2024 IEEE International Conference on Data Mining (ICDM) | 979-8-3315-0668-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICDM59182.2024.00091

Controllable Visit Trajectory Generation with Spatiotemporal Constraints

Haowen Lin[1], John Krumm[1], Cyrus Shahabi[1], Li Xiong[2]
[1]*University of Southern California, USA* [2]*Emory University, USA*
{haowenli, jkrumm, shahahbi}@usc.edu, lxiong@emory.edu

***Abstract*—Human mobility data, represented as sequences of visits, are crucial for various application domains, including transportation, urban planning, and public health. However, large-scale human mobility data is typically inaccessible to researchers due to the high cost of data collection and privacy concerns. This limitation has led to several studies proposing learned models to generate synthetic visit sequences. Despite this progress, existing approaches lack mechanisms to control the generation process, which prevents the incorporation of prior knowledge and the spatiotemporal specification of certain visits. To address these limitations, we formally define the Constraint Trajectory Generation problem and introduce Geo-CETRA (Constraint Enforced Trajectory Generation), a novel framework that operates within the continuous spatiotemporal space, enabling direct generation of geographical coordinates and durations of each visit in a trajectory. Geo-CETRA reparameterizes the sampling space for effective enforcement of various spatiotemporal constraints. Furthermore, incorporating a constraint factorization approach along with an innovative beam decoding module, Geo-CETRA facilitates the production of high-quality synthetic trajectories that realistically emulate human movement while satisfying predefined spatiotemporal constraints. Experiments on real and synthetic datasets demonstrate Geo-CETRA's superior precision and contextual accuracy compared to existing approaches.**

***Index Terms*—Spatial-temporal systems, Controlled generation**

I. Introduction

Human mobility data, conceptualized as trajectories (or sequences of visits), are critical for various studies, including transportation, urban planning, social dynamics, and epidemiology [1]. However, researchers in these fields often lack access to large-scale trajectory data due to the high cost of collection and the privacy sensitivity of location data [2]. Recent advances in generative AI have facilitated the development of data-driven models that can generate large-scale and diverse synthetic trajectories directly from real-world samples [3]–[5]. However, a significant limitation of these techniques is the lack of control over the generation process. Control is important, for example, to incorporate prior knowledge into the generation process. If we know that certain trajectories typically start from a residential area in the morning, proceed to a business district during working hours, and return to the residential area at night, then controlling the synthetic trajectories to adhere to these patterns results in more realistic simulations, reflecting feasible speeds and complying with specific personal preferences. Alternatively, for specific downstream tasks such as epidemiology, we may want to ensure that synthetic trajectories pass through specific hotspots, like a concert venue, at certain times of the day. This approach would aid policy makers in effectively targeting interventions in specific areas.

In this paper, we focus on visit-based trajectories that capture the sequence of locations and individual visits, such as home, the office, and the gym [6]. Each visit is specified by its geo-coordinates and the time range of the visit. A visit-based trajectory consists of a few visits per person per day, making it temporally coarse with irregular inter-arrival times, and hence, challenging to model [7], [8]. We employ spatiotemporal (ST) constraints as our control specification. Essentially, a spatiotemporal constraint can be thought of as a 3D cube that defines the valid ranges for a visit in terms of latitude, longitude, and time. The generation of visit-based synthetic trajectories should take these spatiotemporal constraints as input and produce synthetic trajectories that are learned from previously provided training data and adhere to these constraints.

Imposing spatiotemporal constraint into trajectory generation is challenging. A common approach for achieving constrained generation involves collecting task-specific examples and introducing additional input signals or conditioning information into the model during the training or fine-tuning stages. This method is prevalent in many natural language processing (NLP) applications, ensuring that generated text adheres to specific requirements for style and tone [9], [10]. However, this approach often fails to follow fine-grained constraints (e.g., enforcing a specific word in neural text generation or following spatiotemporal constraints in our task), even with large training datasets [11] since they do not guarantee the enforcement of the desired constraints. Another potential approach to circumvent this problem is to optimize the generator in an unconstrained manner and then post-process the generated trajectories by filtering out trajectories that are not valid or deliberately change the results to satisfy constaints [12]. However, this method can be limited because trajectories inherently exhibit complex and irregular characteristics of moving patterns, with each data point existing in continuous time and space. This makes it highly unlikely for a trajectory in the unconstrained space to satisfy different spatiotemporal control, rendering post-processing methods both inefficient and ineffective.

To address these challenges, we propose Geo-CETRA (Constraint Enforced Trajectory Generation), a general framework that can generate high-quality synthetic trajectories in continuous space-time while effectively enforcing given spatiotemporal constraints by reparameterizing the sampling space itself and controlling the decoding of the generation

2374-8486/24/$31.00 ©2024 IEEE
DOI 10.1109/ICDM59182.2024.00091

process. Specifically, we first show how different constraints can be factorized and interpreted for the model design. Secondly, we propose a novel constrained spatiotemporal generation model that extracts knowledge from continuous space and time. The model reflects real-world moving patterns more accurately with a temporal and spatial encoder using random Fourier features, and it explicitly enforces constraints using reparameterization. Finally, we design a new decoding module to control the generation process and further improve the final performance. The major contributions of this work can be summarized as:

- To the best of our knowledge, we are the first to formally formulate and define the problem of constrained trajectory generation with continuous spatiotemporal constraints and generate trajectories in a continuous spatiotemporal space.
- We present our Geo-CETRA model, which learns from the inherent complex and continuous spatial and temporal patterns and produces high-dimensional spatiotemporal representations to support generating realistic trajectories with constraints.
- We evaluate our Geo-CETRA on two datasets. The experimental results demonstrate that Geo-CETRA outperforms other state-of-the-art methods.

II. Problem Definition

Definition 1. ***Visit-based trajectory*** A visit-based trajectory is defined as $\tau = \{(t_i, \boldsymbol{l}_i)\}_{i=1}^{L}$, where $t_i \in \mathbb{R}^+$ is the timestamp, $\boldsymbol{l}_i \in \mathbb{R}^d$ denotes the spatial location associated with each timestamp, and L is the total number of spatiotemporal points. Each spatial location corresponds to a two-dimensional latitude-longitude coordinate such that $\boldsymbol{l}_i = (x_i, y_i)$.

Problem 1 (Constrained trajectory generation)**.** *Given a real-world trajectory dataset where each trajectory is subject to a distinct spatiotemporal constraint $C(\tau)$, the objective is to develop a model that captures the spatiotemporal patterns $p(\tau)$ and generate new trajectories that can retain the key characteristics of the original dataset, which can be formulated as:*

$$\begin{aligned} &\textit{maximize} && \log p(\tau) \\ &\textit{subject to} && C(\tau) \end{aligned} \tag{1}$$

Trajectory models rely on an auto-regressive factorization to perform likelihood estimation and generation of trajectories, which is defined as follows:

$$p_\theta(\tau) = \prod_{i=1}^{L} p_\theta((t_i, \boldsymbol{l}_i)|(t_{j<i}, \boldsymbol{l}_{j<i})) \tag{2}$$

Definition 2. ***Spatiotemporal constraint*** In this study, we focus on spatiotemporal constraints such that the trajectory τ must go through location $\boldsymbol{l}_C$ within a confined area $A \subset \mathbb{R}^2$ during a certain time frame $[t_s, t_e]$, i.e. the traveler is known to arrive within A at time $t_C \in [t_s, t_e]$ which denote the starting time and the ending time of the constrained time frame. We assume A defines a rectangular region $\{\boldsymbol{l} = (x, y)|x_s \le x \le x_e, y_s \le y \le y_e\}$. We therefore reformulate the original optimization problem under the constraint to be:

$$\begin{aligned} &\text{maximize} && \log p(\tau) \\ &\text{subject to} && (t_C, \boldsymbol{l}_C) \in \tau, \\ & && t_C \in [t_s, t_e], \\ & && \boldsymbol{l}_C \in \{\boldsymbol{l} = (x, y)|x_s \le x \le x_e, y_s \le y \le y_e\} \end{aligned} \tag{3}$$

III. Methods

In this section, we detail our Geo-CETRA for constrainted trajectory generation. The architecture is shown in Figure 1.

A. Constraint Factorization

Since the auto-regressive trajectory generation model needs to iteratively sample the next point based on the distribution conditioned on the past trajectory, it is necessary to decouple the spatiotemporal (ST) constraint for each step when evaluating if a generation hypothesis satisfies the constraint $C(\tau)$ during generation. For a given ST constraint for each trajectory τ, the result after constraint factorization is a set of valid ST constraints at each i-th step of trajectory (i.e., $C = \{C_1, C_2, ...\}$). The generated visit-level constraints will be incorporated as input in Section III-C to define the constrained sampling space. At the generation stage, we first sample i_C from a prior index distribution p_C where $p_C(i)$ determines how likely the i-th visit point falls into the constrained time frame defined in $C(\tau)$, i.e. i is the constrained index ($i = i_C$). With no observed trajectory dataset, the index distribution would be a uniform distribution, i.e., $P(i_C|C) = 1/L$ over $i_C \in \{1, ..., L\}$. With the observed trajectory dataset $\mathcal{T}$, the index distribution p_C is calculated as below:

$$P(i_C|C, \mathcal{T}) = \frac{\sum\limits_{\tau \in \mathcal{T}} \sum\limits_{(t_j, \boldsymbol{l}_j) \in \tau} [\mathbb{1}_{\{j=i_C\}} \mathbb{1}_{\{t_j \in [t_s, t_e]\}}]}{\sum\limits_{\tau \in \mathcal{T}} \sum\limits_{(t_j, \boldsymbol{l}_j) \in \tau} \mathbb{1}_{\{t_j \in [t_s, t_e]\}}} \tag{4}$$

where time frame $[t_s, t_e]$ is used to define the temporal constraint within $C(\tau)$, and $\mathbb{1}_{\{\cdot\}}$ is the indicator function.

After identifying the index of the visits with specified location and time constraints in $C(\tau)$, we define the constrained time and space for other points in the generated trajectory in Algorithm 1. Specifically, we iterate over each step of the trajectory, ensuring that visits prior to the i_C-th arrive before t_s (line 2-4). For the visit at the i_C-th step, we establish a temporal boundary, requiring the i_C-th point to arrive between t_s and t_e, with its geo-coordinate within a constrained spatial range (line 5). For visits following the i_C-th, no further spatial or temporal constraints are set (line 6-8). Practically, for points without specific constraints, we apply a minimal and maximal time duration, ΔT_{min} and ΔT_{max}, and spatial boundaries, X_{min}, X_{max}, Y_{min} and Y_{max} (defined as the boundary of the trajectories in the study area), to prevent numerical errors and wildly unrealistic results.

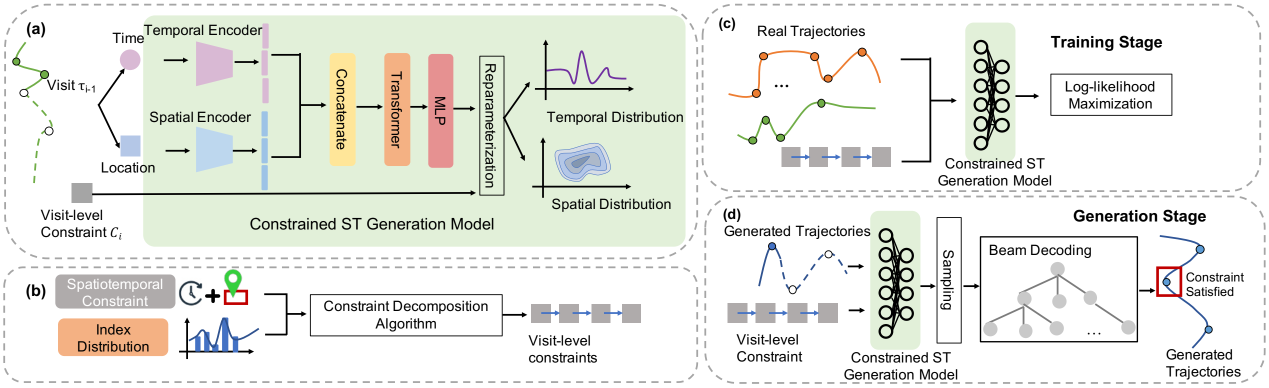

Fig. 1. The workflow of our proposed Geo-CETRA framework.

Algorithm 1: Constraint Factorization

Input: ST constraint $C = \{t_s, t_e, x_s, x_e, y_s, y_e\}$; Index distribution p_C; Minimal and maximal time duration $\Delta T_{\min}$, $\Delta T_{\max}$; Minimal and maximal spatial boundary $X_{\min}$, $X_{\max}$, $Y_{\min}$, $Y_{\max}$.

Output: Factorized constraint for each spatiotemporal point $C_i = \{\Delta t_{is}, \Delta t_{ie}, x_{is}, x_{ie}, y_{is}, y_{ie}\}$, $i = 1, ..., L$

Sample the constrained index $i_C \sim p_C$
for $i \longleftarrow 1, \ldots, i_C - 1$ **do**
  $C_i = \{\Delta T_{\min},\ t_s - \sum_{j=1}^{i-1} \Delta t_j,\ X_{\min},\ X_{\max},\ Y_{\min},\ Y_{\max}\}$
end
$C_{i_C} = \{t_s - \sum_{j=1}^{i_C - 1} \Delta t_j,\ t_e - \sum_{j=1}^{i_C - 1} \Delta t_j,\ x_s,\ x_e,\ y_s,\ y_e\}$
for $i \longleftarrow i_C + 1, \ldots$ **do**
  $C_i = \{\Delta T_{\min},\ \Delta T_{\max},\ X_{\min},\ X_{\max},\ Y_{\min},\ Y_{\max}\}$
end

B. Spatiotemporal Encoder

In this section, we introduce the details of our constrained ST generation model design which first employs a temporal encoder, a Fourier-based location encoder, and a multi-layer transformer.

Temporal Encoder. Extracting temporal features from visit-based trajectories is challenging due to their inherently irregular and asynchronous visiting patterns. To preserve the uneven temporal information of inputs (i.e., timestamps) [13], we utilize a deterministic trigonometric temporal encoding function which is analogous to the positional encoding method proposed in [14] and is defined as following:

$$[\boldsymbol{h}_t(t_i)]_j = \begin{cases} \cos\left(\frac{t_i}{\lambda^{\frac{j-1}{H_t}}}\right) & \text{if } j \text{ is odd,} \\ \sin\left(\frac{t_i}{\lambda^{\frac{j}{H_t}}}\right) & \text{if } j \text{ is even.} \end{cases} \tag{5}$$

where $[\boldsymbol{h}_t(t_i)]_j$ is the j-th element of the i-th row of the temporal embedding matrix $\boldsymbol{h}_t \in \mathbb{R}^{L \times H_t}$, $j \in \{1, ..., H_t\}$, and λ is a hyperparameter that controls the scaling.

Spatial Encoder. Similar to temporal embedding, encoding geo-coordinates into a high-dimensional representation while preserving useful spatial information is a challenging task. Instead of employing a linear projection, which can result in spectral bias as the linear layer only approximates low-frequency functions and fails to capture high-frequency details from low-dimensional inputs [15], we apply a Fourier-based location encoder to $\boldsymbol{l}_{1:L}$ to capture rich high-frequency details from low-dimensional GPS inputs [16]. Specifically, we leverage random Fourier features (RFFs) for this positional encoding. We construct an encoding layer that determines the range of frequencies to be encoded and set the frequencies using a fixed matrix R, whose entries are sampled from a normal distribution with standard deviation σ. The matrix R is determined at the beginning of training and remains unchanged throughout the training and generation process. The RFF operation $\gamma(\cdot)$ encodes GPS coordinate $\boldsymbol{l}_{1:L}$ as

$$\boldsymbol{h_l}(\boldsymbol{l}_i) = \gamma(\boldsymbol{l}_i) \equiv \text{concat}(\cos(2\pi R \boldsymbol{l}_i), \sin(2\pi R \boldsymbol{l}_i)) \tag{6}$$

where $\boldsymbol{h_l}(\boldsymbol{l}_i)$ is the i-th row of the spatial embedding matrix $\boldsymbol{h_l} \in \mathbb{R}^{L \times H_l}$, and the entries of the m-th row and n-th column of matrix R are sampled from the normal distribution: $R_{m,n} \sim \mathcal{N}(0, \sigma)$.

Transformer Encoder. The transformer encoder [14] has been widely employed to learn spatiotemporal patterns from trajectories [17], [18]. To capture sequential visits with complex spatiotemporal contexts, we first derive dense spatiotemporal representation $Z \in \mathbb{R}^{L \times (H_t + H_l)}$ via concatenating the temporal and spatial embedding matrices:

$$Z = \text{concat}(\boldsymbol{h}_t, \boldsymbol{h_l}) \tag{7}$$

After the initial encoding and embedding layers, we pass Z through a multi-head self-attention (MSA) module. Specifically, we compute the self-attention head output $\boldsymbol{h}_A$ by:

$$\begin{aligned} Q = ZW_Q, K &= ZW_K, V = ZW_V, \\ \boldsymbol{h}_A &= \text{softmax}\left(\frac{QK^T}{\sqrt{H_t + H_l}}\right) \times V, \end{aligned} \tag{8}$$

where Q, K, and V are the query, key, and value matrices obtained by different transformations of Z, and $W_Q \in \mathbb{R}^{(H_t+H_l)\times H_Q}$, $W_K \in \mathbb{R}^{(H_t+H_l)\times H_K}$, and $W_V \in \mathbb{R}^{(H_t+H_l)\times H_V}$ are learnable weight matrices for the linear transformations, respectively.

The MSA operation involves running multiple parallel self-attention heads to obtain a comprehensive representation of the past trajectory. The outputs of all attention heads are then concatenated and projected using a learnable weight matrix $W_H \in \mathbb{R}^{H\times H}$ to get the final encoding matrix $\boldsymbol{h} \in \mathbb{R}^{L\times H}$:

$$\boldsymbol{h} = \text{concat}(\boldsymbol{h}_A^{(1)}, \boldsymbol{h}_A^{(2)}, \ldots, \boldsymbol{h}_A^{(\frac{H}{H_V})}) \times W_H \tag{9}$$

C. Modeling $p(\tau)$ with Reparameterization

In this section, our objective is to model the trajectory distribution $p(\tau)$ using the embedding $\boldsymbol{h}$ derived from past trajectories and define the probability distribution by transforming it into a continuous constrained spatiotemporal space utilizing the visit-level constraint extracted from Section III-A.

Inter-visit Time Distribution Reparameterization. The distribution of one-dimensional inter-visit time is modeled using a mixture of log-normal distributions, which has proven to be effective for irregular temporal density estimation [19]. It is important to note that the encoding vector $\boldsymbol{h}_i \in \mathbb{R}^H$ serves as a summary of past history and acts as the condition for predicting the next visit. To generate the i-th visit, the $i-1$-th point of the encoding matrix $\boldsymbol{h}$ is processed through a Multilayer Perceptron (MLP) layer, which then outputs a set of means, standard deviations, and weights that define a Gaussian mixture model (GMM):

$$(\mu_{t_i,1:M}, \sigma_{t_i,1:M}, w_{t_i,1:M}) = \text{MLP}_t(\boldsymbol{h}_i) \tag{10}$$

where the Gaussian mixture model is defined as:

$$\mathcal{G}_t = \sum_{m=1}^{M} \frac{w_{t,m}}{\sqrt{2\pi}\sigma_{t,m}} \exp\left(-\frac{(z_t-\mu_{t,m})^2}{2\sigma_{t,m}^2}\right) \tag{11}$$

The unconstrained latent time variable z_{t_i} can be sampled from the Gaussian mixture model

$$z_{t_i} \sim \mathcal{G}_t(\mu_{t_i,1:M}, \sigma_{t_i,1:M}, w_{t_i,1:M}) \tag{12}$$

Next, the latent time variable is mapped to the desired constraint range through the following set of differentiable transformation functions.

$$u_{t_i} = \text{sigmoid}(\exp(z_{t_i})) \tag{13}$$

$$\Delta t_i = (2\cdot u_{t_i} - 1)\cdot(\Delta t_{ie} - \Delta t_{is}) + \Delta t_{is} \tag{14}$$

where Δt_{is} and Δt_{ie} are temporal components defined within the factorized constraint C_i obtained using Algorithm 1, ensuring that sampled timestamps always fall within the desired range.

Spatial Distribution Reparameterization. Similar to temporal distribution modeling, for spatial distribution, we adopt the 2-D Gaussian mixture model with a MLP layer:

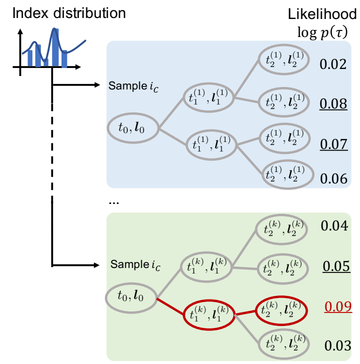

Fig. 2. Illustration of the beam decoding procedure. In this example, $B = 2$

$$(\boldsymbol{\mu}_{\boldsymbol{l}_i,1:M}, \Sigma_{\boldsymbol{l}_i,1:M}, w_{\boldsymbol{l}_i,1:M}) = \text{MLP}_{\boldsymbol{l}}(\boldsymbol{h}_i) \tag{15}$$

where the Gaussian mixture model is defined as:

$$\mathcal{G}_{\boldsymbol{l}} = \sum_{m=1}^{M} \frac{w_{\boldsymbol{l},m}}{2\pi\sqrt{|\Sigma_{\boldsymbol{l},m}|}} \exp\left(-\frac{(\boldsymbol{z}_{\boldsymbol{l}}-\boldsymbol{\mu}_{\boldsymbol{l},m})^T \Sigma_{\boldsymbol{l},m}^{-1} (\boldsymbol{z}_{\boldsymbol{l}}-\boldsymbol{\mu}_{\boldsymbol{l},m})}{2}\right) \tag{16}$$

$$\boldsymbol{z}_{\boldsymbol{l}_i} \sim \mathcal{G}_{\boldsymbol{l}}(\boldsymbol{\mu}_{\boldsymbol{l}_i,1:M}, \Sigma_{\boldsymbol{l}_i,1:M}, w_{\boldsymbol{l}_i,1:M}) \tag{17}$$

In a similar manner, the constrained spatial distribution can be mapped from the unconstrained GMM distribution through differentiable transformations including a sigmoid and an affine operator as follows:

$$\boldsymbol{u}_{\boldsymbol{l}_i} = \text{sigmoid}(\boldsymbol{z}_{\boldsymbol{l}_i}) \tag{18}$$

$$x_i = [u_{\boldsymbol{l}_i}]_1 \cdot (x_{ie} - x_{is}) + x_{is}, \quad y_i = [u_{\boldsymbol{l}_i}]_2 \cdot (y_{ie} - y_{is}) + y_{is} \tag{19}$$

where x_{is}, x_{ie}, y_{is}, and y_{ie} are spatial components defined within the factorized constraint C_i.

D. Generation

During generation, each trajectory in Geo-CETRA follows a spatiotemporal constraint. Since optimizing $p(\tau)$ in continuous space-time is computationally expensive, we use a beam search inspired by text generation to improve generation quality. As shown in Fig 2, we sample K different i_C values from the constrained index distribution and apply a beam width of B within each group. For each i_C, the model samples B candidate visits at each timestamp, ranking them based on the sum of temporal and spatial log-likelihoods. We select the top-B candidates and use Monte Carlo sampling to repeat this process K times. The final trajectory is chosen as the candidate with the highest likelihood from each group (red in Fig 2).

IV. Experiments

A. Dataset and Preparation

We evaluate the performance of our proposed model using two datasets, the Houston and MobilitySim datasets. The Houston dataset is sampled from Veraset [1], which provides movement data collected through GPS signals from approximately 10% of cell phones across the US. The MobilitySim is a realistic simulation of 10,000 agents performing daily activities in US for a week. each trajectory contains four primary data fields: an anonymous user ID, latitude, longitude, and a timestamp for a handful of visits per day. For each real trajectory, we generate spatiotemporal constraints by randomly selecting a visit along the trajectory, and then create a constrained time window and spatial range that only encompasses the selected point. This method ensures that the spatiotemporal constraints match the real data distribution and incorporate realistic human mobility information.

B. Baseline Methods

We compare the performance of our model with the following state-of-the-art baselines, including **MC$_{\textbf{First-Order}}$** [20] which defines all visited locations as states and builds a transition matrix to capture the next transition probabilities between them. **GRU$_{\textbf{pred}}$** [21] a type of recurrent neural network (RNN) architecture commonly used as a baseline in various sequential data analysis tasks. **Transformer$_{\textbf{decoder}}$** [22] Transformer is a powerful deep-learning model commonly used in various NLP and computer vision tasks. **Transformer$_{\textbf{VAE}}$** [14], [23] A variational autoencoder (VAE)-based generative model where the encoder and decoder are designed with the Transformer architecture. **TrajGAN** [24] is a CNN-based structure within a GAN framework. It converts the trajectories into 2D matrices where the value of each cell contains information about the time and duration of visiting that cell in the given trajectory. **MoveSim** [25] adopts SeqGAN, a sequence generative adversarial network and reinforcement learning to generate the next location based on past states.

C. Evaluation Metrics

We adopt the following 5 evaluation metrics commonly used in prior research to evaluate the quality of generated data for statistical similarity [5], [25]: Length, Cum Distance, Move Distance, Radius and $P(r_1, r_2)$. We use the Jensen–Shannon divergence (JSD) to measure the similarity between the mobility pattern distributions of generated trajectory and real-world trajectory data. Lower JSD indicates a better generation result.

D. Overall Performance

The performance of our Geo-CETRA and baseline methods on the Houston and MobilitySim datasets is detailed in Table I. Our model not only achieves the best performance across almost most usability metrics but is also the only model capable of integrating and enforcing spatiotemporal constraints. For a fair comparison, we also report the performance of our model without constraints in Table I. The evaluation shows that it outperforms all baselines in nearly all metrics on both datasets, highlighting the importance of capturing irregularly timed visits in continuous time and space. Finally, all baselines can only handle discrete trajectory data, which fails to accurately model real-world trajectories with irregular movement patterns. Furthermore, it is important to note that even though our method adheres to constraints, while the others do not, our method most often outperforms the others. With realistic constraints enforced, Geo-CETRA$_{\text{constraint}}$ achieves superior performance, with up to 26% improvement in distance metrics even compared to our unconstrained model, demonstrating the effectiveness of our approach in generating realistic trajectories.

[1] https://www.veraset.com/about-veraset

V. Related Work

The problem of trajectory generation has a long history. Traditional methods simulate human movement using physical parameters from surveys and sensors (e.g., propensity to travel or not), but these require extensive programming efforts [26]. The current emerging trend for trajectory generation is to apply deep generative models to learn the latent representation of trajectories directly from the data. one type of this work converts location traces into two-dimensional images and uses CNN-based GANs for generation [24]. Another approach applies the GAN structure and integrates prior knowledge of the urban structure and the pre-defined mobility regularities of spatial continuity and temporal periodicity to generate realistic trajectories [25]. Despite the progress, there are still two main limitations existing in current approaches. First, prior research on trajectory generation often projects GPS coordinates onto discrete geographical grids and time intervals, while in reality, each spatiotemporal point exists in continuous time and space. Recent works demonstrate that modeling trajectories in a continuous space can more accurately model real-world trajectories with irregular moving patterns [8]. Second, the generation frameworks that generalize to arbitrary spatiotemporal constraints are seriously under-explored yet imperative. A recent study by [27] also addresses the problem of constraint trajectory generation, but our work differs in key aspects. In [27], the constraints are defined as a set of known visits that the synthetic trajectory must pass through, whereas we define constraints as spatiotemporal ranges the trajectory must satisfy. Moreover, [27] discretizes space into grid cells to build a “vocabulary” to fill in the visits between these fixed points using a language-model-inspired approach, while our method operates directly in a continuous spatiotemporal space, aligning with our flexible constraint definition. Therefore, our proposed framework address the above two challenges and generates high-quality synthetic trajectories in continuous space while effectively enforcing the satisfaction of given spatiotemporal constraints.

VI. Conclusion

Modeling human movement is key to many applications, and spatiotemporal constraints are essential for generating

Dataset	Model	Length ↓ (JSD)	Cum Distance ↓ (JSD)	Move Distance ↓ (JSD)	Radius ↓ (JSD)	$P(r_1, r_2)$ ↓ (MSE)	Constraint Satisfied
Houston	$\text{MC}_{\text{First-Order}}$	0.3256	0.2386	0.2078	0.1388	**0.0418**	✗
	GRU_{pred}	0.0497	<u>0.1225</u>	0.1420	<u>0.0126</u>	0.0788	✗
	$\text{Transformer}_{\text{decoder}}$	0.0554	0.1883	0.1905	0.0987	0.0798	✗
	$\text{Transformer}_{\text{VAE}}$	0.4931	0.5991	0.3013	0.1510	0.1188	✗
	TrajGAN	0.2532	0.3726	0.2499	0.1329	0.0933	✗
	MoveSim	0.0758	0.1657	0.1502	0.0152	0.0657	✗
	$\text{Geo-CETRA}_{\text{no constraint}}$	**0.0390**	0.1242	<u>0.1392</u>	0.0141	<u>0.0501</u>	✗
	$\text{Geo-CETRA}_{\text{constraint}}$	<u>0.0402</u>	**0.1126**	**0.1021**	**0.0107**	0.0502	✓
MobilitySim	$\text{MC}_{\text{First-Order}}$	0.2352	0.1511	0.1941	0.0180	**0.0401**	✗
	GRU_{pred}	0.0677	0.0803	0.1541	<u>0.0122</u>	0.0595	✗
	$\text{Transformer}_{\text{decoder}}$	0.1326	0.1299	0.1564	0.0526	0.0649	✗
	$\text{Transformer}_{\text{VAE}}$	0.3006	0.2270	0.1952	0.0711	0.1050	✗
	TrajGAN	0.2037	0.1321	0.1488	0.0419	0.0867	✗
	MoveSim	0.0853	0.1234	0.1594	0.0132	0.0544	✗
	$\text{Geo-CETRA}_{\text{no constraint}}$	<u>0.0452</u>	<u>0.0885</u>	<u>0.1341</u>	0.0125	0.0479	✗
	$\text{Geo-CETRA}_{\text{constraint}}$	**0.0426**	**0.0673**	**0.1016**	**0.0120**	<u>0.0431</u>	✓

TABLE I
PERFORMANCE COMPARISON OF OUR MODEL AND BASELINES ON TWO MOBILITY DATASETS, WHERE THE LOWER VALUE INDICATES BETTER PERFORMANCE. BOLD DENOTES THE BEST(LOWEST) RESULTS AND THE UNDERLINE DENOTES THE SECOND-BEST RESULTS.

realistic synthetic trajectories. In this paper, we introduce Geo-CETRA, a framework that generates high-quality trajectories in continuous space-time while enforcing these constraints. This model employs a temporal and spatial encoder with random Fourier features and enforces constraints by reparameterizing the distribution. We also developed a new decoding module to refine control over the generation and enhance overall performance. Extensive experiments demonstrate the superior performance of our framework in generating synthetic trajectories.

VII. ACKNOWLEDGEMENT

Research supported by the Intelligence Advanced Research Projects Activity (IARPA) via the Department of Interior/Interior Business Center (DOI/IBC) contract number 140D0423C0033. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes, notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA or the U.S. Government. This research project has benefitted from the Microsoft Accelerate Foundation Models Research (AFMR) grant program through which leading foundation models hosted by Microsoft Azure along with access to Azure credits were provided to conduct the research.

REFERENCES

[1] Y. Chen, C. Hu, and J. Wang, "Human-centered trajectory tracking control for autonomous vehicles with driver cut-in behavior prediction," *IEEE Transactions on Vehicular Technology*, 2019.
[2] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *WWW*, 2009.
[3] L. Zhang and et al., "Factorized deep generative models for trajectory generation with spatiotemporal-validity constraints," *SIGSPATIAL*, 2021.
[4] H. Lin and et al., "Generating realistic and representative trajectories with mobility behavior clustering," in *SIGSPATIAL*, 2023.
[5] M. Zhang and others., "Csgan: Modality-aware trajectory generation via clustering-based sequence gan," in *MDM*. IEEE, 2023.
[6] M. Yue and et al., "Detect: Deep trajectory clustering for mobility-behavior analysis," *Big Data*, 2019.
[7] R. T. Chen, B. Amos, and M. Nickel, "Neural spatio-temporal point processes," *ICLR*, 2021.
[8] H. Lin *et al.*, "Unified modeling and clustering of mobility trajectories with spatiotemporal point processes," in *SDM*, 2024.
[9] B. Krause *et al.*, "Gedi: Generative discriminator guided sequence generation," *arXiv preprint arXiv:2009.06367*, 2020.
[10] N. S. Keskar *et al.*, "Ctrl: A conditional transformer language model for controllable generation," *arXiv preprint arXiv:1909.05858*, 2019.
[11] X. Lu *et al.*, "Neurologic decoding:(un) supervised neural text generation with predicate logic constraints," *arXiv preprint arXiv:2010.12884*, 2020.
[12] J.-C. Régin, "A filtering algorithm for constraints of difference in csps," in *AAAI*, vol. 94, 1994, pp. 362–367.
[13] S. Zuo and et al., "Transformer hawkes process," in *ICML*, 2020.
[14] A. Vaswani *et al.*, "Attention is all you need," in *NeurIPS*, 2017.
[15] V. V. Cepeda, G. K. Nayak, and M. Shah, "Geoclip: Clip-inspired alignment between locations and images for effective worldwide geo-localization," *arXiv preprint arXiv:2309.16020*, 2023.
[16] M. Tancik *et al.*, "Fourier features let networks learn high frequency functions in low dimensional domains," *NeurIPS*, 2020.
[17] M. Hu *et al.*, "Self-supervised pre-training for robust and generic spatial-temporal representations," in *ICDM*. IEEE, 2023.
[18] Y. Yuan and others., "Activity trajectory generation via modeling spatiotemporal dynamics," in *KDD*, 2022.
[19] O. Shchur, M. Biloš, and S. Günnemann, "Intensity-free learning of temporal point processes," *arXiv preprint arXiv:1909.12127*, 2019.
[20] S. Gambs *et al.*, "Next place prediction using mobility markov chains," in *Proceedings of the first workshop on measurement, privacy, and mobility*, 2012, pp. 1–6.
[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998.
[22] P. J. Liu *et al.*, "Generating wikipedia by summarizing long sequences," *arXiv preprint arXiv:1801.10198*, 2018.
[23] N. Xu *et al.*, "Simulating continuous-time human mobility trajectories," *Deep Learning For Simulation Workshop, ICLR*, 2021.
[24] K. Ouyang, R. Shokri, D. S. Rosenblum, and W. Yang, "A non-parametric generative model for human trajectories." in *IJCAI*, 2018.
[25] J. Feng and others., "Learning to simulate human mobility," in *KDD*, 2020.
[26] S. Jiang *et al.*, "The timegeo modeling framework for urban mobility without travel surveys," in *PNAS*, 2016.
[27] S.-L. Hsu *et al.*, "Trajgpt: Controlled synthetic trajectory generation using a multitask transformer-based spatiotemporal model," in *SIGSPATIAL*, 2024.