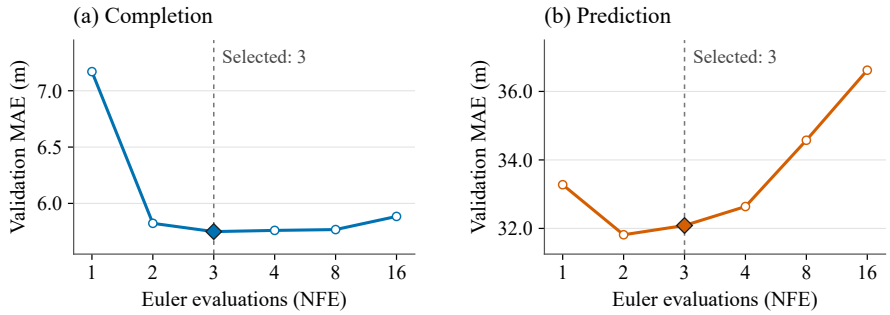


Figure 3: Validation MAE versus Euler evaluations for completion and prediction with a fixed checkpoint. Diamonds and dashed lines mark the shared three-step budget, selected by averaging four normalized validation errors. NFE denotes velocity-network evaluations.

4.4 Sampling Budget and Computational Efficiency

Three-step inference. Three Euler evaluations minimize the mean of the four validation errors normalized by their respective sweep minima (Figure 3). Completion MAE is 5.75 m at three evaluations, compared with 7.17 m at one and 5.88 m at sixteen. Prediction favors two evaluations, at 31.81 m, versus 32.09 m at three, and deteriorates with further integration. The shared three-step budget balances the two tasks: finer numerical discretization does not necessarily improve reconstruction by the learned velocity field. Appendix E gives the full sweep.

Training efficiency. Packing valid tokens before expert computation reduces the cost of padding. In paired 1,000-update runs of Full on a 20,000-trajectory WorldTrace subset, packed routing lowers median update time by 5.11% and peak allocated memory by 15.87% (Appendix Table 15). Both runs start from the same checkpoint and use two RTX 4090 GPUs with global batch 384; timing excludes the first 20 updates.

4.5 Road-Conditioned Generation

Following UniTraj's representation-transfer setting, we use frozen TrajCFM features as road conditions for ControlTraj and train the adapter and diffusion generator on Chengdu routes (Zhu et al., 2025; 2024). The reference uses ControlTraj's native RoadMAE encoder. Both systems are evaluated

on 512 independent sessions with ten paired noise draws, sharing routes, sampling schedule, and spatial grid. We report density, trajectory-length, and step-distance JSDs, together with median road distance and the fraction of points more than 100 m from the conditioning road. Preprocessing, adaptation, and metric definitions are detailed in Appendix G.

Table 4: Road-conditioned generation on 512 independent Chengdu routes with ten paired noise draws. ControlTraj uses its native road encoder; TrajCFM supplies adapted features to the same generator architecture. JSD is in bits, Road P50 is median road distance, and off-road denotes points beyond 100 m. Lower is better.

Method	Density JSD	Length JSD	Step JSD	Road P50 (m)	Off-road (%)
ControlTraj	0.0020	0.0057	0.0043	9.84	0.39
TrajCFM	**0.0018**	**0.0056**	**0.0039**	**9.03**	**0.31**

TrajCFM improves all five generation metrics over ControlTraj (Table 4). Density JSD falls by 9.80% using unrounded values, median road distance decreases from 9.84 to 9.03 m, and off-road rate drops from 0.39% to 0.31%. Length and step-distance JSDs also decrease.

4.6 Trajectory Classification

We evaluate classification on GeoLife (Zheng et al., 2010a) and Grab-Posisi (Huang et al., 2019) using accuracy and macro-F1 (Table 5). The baselines include TrajFormer (Liang et al., 2022) and STGN (Zhao et al., 2019), with STGN adapted to trajectory-level classification. TrajCFM and UniTraj are fully fine-tuned on both datasets; other baselines are trained from scratch. GeoLife uses a user-disjoint split, and Grab-Posisi uses a trajectory-disjoint split. Checkpoints are selected by validation macro-F1. Appendix H provides further details.

Table 5: Classification on GeoLife and Grab-Posisi. TrajCFM and UniTraj are fully fine-tuned on both datasets. GeoLife uses a user-disjoint split and seed 42. Accuracy is in percent; macro-F1 averages across classes. Higher is better; bold and underline mark the best and second-best results.

	GeoLife		Grab-Posisi	
Method	Accuracy (%)	Macro-F1	Accuracy (%)	Macro-F1
GRU	72.33	0.621	63.54	0.548
LSTM	67.19	0.607	69.17	0.621
STGN	72.42	0.673	72.32	0.675
TrajFormer	73.19	0.660	76.36	0.686
UniTraj	<u>74.46</u>	<u>0.681</u>	<u>79.28</u>	<u>0.728</u>
TrajCFM	**75.79**	**0.719**	**79.41**	**0.731**

TrajCFM ranks first on both metrics across the two datasets. On GeoLife, it achieves 75.79% accuracy and 0.719 macro-F1, exceeding UniTraj by 1.33 percentage points and 0.038, respectively.

On Grab-Posisi, TrajCFM reaches 79.41% accuracy and 0.731 macro-F1, with smaller gains over UniTraj of 0.13 percentage points and 0.003. Compared with TrajFormer, the gains are 3.05 percentage points and 0.045 macro-F1. The advantage over UniTraj is more pronounced on GeoLife.

5 Conclusion

We presented TrajCFM, an observation-conditioned flow matching framework combining adaptive patches and sparse experts for trajectory completion and prediction. Experiments demonstrate strong reconstruction accuracy with three integration steps and gains from both components. The pretrained backbone also supports transport-mode classification and road-conditioned generation through task-specific adaptation.

AI USE STATEMENT

Generative AI tools assisted with writing and editing, literature search and discovery, and research ideation and execution. The authors reviewed the AI-assisted work and take responsibility for the final manuscript and results.

REPRODUCIBILITY STATEMENT

Section 3 defines the model and objective. Appendix A gives the complete adaptive-patch algorithm (Algorithm 1). Section 4 describes the datasets, metrics and evaluation settings; the appendices provide supplementary implementation and protocol details.

REFERENCES

Peng Chen, Yingying Zhang, Yunyao Cheng, Yang Shu, Yihang Wang, Qingsong Wen, Bin Yang, and Chenjuan Guo. Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. In *International Conference on Learning Representations*, 2024a. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/2be6705de7412adf107900add727a795-Abstract-Conference.html.

Wei Chen, Yuanshao Zhu, Yanchuan Chang, Kang Luo, Haomin Wen, Lei Li, Yanwei Yu, Qingsong Wen, Chao Chen, Kai Zheng, Yunjun Gao, Yu Zheng, Xiaofang Zhou, and Yuxuan Liang. Trajectory data management and mining: A survey from deep learning to the LLM era. *arXiv preprint arXiv:2403.14151*, 2024b. URL https://arxiv.org/abs/2403.14151.

David H. Douglas and Thomas K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2):112–122, 1973.

Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. DeepMove: Predicting human mobility with attentional recurrent networks. In *Proceedings of the 2018 World Wide Web Conference*, pp. 1459–1468, 2018.

Yuxiang Fu, Qi Yan, Lele Wang, Ke Li, and Renjie Liao. MoFlow: One-step flow matching for human trajectory forecasting via implicit maximum likelihood estimation based distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2503.09950.

Chonghua Han, Yuan Yuan, Jingtao Ding, Jie Feng, Fanjin Meng, and Yong Li. MoveGPT: Scaling mobility foundation models with spatially-aware mixture of experts. *arXiv preprint arXiv:2505.18670*, 2025a. URL https://arxiv.org/abs/2505.18670.

Chonghua Han, Yuan Yuan, Yukun Liu, Jingtao Ding, Jie Feng, and Yong Li. UniMove: A unified model for multi-city human mobility prediction. *arXiv preprint arXiv:2508.06986*, 2025b. URL https://arxiv.org/abs/2508.06986.

Xiaocheng Huang, Yifang Yin, Simon Lim, Guanfeng Wang, Bo Hu, Jagannadan Varadarajan, Shaolin Zheng, Ajay Bulusu, and Roger Zimmermann. Grab-Posisi: An extensive real-life GPS trajectory dataset in southeast asia. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Prediction of Human Mobility*, pp. 1–10, 2019. doi: 10.1145/3356995.3364536.

Peiran Li, Jiawei Wang, Haoran Zhang, Xiaodan Shi, Noboru Koshizuka, Chihiro Shimizu, and Renhe Jiang. TrajFlow: Nation-wide pseudo GPS trajectory generation with flow matching models. *arXiv preprint arXiv:2603.15009*, 2026. URL https://arxiv.org/abs/2603.15009.

Yuxuan Liang, Kun Ouyang, Yiwei Wang, Xu Liu, Hongyang Chen, Junbo Zhang, Yu Zheng, and Roger Zimmermann. TrajFormer: Efficient trajectory classification with transformers. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management*, pp. 1229–1237, 2022. doi: 10.1145/3511808.3557481.

Yan Lin, Tonglong Wei, Zeyu Zhou, Haomin Wen, Jilin Hu, Shengnan Guo, Youfang Lin, and Huaiyu Wan. TrajFM: A vehicle trajectory foundation model for region and task transferability. *arXiv preprint arXiv:2408.15251*, 2024. URL https://arxiv.org/abs/2408.15251.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.03003.

Yanghong Liu, Xingping Dong, Ming Li, Weixing Zhang, and Yidong Lou. PatchTraj: Unified time-frequency representation learning via dynamic patches for trajectory prediction. *arXiv preprint arXiv:2507.19119*, 2025. URL https://arxiv.org/abs/2507.19119.

Qingyue Long, Can Rong, Huandong Wang, and Yong Li. One fits all: General mobility trajectory modeling via masked conditional diffusion. *arXiv preprint arXiv:2501.13347*, 2025. URL https://arxiv.org/abs/2501.13347.

Zhenyu Mao, Ziyue Li, Dedong Li, Lei Bai, and Rui Zhao. Jointly contrastive representation learning on road network and trajectory. *arXiv preprint arXiv:2209.06389*, 2022. URL https://arxiv.org/abs/2209.06389.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2211.14730.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Urs Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing*, 1(3):244–256, 1972.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1701.06538.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-MoE: Billion-scale time series foundation models with mixture of experts. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2409.16040.

Junjun Si, Jin Yang, Yang Xiang, Hanqiu Wang, Li Li, Rongqing Zhang, Bo Tu, and Xiangqun Chen. TrajBERT: BERT-based trajectory recovery with spatial-temporal refinement for implicit sparse trajectories. *IEEE Transactions on Mobile Computing*, 23(5):4849–4860, 2024. doi: 10.1109/TMC.2023.3297115.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional score-based diffusion models for probabilistic time series imputation. In *Advances in Neural Information Processing Systems*, volume 34, pp. 24804–24816, 2021.

UCI Machine Learning Repository. Taxi service trajectory—prediction challenge, ECML PKDD 2015. Dataset, 2015. URL https://archive.ics.uci.edu/dataset/339.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Jingyuan Wang, Ning Wu, Xinxi Lu, Wayne Xin Zhao, and Kai Feng. Deep trajectory recovery with fine-grained calibration using kalman filter. *IEEE Transactions on Knowledge and Data Engineering*, 33(3):921–934, 2021. doi: 10.1109/TKDE.2019.2940950.

Pengpeng Zhao, Haifeng Zhu, Yanchi Liu, Jiajie Xu, Zhixu Li, Fuzhen Zhuang, Victor S. Sheng, and Xiaofang Zhou. Where to go next: A spatio-temporal gated network for next POI recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019. URL https://ojs.aaai.org/index.php/AAAI/article/view/4537.

Yu Zheng. Trajectory data mining: An overview. *ACM Transactions on Intelligent Systems and Technology*, 6(3):29:1–29:41, 2015.

Yu Zheng, Yukun Chen, Quannan Li, Xing Xie, and Wei-Ying Ma. Understanding transportation modes based on GPS data for web applications. *ACM Transactions on the Web*, 4(1):1:1–1:36, 2010a. doi: 10.1145/1658373.1658374.

Yu Zheng, Xing Xie, and Wei-Ying Ma. GeoLife: A collaborative social networking service among user, location and trajectory. *IEEE Data Engineering Bulletin*, 33(2):32–39, 2010b.

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*, pp. 4676–4687, 2024. doi: 10.1145/3637528.3671866.

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. In *Advances in Neural Information Processing Systems*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/3c215225324f9988858602dc92219615-Abstract-Conference.html.

A ADAPTIVE-PATCH IMPLEMENTATION

Features and coordinate convention. All patch features use the zero-filled, normalized observation condition C. For valid adjacent points, define

$$d_i = C_i - C_{i-1}, \qquad v_i = d_i / \max(\Delta_i, 10^{-3}), \tag{11}$$

$$a_i^{\text{motion}} = (v_i - v_{i-1}) / \max(\Delta_i, 10^{-3}), \qquad z_i = \|v_i\|_2. \tag{12}$$

Differences at the first position and across invalid padding transitions are zero. Heading is computed with $\mathrm{atan2}$ from the two velocity components; its change is wrapped to $[-\pi, \pi]$ and taken in absolute value. The speed-change cue is $|z_i - z_{i-1}|$, and the observation-transition cue is $|m_i - m_{i-1}|$. These descriptors measure changes in normalized input space, including transitions to zero-filled coordinates.

Robust normalization and smoothing. For a nonnegative feature f, we normalize by its valid-position 95th percentile, falling back to the maximum for a degenerate scale and applying a 10^{-6} lower bound. Let $\mathcal{N}(f)$ denote this normalization with padding masked out. The rule score is

$$r^{\text{raw}} = 0.35\mathcal{N}(|\Delta\text{heading}|) + 0.30\mathcal{N}(\|a^{\text{motion}}\|_2) + 0.25\mathcal{N}(|\Delta z|) + 0.10|\Delta m|, \tag{13}$$

$$r = \text{clip}\big(\mathcal{N}(\text{AvgPool}_5(r^{\text{raw}})), 0, 1\big). \tag{14}$$

Width-five pooling has stride one and zero padding of two positions. The learned scorer takes ten features per position: two coordinates, two velocity components, two acceleration components, speed, heading change, interval, and observation flag. Two width-five Conv1D layers with 64 hidden channels and GELU activations precede a width-one scalar output and sigmoid. Scores are validity-masked, smoothed with width-five average pooling, and combined with the centered rule score as in Eq. 5.

Boundary procedure. We set $\lambda_r = 0.1$, $\theta = 0.65$, $\delta = 0.10$, $L_{\min} = 4$, and $L_{\max} = 24$ in Eqs. 5–6. Algorithm 1 gives the complete scan. Patches are nonoverlapping and constructed by the sequential scan in Algorithm 1. A terminal segment shorter than four points is merged with its predecessor, allowing the last patch to contain up to 27 points; shorter trajectories form one patch.

Algorithm 1 Observation-conditioned adaptive patching

Require: Condition C, intervals Δ, masks m, a; valid length n
1: Construct motion and mask-transition features from C, Δ, m, a
2: Robust-normalize motion cues and compute weighted rule score
3: Smooth and normalize rule score to obtain r; smooth learned score to obtain ℓ
4: Fuse r and ℓ into s using Eq. 5
5: $j \leftarrow 1$; $\mathcal{P} \leftarrow [\,]$
6: **while** $j \le n$ **do**
7: $\quad i \leftarrow \min(j + 23, n)$
8: $\quad$ Scan $k = j + 3, \ldots, \min(j + 22, n)$; if Eq. 6 holds, set $i \leftarrow k$ and stop the scan
9: $\quad$ Append $[j, i]$ to $\mathcal{P}$; $j \leftarrow i + 1$
10: **end while**
11: Merge a final patch shorter than four points into its predecessor, if present
12: **return** $\mathcal{P}$ and s

Patch encoding and batching. The patch point MLP maps four inputs (coordinates, interval, observation flag) through width 256, GELU, and a second width-256 layer. Eq. 7 initializes one token per patch with a learned length embedding. Each of the two refinement blocks uses four-head cross-attention from this token to the points within the patch, followed by a residual feed-forward block of expansion factor four; dropout is 0.1. Sequences of patches are padded to the batch maximum and masked in attention. Refined patch tokens enter the main Transformer by concatenation with point tokens. At inference, condition embeddings and patch features are cached across the three field evaluations.

B VELOCITY OBJECTIVE

TrajCFM uses a linear conditional flow path and target-position L1 velocity regression for coordinate reconstruction. L1 gave the best reconstruction accuracy among the objectives evaluated in preliminary experiments. At a fixed state and condition, the population optimum of coordinatewise L1 regression is a conditional median of path velocity; squared-error FM regression instead estimates its conditional mean. The objectives therefore define different velocity estimators, and the standard squared-error FM distribution-matching result does not directly apply. Eq. 9 combines the L1 term with patch-score variance regularization. Scorer supervision and routing auxiliary losses have zero weight.

C RECONSTRUCTION PROTOCOL

Data processing. The loader represents coordinates internally as longitude and latitude, applies the recorded resampling pipeline, and retains at most 200 points. Each trajectory is centered at its first observed point and transformed with the checkpoint's fixed affine normalization. Intervals are in seconds; the first is zero. The normalization constants, in longitude/latitude order, are

$$\mu = (3.0675081728314398 \times 10^{-6}, -5.4481778948893874 \times 10^{-5}), \tag{15}$$

$$\sigma = (0.03790469261263497, 0.030818511463853598). \tag{16}$$

The same transform and inverse are used across the reported WorldTrace controls. The coordinate origin is selected from the observed points without accessing masked coordinates; it is also used for inverse normalization during scoring. Input intervals and sequence length remain available at target positions.

Training masks. The model samples completion with probability 0.7 and five-point suffix prediction with probability 0.3. Within completion, random masking selects half the valid positions; the shape-aware branch uses RDP-derived candidates and adjusts their count to the target mask size. The block branch selects a short contiguous region and supplements it with random positions to reach the mask budget. RDP-based selection uses the full training trajectory to construct a self-supervised mask; hidden coordinates are then removed from the model condition. Formal completion evaluation uses random masks only.

Architecture and optimization. Table 6 summarizes the shared model and WorldTrace training configuration. WorldTrace contains 1,985,551 training and 220,617 validation trajectories after the deterministic split, with 245,129 trajectories in the original test set. We use AdamW and select the checkpoint with the lowest validation loss. A validation-loss plateau triggers learning-rate decay and early stopping.

Table 6: Shared architecture and WorldTrace reconstruction configuration. Adaptive-boundary parameters are specified in Appendix A.

Setting	Value
Backbone width / depth / attention heads	256 / 6 / 4
Feed-forward expansion	4
Experts per block / active experts per token	4 / 2
Patch refinement layers / heads / dropout	2 / 4 / 0.1
Maximum trajectory length	200 points
Completion / prediction sampling probability	0.7 / 0.3
Completion masking / prediction horizon	50% / final 5 points
WorldTrace optimizer / initial learning rate	AdamW / 10^{-4}
Weight decay / gradient clipping	10^{-4} / 1.0
WorldTrace global batch	384
WorldTrace training seeds	0, 1, 2, 42
Reconstruction sampler	Euler, 3 evaluations

Cross-dataset adaptation. Cross-dataset experiments fine-tune the WorldTrace-pretrained model on each target dataset, retaining the shared normalization, mask mixture, architecture, and three-step evaluation. Chengdu and Xi'an use the processed JCLRNT full splits (Mao et al., 2022); GeoLife and Porto use their respective trajectory splits. All four datasets use the WorldTrace optimization settings in Table 6: AdamW with an initial learning rate of 10^{-4}, weight decay 10^{-4}, gradient clipping at 1.0, and a global batch size of 384. Training uses FP16 AMP, with learning-rate decay and early stopping based on validation loss; the lowest-validation-loss checkpoint is retained.

For these four reconstruction datasets, eligible original trips are sorted chronologically and divided into training, validation, and test sets at a 70:10:20 ratio before windowing. All windows from a trip inherit its split. Windows are nonoverlapping, contain 36–200 points, and discard a final remainder shorter than 36 points. Table 7 reports the resulting window counts; the proportions apply to trips, not windows.

Chengdu and Xi'an coordinates are reconstructed along the released map-matched road polylines using segment passage times, then sampled at five-second intervals. GeoLife is resampled at five-second intervals after splitting at gaps longer than 60 seconds and non-increasing timestamps; interpolation does not cross these breaks. Its reconstruction split is trip-disjoint, unlike the user-disjoint classification split. Porto retains its native 15-second samples without resampling and excludes incomplete or invalid trips.

Table 7: Processed reconstruction windows for cross-dataset evaluation. Original trips are split chronologically before forming nonoverlapping windows.

Dataset	Interval (s)	Train	Validation	Test
Chengdu	5	50,808	7,218	14,444
Xi'an	5	151,335	20,738	43,312
GeoLife	5	73,482	12,513	14,810
Porto	15	738,316	105,448	210,912

Metric reduction. Let d_{bi} be the geodesic distance in meters between predicted and true coordinates at target position i of trajectory b, after inverse normalization. With n_b target points, define

$$\mathrm{MAE}_b = \frac{1}{n_b}\sum_i d_{bi}, \qquad \mathrm{RMSE}_b = \sqrt{\frac{1}{n_b}\sum_i d_{bi}^2}. \tag{17}$$

Dataset MAE and RMSE are the arithmetic means of their respective per-trajectory values, giving each trajectory equal evaluation weight. The main reconstruction comparisons in Tables 1 and 2 report means and sample standard deviations over four independently trained seeds on each dataset; architectural comparisons use seed 42. Following the WorldTrace protocol, evaluation uses one Gaussian noise draw per trajectory and task, scoring the resulting reconstruction directly.

Table 8: WorldTrace completion and prediction errors for four independently trained seeds, evaluated on 245,129 test trajectories with three Euler evaluations. MAE and RMSE are in meters; lower is better.

	Completion		Prediction	
Seed	MAE	RMSE	MAE	RMSE
42	5.71	8.64	31.81	37.56
0	5.71	8.64	31.56	37.27
1	5.81	8.78	32.08	37.87
2	5.77	8.71	31.56	37.27

D Architectural Controls

Factorial variants. Plain DiT removes both the patch branch and sparse expert layers. AP-only retains the patch branch and uses dense feed-forward layers. MoE-only removes the entire patch

branch and retains sparse expert layers. Full includes both components. AP appends patch tokens to the retained point sequence, while MoE replaces the dense feed-forward transformation with sparse expert computation.

The architectural comparisons use seed 42; the WorldTrace main-model summary averages four training seeds.

Targeted controls. The total-parameter-matched dense model retains adaptive patching and replaces the sparse experts with width-4096 feed-forward layers. Full has 18,705,243 parameters; Dense has 18,694,467, a difference of 10,776 (0.06%). Full uses four width-1024 experts per block, activating two per token. Fixed16+MoE retains the scorer, weighted pooling, patch refinement, and sparse experts, but partitions the sequence into fixed-length patches of 16 points. Full gives lower errors than both controls on all four metrics (Table 9). The dense match concerns total parameters; activated computation can differ. On 2,000 completion validation records, AP has 12.66% higher within-patch speed variance and 16.40% higher heading dispersion than equal-width partitions with the same patch count; paired 95% intervals exclude zero. Its reconstruction benefit therefore does not imply more homogeneous physical motion within patches.

Table 9: Targeted architectural controls on the full WorldTrace test set, using seed 42 and three Euler evaluations. Errors are in meters; bold marks the lowest errors.

	Completion		Prediction	
Model	MAE	RMSE	MAE	RMSE
Parameter-matched Dense	5.94	8.91	32.34	38.14
Fixed16+MoE	6.10	9.12	32.64	38.47
Full	**5.71**	**8.64**	**31.81**	**37.56**

Direct coordinate regression. We compare TrajCFM with a direct coordinate-regression variant retaining the DiT backbone, observation conditioning, adaptive patches, and sparse experts. Under the same data split and evaluation protocol, the regression variant uses an L1 coordinate loss and yields higher errors on both tasks (Table 10).

Table 10: Direct coordinate regression versus TrajCFM on WorldTrace under the same data split and evaluation protocol. Both retain the DiT backbone, observation conditioning, AP, and MoE. The regression variant uses an L1 coordinate loss; TrajCFM uses L1 velocity regression and three Euler evaluations. Errors are in meters; bold marks the lowest errors.

	Completion		Prediction	
Model	MAE	RMSE	MAE	RMSE
Direct coordinate regression	9.34	14.01	46.74	53.58
TrajCFM	**5.71**	**8.64**	**31.81**	**37.56**

E Euler-Step Selection

One seed-42 checkpoint is evaluated on validation data with $S \in \{1, 2, 3, 4, 8, 16\}$. For each of four error metrics, divide each candidate's error by the minimum across this sweep. Select the step count with the lowest mean of these four normalized errors; ties favor fewer steps. The joint criterion selects three steps, which are then fixed for the formal tests. This reduces velocity-network evaluations by 81.25% relative to sixteen steps.

Table 11: Euler-step sweep on WorldTrace validation data with a fixed checkpoint. Errors are in meters. Three steps minimize the mean of the four errors normalized by their respective sweep minima.

	Completion		Prediction	
Steps	MAE	RMSE	MAE	RMSE
1	7.17	9.97	33.28	38.90
2	5.82	8.77	31.81	37.51
3 (selected)	5.75	8.71	32.09	37.86
4	5.76	8.73	32.64	38.51
8	5.77	8.82	34.57	40.82
16	5.88	9.07	36.62	43.28

F Motion Groups and Routing

The performance diagnostic evaluates the frozen seed-42 Full and validation-selected AP-only models on 5,000 validation trajectories. Both use paired conditions/noise, FP32 inference, batch 32, and three Euler evaluations. Table 13 reports the per-group errors for this checkpoint pair. Motion descriptors are computed from ground truth for post-hoc analysis. Local tangent-plane displacement gives speed; the angle between consecutive displacements gives turn angle; the absolute speed difference divided by the mean of the two intervals gives the speed-change descriptor. Turn angles are eligible only when both displacements exceed 3 m and both speeds exceed 0.5 m/s. Reconstruction errors use geodesic distance.

Thresholds are fitted on 2,048 training trajectories. Low speed is at or below the training speed Q_{25} (approximately 5.696 m/s). Steady straight motion has turn angle at or below Q_{25} and speed change at or below Q_{50}; the performance analysis additionally excludes low-speed points from this group. Larger local turns have angle at or above Q_{75} (approximately 3.832 degrees), and speed changes exceed their descriptor's Q_{75} (approximately 0.855 m/s^2).

For each trajectory and group, MAE and RMSE use only target points in that group; the table averages over trajectories with at least one such point. The same trajectory can contribute to several groups. Table 12 gives these trajectory counts.

Table 12: Eligible trajectory counts for the paired motion-group analysis on 5,000 validation trajectories. A trajectory contributes when it contains target points in the group; groups may overlap.

Group	Completion	Prediction
Low speed	4191	2829
Steady straight	4122	1156
Larger local turns	4813	1870
Speed changes	4580	2140

The routing visualization uses a separate 2,000-trajectory validation analysis with the same thresholds and low-speed exclusion for the steady-straight group. Normalized selected-expert weights are averaged over eligible unobserved point tokens, assigning zero to unselected experts. We display the third Euler evaluation, selected after inspecting the three evaluations. Each layer has its own expert networks, so expert indices are interpreted within a layer.

Routing intervention. A complementary probe exchanges point-token routing weights within and across low-speed and steady-straight groups at the second layer and second Euler evaluation during prediction. The paired exchanges use the same tokens and match changes in the top-2 expert set and routing-weight magnitude. The completed probe found no consistent error advantage for within-group over cross-group exchange. The routing heatmaps therefore capture associations between motion and expert use, while the intervention leaves their causal role unresolved.

Table 13: Motion-group MAE on 5,000 paired validation trajectories for the diagnostic checkpoint pair in Appendix F. Reductions are computed from unrounded errors, in meters. Motion groups may overlap.

	Completion			Prediction		
Group	AP-only	Full	Reduction	AP-only	Full	Reduction
Low speed	5.97	5.68	0.29	22.07	20.80	1.26
Steady straight	4.90	4.49	0.41	44.92	42.51	2.41
Larger local turns	8.01	7.61	0.39	70.79	69.77	1.02
Speed changes	5.63	5.27	0.35	28.95	27.60	1.35

G Downstream Generation Details

Adaptation and reference. Each conditioning road route is resampled by arc length to 200 points. The TrajCFM encoder is frozen and evaluated once with all route points observed, unit intervals, flow state equal to the normalized route, and flow time $\tau = 0.5$. We extract the last main block's point tokens and average consecutive groups of five, obtaining 40 condition vectors of dimension 256.

The adapter standardizes features using training-split statistics and projects $256 \rightarrow 128$. The generator and adapter are trained with denoising MSE and auxiliary geometry regression, weighted by approximately 9.78, to the mean relative coordinates of each five-point window; TrajCFM remains frozen. The downstream split contains 16,384 training routes and 128 validation routes; batch size is 32. The learning rate decreases from 2×10^{-4} to 5×10^{-5} and then 10^{-5}.

The reference trains ControlTraj with frozen RoadMAE conditions and denoising MSE. We compare the complete adaptation systems with shared evaluation routes, noise draws, sampling schedule, output length, and metrics; adaptation objectives and training budgets differ.

Generator inputs and sampling. Both arms also receive the same six-field ControlTraj head input: departure in five-minute bins, road-route distance divided by 10,000 m, duration divided by 3,600 s, distance per road-record transition divided by 100 m, average speed divided by 20 m/s, and road-record count divided by 200. These fields are constructed from the conditioning session, separately from the encoder's unit intervals. This is our compatible input construction rather than the original unpublished continuous-head preprocessing. Sampling uses the same deterministic diffusion update in both arms, with a 500-position linear noise schedule (β from 10^{-4} to 0.05) and stride five, giving 100 generator evaluations. The three-step flow sampler applies only to completion and prediction.

Independent evaluation. The 512 Chengdu sessions are disjoint from training and development by session key and source row. Ten noise draws are paired by route. Reported values average per-draw metrics.

Metric definitions. For occupancy histograms P and Q, define $M = (P + Q)/2$ and

$$\mathrm{JSD}(P, Q) = \tfrac{1}{2}\,\mathrm{KL}_2(P\|M) + \tfrac{1}{2}\,\mathrm{KL}_2(Q\|M), \tag{18}$$

where logarithms have base two and zero-probability contributions are zero. Density uses a frozen reference-anchored 16×16 spatial grid and an additional out-of-grid bin, shared across arms. Length JSD compares total path lengths of sampled windows; step JSD compares adjacent-point distances. Road P50/P95 are distances to the conditioning polyline; off-road is the fraction of sampled points farther than 100 m from it.

For length and step-distance histograms, distances d in meters are transformed as $\log(1 + d)$. Each uses 32 equal-width intervals between the corresponding training-set minimum and maximum in this transformed space, plus one upper-overflow bin. Interior intervals are left-closed and right-open; values at or above the upper boundary enter the overflow bin, and values below the lower boundary enter the first bin. The edges are fixed across methods and noise draws, and counts are normalized before computing JSD. For density, points outside the spatial grid, including its upper boundary, enter the shared out-of-grid bin rather than being clipped to edge cells.

H CLASSIFICATION DETAILS

GeoLife. GeoLife labels are grouped into walk, bus, bike, and car, with taxi included in car. Five-second sampling produces windows of 20–200 points. The user-disjoint split and training configuration are listed in Table 14. TrajCFM and UniTraj use full-model fine-tuning; the remaining baselines are trained from scratch. TrajCFM features are extracted before the coordinate head with fully observed inputs and $\tau = 1$, then mean-pooled over valid points. The classifier and optimizer settings are shared across both classification datasets; the output dimension matches each dataset's label set.

Grab-Posisi. Grab-Posisi provides car/motorcycle labels for Jakarta trajectories, used for binary transport-mode classification in prior work (Huang et al., 2019; Liang et al., 2022). The processed classification set contains 319,587 trajectory windows. We split Grab-Posisi into training, validation, and test sets at an 8:1:1 ratio by original trajectory, stratified by transport mode. All windows from the same trajectory remain in the same split. Both TrajCFM and UniTraj are fully fine-tuned using the shared classification settings, with dataset-specific output layers. Accuracy and macro-F1 are reported in Table 5.

Table 14: Classification configuration. Split counts and the four-class output refer to GeoLife; Grab-Posisi shares the training settings with a dataset-specific output layer.

Setting	Value
Labels	Walk, bus, bike, car (including taxi)
Split unit / proportions	User / 70%–10%–20%
Sampling interval / maximum length	5 s / 200 points
Training seed / selection criterion	42 / validation macro-F1
TrajCFM / UniTraj adaptation	Full-model fine-tuning
Other baselines	Training from scratch
Train / validation / test users	42 / 6 / 12
Train / validation / test windows	9,144 / 1,988 / 2,313
TrajCFM pooling	Mean of valid point tokens
Classifier	LayerNorm, Linear(256,256), GELU, Dropout(0.1), Linear(256,4)
Optimizer / weight decay	AdamW / 10^{-4}
Batch size / precision	64 / FP32
Loss	Training-class-weighted cross-entropy
Fine-tuning LR: encoder / classifier	2×10^{-5} / 10^{-3}
Scratch-trained baseline LR	10^{-3}

I PACKED-ROUTING EFFICIENCY

Table 15 compares the same Full model with standard and packed token routing. Packing reduces padding computation without changing the architecture.

Table 15: Training cost of Full with standard and packed routing on two RTX 4090 GPUs, global batch 384. Median update time is measured after warm-up; memory is peak allocated GPU memory. The paired runs use the same data and model state.

Implementation	Time (ms/update)	Memory (GB)
Full, standard routing	298.39	10.21
Full, packed routing	283.15	8.59
Reduction	5.11%	15.87%