Deep Trajectory Recovery Approach of Offline Vehicles in the Internet of Vehicles

Xiao Han, Ding-Xuan Zhou, Guojiang Shen, Xiangjie Kong, *Senior Member, IEEE*, and Yulong Zhao

***Abstract*—Multi-modal trajectory analysis and trajectory recovery are essential tasks in transportation research, especially for offline vehicles, which enable comprehensive understanding of complex transportation systems and address the issue of incomplete or missing trajectory data. In this paper, we propose a novel Deep Trajectory Recovery Framework, DTRF, which can effectively tackle both challenges by using a combination of a Cellular Automata (CA) model and a Multi-Kernel Graph Neural Network (MKGNN) model. The CA model plays a crucial role in normalizing and representing multi-modal traffic data with diverse structures, sampling frequencies, and physical meanings. By capturing the inherent relationships among different modalities, the CA model enables our proposed framework to make better use of these multi-modal data from networked vehicles and roadside detectors and then generate data for traditional vehicles. The MKGNN model, built on the foundation of spectral graph theory, employs various kernels to model different driving characteristics. The use of multiple kernels allows the GNN model to capture a wide range of driving patterns, enhancing its ability to reconstruct missing trajectories accurately. To validate the effectiveness of our proposed model, extensive experiments are conducted on two datasets. The results demonstrate that our framework outperforms state-of-the-art baselines in terms of trajectory recovery, showcasing its efficiency and robustness.**

***Index Terms*—Internet of Vehicles, spatiotemporal data processing, cellular automata, trajectory recovery, graph neural networks.**

I. Introduction

With the emergence of Internet of Vehicle (IoV) technology, urban traffic has entered a new phase where networked vehicles and conventional vehicles are integrated [1], [2], [3], [4], [5]. One of the key research areas in this field is the recovery and reconstruction of missing trajectories [6], [7], [8]. It involves analyzing vehicle movement patterns, driving paths, and behavioral characteristics to fill in the gaps in trajectory data.

By improving data completeness and quality, this process enhances performances in various localization-based downstream tasks, such as vehicle travel time estimating, which demands high precision regarding vehicle availability within specific geographical zones.

Therefore, it is crucial to develop a comprehensive trajectory recovery model for all vehicles in the road network. It could provide us with a deeper understanding of traffic dynamics, and enable better insights into traffic patterns and behaviors [9], [10].

However, trajectory recovery of conventional vehicles is even more difficult, with two core challenges: a) *trajectory collection and representation*, and b) *identifying and stitching noncontiguous trajectory segments*. On the one hand, conventional vehicles lack the ability to perceive their own locations in contrast to intelligent networked vehicles and rely on roadside detectors and surrounding vehicles to assist with their positioning. Fig. 1(a) illustrates that different types of traffic data can be generated by the deep purple vehicles equipped with in-vehicle detectors, allowing modeling trajectories of surrounding vehicles without such detectors. Given the variations in detection frequencies, data types from different detectors, and the diversity in types and sizes of vehicles detected, establishing uniform standards for representing vehicle trajectory data from multiple sources presents a significant challenge. On the other hand, conventional vehicles frequently encounter challenges related to missing data and data inconsistencies due to the passive nature of their data collection methods. Both Fig. 1(b) and (c) show the trajectory modeling of vehicles on road segments and intersections, respectively. Fig. 1(b) shows that each of the three different networked vehicles has detected a part of the trajectories of the surrounding vehicles, and it is necessary to fuse these trajectory segments and restore the complete trajectory of the yellow conventional vehicle. Fig. 1(c) shows that the blue vehicle in the white dotted box is blocked by the bus from the view in this figure. If the detectors from other angles fail to function their trajectories properly, it is also hard to restore the accurate trajectory of this blue vehicle during the blocking period. Therefore, designing an algorithm that can effectively relocate the missing part of the vehicle trajectory and relink the segmented trajectories is another major challenge.

Some studies have attempted to address some of those challenges. Many researchers [11], [12], [13] have redefined the vehicle trajectory recovery issue as a sequence-completion task by simplifying the vehicle trajectory into a time-ordered sequence

Manuscript received 10 January 2024; revised 28 May 2024; accepted 1 July 2024. Date of publication 4 July 2024; date of current version 7 November 2024. The work of Ding-Xuan Zhou was supported by the Discovery Project of the Australian Research Council under Project DP240101919. The review of this article was coordinated by Prof. Shahid Mumtaz. *(Corresponding author: Yulong Zhao.)*

Xiao Han is with the School of Data Science, City University of Hong Kong, Hong Kong (e-mail: hahahenha@gmail.com).

Ding-Xuan Zhou is with the School of Mathematics and Statistics, University of Sydney, Sydney, NSW 2050, Australia (e-mail: dingxuan.zhou@sydney.edu.au).

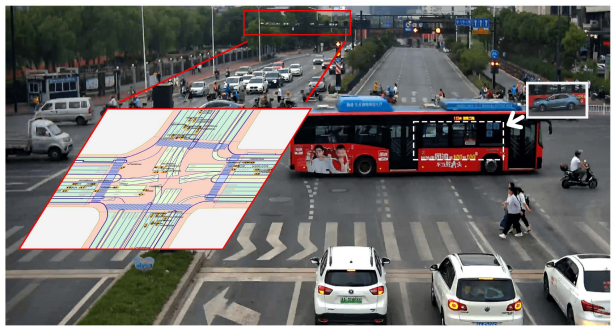
Guojiang Shen and Xiangjie Kong are with the College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China (e-mail: gjshen1975@zjut.edu.cn; xjkong@ieee.org).

Yulong Zhao is with the AI and Trust Technologies Division, Hong Kong Applied Science and Technology Research Institute, Hong Kong (e-mail: zhaoyulong@gmail.com).

Digital Object Identifier 10.1109/TVT.2024.3423348

(a) The scenario for the collected real-time data from networked vehicles with in-vehicle sensors

(b) The scenario for auxiliary recognition of conventional trajectory



(c) The scenario for static detectors at an intersection

Fig. 1. An example of the IoV environment. (a) The scenario for the collected real-time data from networked vehicles with in-vehicle sensors. (b) The scenario for auxiliary recognition of conventional trajectory. (c) The scenario for static detectors at an intersection.

of latitude and longitude coordinates. These methods assume a uniform vehicle volume for all vehicles, which compromises the accuracy of representing the locations of diverse vehicle types. Furthermore, some studies [14] have introduced sophisticated lane-level trajectory recovery algorithms designed to reconstruct vehicle trajectories at a micro level. However, when multiple trajectory segments are present simultaneously in a fixed period, these methods struggle to accurately identify which ones originate from the same vehicle.

To address all the challenges mentioned above, in this paper, we propose a novel approach, DTRF, to present and recover vehicle trajectories accurately. Specifically, we present a road network division and representation method based on the Cellular Automata (CA) model to unify the data structure and the vehicle feature difference of multi-modal traffic data and use the spatiotemporal trajectory graphs to represent the trajectories of conventional vehicles. To efficiently and accurately extract different vehicle features for trajectory recovery, we quantify driving behaviors in the form of multi-scale graph representation learning and use a spectral-based graph analysis method to characterize the continuity of trajectories. To the best of our knowledge, we are the first to implement a new deep trajectory recovery approach combining the traffic flow model to solve the problems of trajectory recognition and repair of offline vehicles. Our contributions are demonstrated as follows:

- We propose a framework, DTRF, which uses CA to represent the multi-modal traffic data and incorporates a multi-kernel graph learning method to address the issue of low data quality of traditional vehicle trajectories. To our understanding, this is a novel approach that has not been explored before.
- We develop a Multiple Kernel Graph Neural Network (MKGNN) to choose the most appropriate combination of kernels for different vehicles based on their unique driving characteristics, which can significantly improve the representation ability and flexibility of trajectory locality. Besides, a quantitative result is given for the Chebyshev approximation of graph Laplacian operators in the MKGNN. It shows that only a small order of the Chebyshev polynomial can achieve an accurate approximation, and our experimental results further verify it.
- Extensive experiments based on both simulated and real-world datasets demonstrate the effectiveness and efficiency of the framework DTRF. Besides, several ablation studies have been performed to illustrate the impact of different hyperparameters on the model more precisely.

The rest of this paper is organized as follows: Section II briefly reviews related work. In Section III, we detail the framework of DTRF, including the proposed CA model and multi-kernel graph neural network, and introduce the training and testing process of the DTRF. Section IV carries out experiments based on both the simulated environment and the real-world dataset and presents experimental results accordingly. Section V concludes this work and discusses future work.

II. Related Work

In this section, we briefly review some recent work on multi-modal trajectory analysis and data recovery models.

A. Multi-Modal Trajectory Representation

Multi-modal trajectory analysis is a popular and cross-cutting research topic in the fields of computer vision, deep learning, and transportation. It has gained significant attention in recent years, as it enables more comprehensive understanding of complex transportation systems and human mobility patterns.

Several studies have explored methods to merge trajectory data from different sources (i.e., GPS, in-vehicle sensors, roadside cameras), which is crucial to obtain a holistic view of an individual's or a population's movement patterns. Luo et al. [15] proposes a novel instance-aware representation for lane representation. In their work, a goal-oriented lane attention module is proposed to accurately predict the future locations of the vehicle by integrating the lane features and trajectory features. Yin et al. [16] proposes a novel solution for predicting future trajectories of pedestrians, which uses a multimodal encoder-decoder transformer architecture. It predicts the entire future trajectory in a single-pass, which makes the method effective for embedded edge deployment. Hu et al. [17] presents a multi-modal probabilistic prediction approach to predict the sequential motions of each pair of interacting agents. They address the challenge of

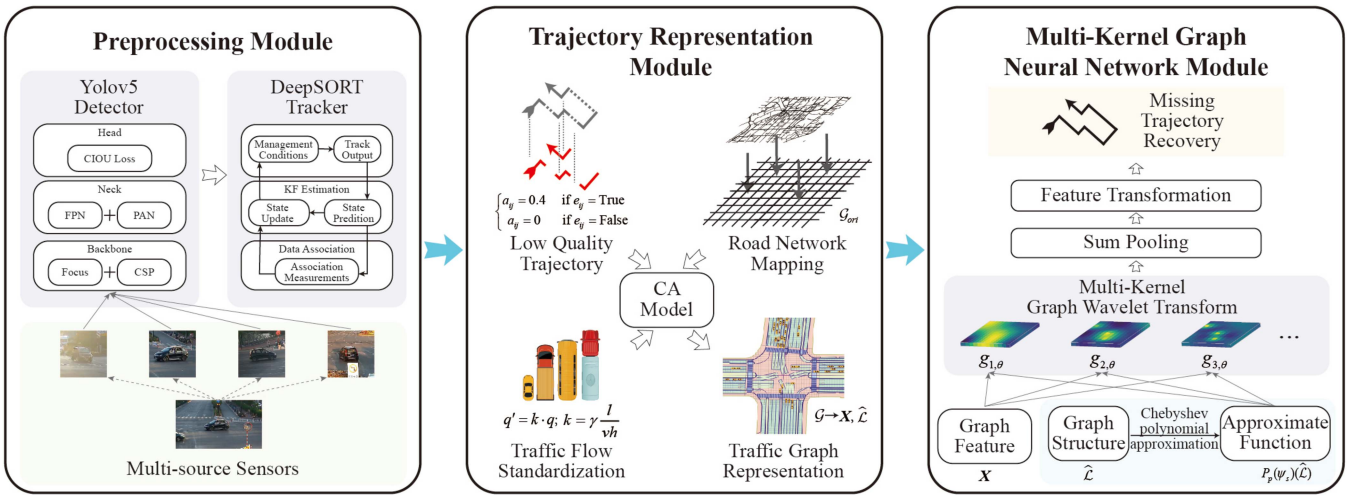


Fig. 2. The framework overview. It contains preprocessing, trajectory representation, and MKGNN modules. The first module combines video from various angles to create vehicle paths. The second one employs the CA model to build vehicle graphs in the road network. Lastly, the MKGNN module extracts data from the graph for trajectory recovery.

modeling prediction uncertainties and multi-modal distributions over future sequences efficiently.

However, most traditional vehicles do not have any sensors, and it's hard for us to collect accurate data from these vehicles. One possible solution to this problem is to fuse and analyze multi-source data from other networked vehicles while extrapolating the trajectories of traditional vehicles around these networked vehicles. Notice that the multi-modal traffic data generated by each vehicle has the characteristics of non-uniform dimensions, different sampling frequencies, and different physical meanings. Therefore, it is difficult for us to utilize these traffic features directly and effectively. We urgently need to design a method to normalize these data features and improve data availability.

B. Trajectory Recovery

Trajectory recovery refers to the task of reconstructing or inferring missing portions of trajectory data. It is a critical issue in trajectory analysis, as incomplete or missing data can significantly impact the accuracy and reliability of subsequent analysis tasks. With the recent advancements in machine learning and deep learning, several learning-based methods have been developed for it. Wang et al. [18] address the problem of recovering a high-sampled trajectory from a low-sampled one and propose a novel deep hybrid model, named DHTR, that combines a subsequence-to-sequence recovery model (subseq2seq) with a Kalman filter (KF) component for trajectory calibration. Xia et al. [19] proposes a novel attentional neural network-based model, AttnMove, that leverages both intra- and inter-trajectory attention mechanisms to capture the mobility regularity and periodicity of users. Ren et al. [20] introduces a novel sequence-to-sequence (Seq2Seq) multi-task learning framework, MTrajRec, to recover the fine-grained points in trajectories and match them on the road network in an end-to-end manner. These methods leverage the patterns and relationships present in the available trajectory data to predict the missing segments. However, these methods usually use discrete trajectory points to represent the vehicle's trajectory and ignore the characteristics of the space occupied by the vehicle itself in the road network and the continuity of the trajectory in the time dimension. Therefore, we need to design a method that can describe the shape differences and trajectory continuity of different types of vehicles, and restore the vehicle trajectory more accurately.

III. Proposed Method

In this section, we provide a detailed overview of our proposed framework. We begin by providing a brief introduction to the general process structure. Next, a CA-based graph representation method of the multi-modal trajectory data is proposed. Finally, the MKGNN is introduced to recover high-accuracy trajectories.

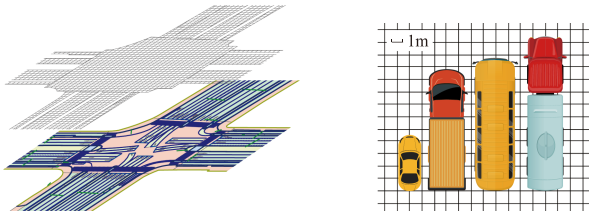
A. Overview

Fig. 2 shows the overall framework, which consists of a CNN-based preprocessing model, one CA-based spatiotemporal trajectory representation, and a multi-kernel graph neural network.

The preprocessing model in our framework utilizes YOLOv5 [21] + DeepSORT [22], a popular and widely used combination in engineering to extract coarse-grained conventional vehicle trajectory segments from multimodal traffic data. Details of the preprocessing model can be found in the experiments section.

In the CA module, we quantitatively model the characteristics of various vehicles within microscopic traffic flows using traffic flow theory. This approach ensures that the generated vehicle trajectory maps are consistent in terms of frequency, dimension, and physical meaning, thereby maintaining uniformity across the dataset.

In the graph neural network module, a spectral-based GNN is designed rather than a spatial one (e.g., message passing



(a) The representation of the road network (b) Proportion of different types of vehicles

Fig. 3. The representation of traffic data. (a) The representation of the road network. (b) Proportion of different types of vehicles.

network) to efficiently analyze the correlation between any nodes via a single calculation instead of relying on multiple aggregation operations. To facilitate this, we introduce the graph Laplacian matrix for spectral decomposition. Additionally, we implement the Chebyshev polynomial approximation to minimize the computational overhead associated with the eigen-decomposition of the graph Laplacian matrix. Subsequently, a multi-kernel filter is used to categorize different driving behaviors, and trajectory segments exhibiting similar behaviors are linked to accurately reconstruct complete trajectories.

B. CA-Based Spatiotemporal Trajectory Representation

Traditionally, when modeling traffic trajectories, GPS points arranged in a time series are commonly used to represent vehicle trajectories. However, this approach overlooks the influence of vehicle size and its interaction with the road network. Particularly, when capturing fine-grained micro-traffic behaviors like lane changes and overtaking, different types of vehicles cannot be adequately represented as individual points. To address this issue, we propose a CA-based method in this section. This method offers a more accurate representation of each component of city traffic. By incorporating CA principles, we can capture the dynamic interactions between vehicles and their surroundings, considering factors such as vehicle size, lane occupancy, and traffic flow. This approach provides more comprehensive and detailed understanding of the complex dynamics within urban traffic scenarios.

First, we represent the road network at the lane level as a grid-partitioned graph, as shown in Fig. 3(a). We treat each grid as a node $v \in \boldsymbol{V}$ in the graph $\mathcal{G}$, and each grid has edges $e \in \boldsymbol{E}$ to the adjacent grids. We can represent the road network in the form of an adjacency matrix $\boldsymbol{A} = \{a_{ij}\}_{i,j\in[N]}$:

$$\begin{cases} a_{ij} = 0.4 & \text{if } e_{ij} = \text{True}, \\ a_{ij} = 0 & \text{if } e_{ij} = \text{False}, \end{cases} \tag{1}$$

where $e_{ij} =$ True when there is an edge between nodes i and j, N is the total number of nodes in the graph.

Due to the different specifications of buses, trucks, cars, and other vehicles, there are differences in volume and traffic speed. Therefore, it becomes essential to convert them into a standardized metric for statistical purposes and related derived parameters. This standardization is achieved by converting the vehicles into Passenger Car Units (PCUs). As shown in Fig. 3(b), we illustrate the process of PCU conversions for four common types of vehicles: cars, pickups, buses, and heavy trucks. By applying the CA model, we analyze the traffic characteristics specific to each vehicle type. This analysis allows us to gain valuable insights into the behavior and patterns exhibited by different vehicle types in traffic scenarios. Take

$$\begin{aligned} q' &= k \cdot q, \\ k &= \gamma \frac{l}{vh}, \end{aligned} \tag{2}$$

where q stands for vehicle features, l (n/veh) is the distance between the fronts of two adjacent vehicles in saturated traffic, v (m/s) is the average driving speed when saturated traffic flow passes through the intersection, h (s/veh) is the headway time of standard car saturated traffic, γ is a correction factor which uses to correct different flow directions or special traffic behavior, and k is the conversion factor. Due to the randomness of the behavior of vehicles in practice, a large amount of data investigation is usually required to calibrate a reasonable correction factor γ.

Next, we map the trajectories into the road network graph as training samples. For each vehicle, first, we map each GPS point with traffic features in the trajectory to several connected regions:

$$f_{\mathcal{G}}(\tau_a) = \{v_i : v_i \in \mathcal{N}(a)\}, \tag{3}$$

where $\mathcal{N}(a)$ is a set containing all nodes next to the node a.

Moreover, we update the graph in a sample by connecting the chronological trajectory $\Gamma = \{\tau_1, \ldots, \tau_t, \ldots, \tau_T\}$ though weighted edges:

$$\begin{cases} a'_{ij} = a_{ij} + 0.6, & \text{if } \tau_t, \tau_{t+1} \in \Gamma; v_i, v_j \in f_{\mathcal{G}}(\tau_t), \\ a'_{ij} = a_{ij}, & \text{if } \tau_t \text{ or } \tau_{t+1} \notin \Gamma; v_i, v_j \in f_{\mathcal{G}}(\tau_t). \end{cases} \tag{4}$$

Then we can obtain an adjacency matrix of each sample $\boldsymbol{A}' = \{a'_{ij}\}$, $i, j \in [N]$. We take the timestamp when a vehicle enters the area corresponding to the node, the time period it stays, the average speed and acceleration of the vehicle, and the traffic flow in the area under the current time slice to constitute the static feature of the node Fea_{static}. Moreover, the preprocessing model generates two types of features, $Fea_{\text{sensor(v)}}$ and $Fea_{\text{sensor(r)}}$, which are obtained from in-vehicle data and data collected by roadside detectors, respectively. These features also need to be mapped onto the road network graph for further analysis and processing. Thus, the feature matrix of the updated graph is defined as the matrix form of all node features $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^\top =$ concat$(Fea_{\text{sensor(v)}}, Fea_{\text{sensor(r)}}, Fea_{\text{static}})$, where $Fea_{\text{sensor(v)}} \in \mathbb{R}^{N\times M_v}$ and $Fea_{\text{sensor(r)}} \in \mathbb{R}^{N\times M_r}$, and $Fea_{\text{static}} \in \mathbb{R}^{N\times M_s}$. For each sample, we construct an adjacency matrix and a feature matrix as the input of the downstream trajectory recovery task. In this paper, we consider this task as a node classification problem whether a vehicle is passed through a specific grid.

C. MKGNN Model

This section presents an MKGNN to restore a missing trajectory. As spectral analysis methods can effectively process continuous trajectory characteristics, we adopt the spectral graph convolution method to achieve better trajectory recovery. Besides,

to model various driving characteristics of different trajectories accurately, we deploy a ChebyNet approximation-based graph wavelet transform, and we provide a quantitative result for this approximation.

Before introducing the proposed MKGNN, we calculate the Laplacian matrix $\mathcal{L} = \boldsymbol{D} - \boldsymbol{A}$ and then normalize it, as shown in (5). This matrix $\hat{\mathcal{L}}$ has very good properties [23], [24], [25]: all of its eigenvalues satisfy $\lambda \in [0, 2]$ with $\lambda_{\max} \approx 2$, where

$$\hat{\mathcal{L}} = \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{A}' \boldsymbol{D}^{-\frac{1}{2}}, \tag{5}$$

and $\boldsymbol{D} = \text{diag}\{d_1, \ldots, d_N\}$ is the graph degree matrix.

The graph wavelet transform based on a mother kernel $\phi : \mathbb{R}^+ \to \mathbb{R}^+$ defines a convolution operator at scale s as $T_\phi^s := \phi(\hat{\mathcal{L}}/s)$ by spectral calculus. In this paper, the mother kernel is chosen to be:

$$\phi_\lambda = \exp\left(\frac{2\lambda}{\lambda_{\max}} - 1\right). \tag{6}$$

Then the scaling function related to this mother kernel is defined as

$$\psi_s(\lambda) = \exp\left(\frac{2\lambda - s\lambda_{\max}}{s\lambda_{\max}}\right), \tag{7}$$

where $s > 0$ is the scaling factor.

This function is used to generate the kernel set of our model, which is used to model various driving characteristics of different trajectories more accurately. The multi-kernel graph Fourier Transform operator can be defined by spectral calculus as

$$\psi_s(\hat{\mathcal{L}}) = \psi_s\left(\mathbf{U}\Lambda\mathbf{U}^{\mathrm{T}}\right) = \mathbf{U}\text{diag}(\psi_s(\lambda_i))_{i=1}^n \mathbf{U}^{\mathrm{T}}. \tag{8}$$

In the same way, the inverse of this operator can be defined as:

$$\psi_s^{-1}(\hat{\mathcal{L}}) = \mathbf{U}\text{diag}\left(\psi_s^{-1}(\lambda_i)\right)_{i=1}^n \mathbf{U}^{\mathrm{T}}. \tag{9}$$

Taking a trainable weight matrix $\mathbf{g}_{s,\theta} = \text{diag}(\theta_{s_1}, \ldots, \theta_{s_S})$ and a frequency sample set $(s_i)_{i=1}^S \subset \mathbb{R}^+$, we define the graph convolution in the MKGNN layer as

$$\mathbf{x} *_G \mathbf{g}_\theta = \frac{1}{S}\sum_{i=1}^{S} \psi_{s_i}(\hat{\mathcal{L}})\mathbf{g}_{s_i,\theta}\psi_{s_i}^{-1}(\hat{\mathcal{L}})\mathbf{x}. \tag{10}$$

Under the settings of semi-supervised classification problems for a graph dataset, our task is to classify nodes with unknown labels according to the nodes with known labels and the characteristics of all nodes in the graph. The graph convolution operators ψ_s and ψ_s^{-1} only depend on the graph once s is given and we can calculate them as a preprocessing process before the training of the neural network.

Here we use the p-th order Chebyshev polynomial to approximate $\psi_s(\hat{\mathcal{L}})$ and $\psi_s^{-1}(\hat{\mathcal{L}})$. For a function g: $\mathbb{R}^+ \to \mathbb{R}^+$, the operator $g(\hat{\mathcal{L}}) = \mathbf{U}(g(\lambda_i))_{i=1}^n \mathbf{U}^{\mathrm{T}}$ is given by spectral calculus and the p-th order Chebyshev approximation is

$$P_p(g)(\hat{\mathcal{L}}) = \sum_{i=0}^{p} c_i(g) T_i(\hat{\mathcal{L}}), \tag{11}$$

where the Chebyshev polynomial $\{T_i\}_{i=0}^p$ is given by $T_0(x) = 1, T_1(x) = x$ and $T_i(x) = 2uT_{i-1}(x) - T_{i-2}(x)$ iteratively. The Chebyshev coefficients $\{c_i(g)\}_{i=0}^p$ are given by

$$c_i(g) = \frac{2}{N+1}\sum_{j=0}^{N} g(u_j) T_i'(u_j), \tag{12}$$

with $u_j = \cos\frac{(2j+1)\pi}{2N+2}$ for $j = 0, \ldots, N$, $T'(u) = T(\frac{u+1}{2}\lambda_{\max})$.

In our proposed method, we approximate $\psi_s(\hat{\mathcal{L}})$ by $P_p(\psi_s)(\hat{\mathcal{L}})$ and $\psi_s^{-1}(\hat{\mathcal{L}})$ by $P_p(\psi_s^{-1})(\hat{\mathcal{L}})$. Our first main result quantifies the rates of approximation in terms of the operator norm of the error operator $P_p(\psi_s)(\hat{\mathcal{L}}) - \psi_s(\hat{\mathcal{L}})$.

Theorem 1: For $s > 0$ and $p - 1 \in \mathbb{Z}_+$, we have

$$||P_p(\psi_s)(\hat{\mathcal{L}}) - \psi_s(\hat{\mathcal{L}})|| \le \frac{4}{\pi(p-1)(p-1)!s^{p-1}}\exp\left(\frac{1}{s}\right), \tag{13}$$

and

$$||P_p(\psi_s^{-1})(\hat{\mathcal{L}}) - \psi_s^{-1}(\hat{\mathcal{L}})|| \le \frac{4}{\pi(p-1)(p-1)!s^{p-1}}\exp\left(\frac{1}{s}\right). \tag{14}$$

Proof: Observe that $T_i'(\hat{\mathcal{L}}) = \mathbf{U}(T_i(\lambda_j'))_{j=1}^n \mathbf{U}^{\mathrm{T}}$, $\lambda' = \frac{2}{\lambda_{\max}}\lambda - 1 = \lambda - 1$. Then

$$\begin{aligned}
&P_p(\psi_s)(\hat{\mathcal{L}}) - \psi_s(\hat{\mathcal{L}}) \\
&= \mathbf{U}\left(\sum_{i=1}^{p} c_i(g) T_i(\lambda_j')\right)_{j=1}^n \mathbf{U}^{\mathrm{T}} - \mathbf{U}\left(\psi_s(\lambda_j')\right)_{j=1}^n \mathbf{U}^{\mathrm{T}} \\
&= \mathbf{U}\left(\left(P_p(\psi_s) - \psi_s\right)(\lambda_j')\right)_{j=1}^n \mathbf{U}^{\mathrm{T}},
\end{aligned} \tag{15}$$

where $P_p(\psi_s)(\lambda) = \sum_{i=1}^p c_i(\psi_s) T_i'(\lambda')$ is the p-th order Chebyshev polynomial approximation of the function ψ_s. Since $\mathbf{U}$ is an orthogonal matrix, we know that for any vector $v \in \mathbb{R}^n$ with norm $||v||$,

$$\begin{aligned}
&|| \left(P_p(\psi_s)(\hat{\mathcal{L}}) - \psi_s(\hat{\mathcal{L}})\right) v|| \\
&= ||\mathbf{U}\left(\left(P_p(\psi_s) - \psi_s\right)(\lambda_j')\right)_{j=1}^n \mathbf{U}^{\mathrm{T}} v|| \\
&= || \left(\left(P_p(\psi_s) - \psi_s\right)(\lambda_j')\right)_{j=1}^n \mathbf{U}^{\mathrm{T}} v|| \\
&= \left(\sum_{j=1}^{n} |\left(P_p(\psi_s) - \psi_s\right)(\lambda_j')|^2 |\left(\mathbf{U}^{\mathrm{T}} v\right)_j|^2\right)^{\frac{1}{2}} \\
&\le ||P_p(\psi_s) - \psi_s||_\infty ||\mathbf{U}^{\mathrm{T}} v|| \\
&= ||P_p(\psi_s) - \psi_s||_\infty ||v||.
\end{aligned} \tag{16}$$

Thus, $||(P_p(\psi_s)(\hat{\mathcal{L}}) - \psi_s(\hat{\mathcal{L}}))|| \le ||P_p(\psi_s) - \psi_s||_\infty$, where $||\cdot||_\infty$ is the norm in $C[-1, 1]$.

Suppose that f is absolutely continuous on $[-1, 1]$. Let q_n denote the Chebyshev interpolant f of degree n in the Chebyshev points of the first kind

$$y_j = \cos\left(\frac{(2j+1)\pi}{2n+2}\right), \quad j = 0, 1, \ldots, n. \tag{17}$$

The Chebyshev series for f is defined as [26], [27]:

$$f(x) = \sum_{j=0}^{\infty}{}' b_j T_j(x), \quad b_j = \frac{2}{\pi}\int_{-1}^{1}\frac{f(x)T_j(x)}{\sqrt{1-x^2}}dx, \tag{18}$$

where the prime denotes a sum whose first term is halved and $T_j(x) = \cos(j\cos^{-1}x)$ is the Chebyshev polynomial of degree j.

From Boyd [28], q_n can be expressed by

$$q_n(x) = \sum_{j=0}^{n}{}'' c_j T_j(x), \quad c_j = \frac{2}{n+1}\sum_{s=0}^{n} f(y_s)\, T_j(y_s), \tag{19}$$

where the double prime denotes a sum whose first and last terms are halved, and the coefficients c_j can be efficiently computed by Fast Fourier Transform (FFT).

Then an error bound for approximation of f in the Chebyshev points can be given, as shown in Lemma 2 [29].

Lemma 2: If $f, f', \ldots, f^{(k-1)}$ are absolutely continuous on $[-1, 1]$ and if $||f^{(k)}||_T = V_k < \infty$ for some $k \geq 1$, then for each $n \geq k+1$,

$$\frac{4V_k}{k\pi n(n-1)\cdots(n-k+1)} \geq ||f - q_n||_\infty, \tag{20}$$

If f is analytic with $|f(z)| \leq M$ in the region bounded by the ellipse with foci ± 1 and major and minor semiaxis lengths summing to $\rho > 1$, then for each $n \geq 0$,

$$||f - q_n||_\infty \leq \frac{4\tilde{}M}{(\rho-1)\rho^n}. \tag{21}$$

According to an estimate of the Chebyshev polynomial approximation [29], we have

$$||P_p(\psi_s) - \psi_s||_\infty \leq \frac{4V_{p-1}}{(p-1)\pi(p-1)!}, \tag{22}$$

where $V_{p-1} = ||\psi_s^{(p-1)}||_\infty$.

But $\psi_s^{(p-1)}(\lambda') = \frac{1}{s^{p-1}}\exp(\frac{\lambda'}{s})$, and $||\psi_s^{(p-1)}||_\infty = \max_{-1\leq\lambda'\leq 1}\frac{1}{s^{p-1}}\exp(\frac{\lambda'}{s}) = \frac{1}{s^{p-1}}\exp(\frac{1}{\tilde{}s})$. Hence

$$||P_p(\psi_s) - \psi_s||_\infty \leq \frac{4}{\pi(p-1)(p-1)!s^{p-1}}\exp\left(\frac{1}{\tilde{}s}\right). \tag{23}$$

The same result can also be found when deriving (14). This proves the theorem. □

Theorem 1 can be used to estimate the norm of the difference between the model network (10) and the approximation $\frac{1}{S}\sum_{i=1}^{S} P_p(\psi_s)(\hat{\mathcal{L}})\mathbf{g}_{s,\theta}P_p(\psi_s^{-1})(\hat{\mathcal{L}})\mathbf{x}$.

Finally, we define the k-th layer of the MKGNN as:

$$\mathbf{X}_{:,j}^{k+1} = \sigma\left(\sum_{i=1}^{d_k} w_{i,j}S^{-1}\sum_{s=1}^{S} P_p(\psi_s)(\hat{\mathcal{L}})\mathbf{g}_{s,\theta_{(j,i)}}^{k}P_p(\psi_s^{-1})(\hat{\mathcal{L}})\mathbf{X}_{:,i}^{k}\right), \tag{24}$$

where $j = 1, \ldots, d_k$, $w_{i,j} \in \mathbb{R}$ is a weight factor, $\mathbf{X}_{:,i}^{k}$ with dimensions d_k is the i-th column of $\mathbf{X}^k \in \mathbb{R}^{d_k \times n}$, $d_0 = d$ is the dimension of node signals, $\mathbf{g}_{s,\theta_{(j,i)}}^{k} \in \mathbb{R}^{n\times n}$ is a diagonal filter matrix to be learned in the spectral domain, σ is a non-linear activation function and we use ReLU function in practice. This layer transforms an input tensor $\mathbf{X}^k$ into an output tensor $\mathbf{X}^{k+1}$.

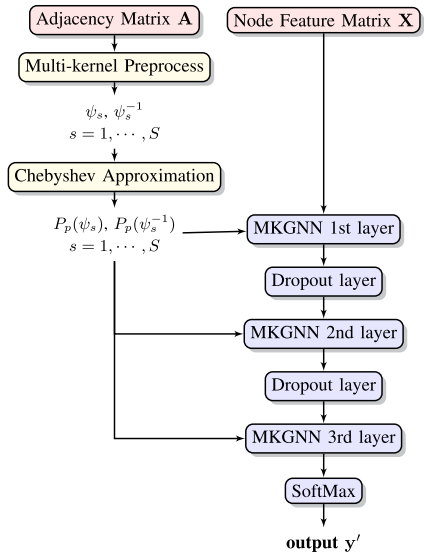


Fig. 4. The Structure of multi kernel graph neural network. It mainly comprises multikernel preprocessing, Chebyshev approximation, and three multi-kernel GNN layers.

With the inherent capability of wavelet kernels to facilitate progressive diffusion interaction of features among graph nodes, the MKGNN can effectively handle missing trajectories and seamlessly reconnect disjointed trajectory segments. By leveraging the wavelet kernels, MKGNN enables the completion of missing trajectory information and ensures a continuous and smooth reconstruction of each discrete trajectory.

In this paper, we adopt a three-layer network structure to achieve missing trajectory recovery, as shown in Fig. 4. At the same time, dropout layers are introduced to prevent overfitting, and cross-entropy loss function, which is analyzed in [30], is used to estimate the degree of inconsistency between the output $\mathbf{y}' := \{y_0', y_1'\}$ and the true label $\mathbf{y} := \{y_0, y_1\}$, as shown in (25).

$$\begin{aligned} Loss &= \frac{1}{n}\sum_{j=1}^{n} L_j \\ &= -\frac{1}{n}\sum_{j=1}^{n}\sum_{i\in\{0,1\}} y_i^{(j)}\log y_i'^{(j)}, \end{aligned} \tag{25}$$

where n is the sample size, m is the number of classes, $y_{ij} \in \{0, 1\}$ is a flag used to indicate whether the true category of the sample j is equal to the class i, $y'_{ij} \in [0, 1]$ is the predicted probability of belonging to class i in the output of the model about the sample j.

D. Inference Process

The process of inferring trajectory data is detailed in Algorithm 1. The initial step involves a preprocessing module, where video data are segmented into frames by channel. Within each frame, features and target positions are extracted using the Yolov5 algorithm, as specified in lines 1–3. For Lidar-generated point cloud data, voxelization is employed to convert the data into a 3D voxel format. This is then processed through a 3D

Algorithm 1: DTRF Framework.

Input: multi-view video $\boldsymbol{X}_V$ and Lidar point data $\boldsymbol{X}_L$ within a short period, road network graph $\mathcal{G}_{ori}$ with its static features Fea_{static}, CA and MKGNN module $\mathbf{CA}(\cdot)\&\mathbf{MKGNN}(\cdot,\cdot)$ in the proposed framework, pre-trained model Yolov5$_v(\cdot)$, Yolov5$_l(\cdot)$ for video and Lidar data preprocessing separately, and pre-trained DeepSORT$(\cdot)$ for vehicle ReID;

Output: The recovered trajectory: τ_{rec}.

1: **for** each frame $x_v^t \in \boldsymbol{X}_V$ **do**
2: $loc^t_{\text{sensor(v)}}, Fea^t_{\text{sensor(v)}} \leftarrow$ Yolov5$_v(x_v^t)$
3: **end for**
4: **for** each frame $x_l^t \in \boldsymbol{X}_L$ **do**
5: $x_v^{t\prime} =$ Voxelization(x_l^t)
6: $x_v^{t\prime\prime} =$ 3D-CNN$(x_v^{t\prime})$
7: $loc^t_{\text{sensor(l)}}, Fea^t_{\text{sensor(l)}} \leftarrow$ Yolov5$_l(x_l^{t\prime\prime})$
8: **end for**
9: $\tau_{ids} \leftarrow$ DeepSORT$(loc^{t_1}, loc^{t_2}, \cdots)$
10: $\tau_{\text{normalized}} =$ CA(τ_{ids})
11: $\mathcal{G} \leftarrow$ mapping $\tau_{\text{normalized}}$ to $\mathcal{G}_{ori}$
12: $\boldsymbol{X} =$ concat$(Fea_{\text{sensor(v)}}, Fea_{\text{sensor(l)}}, Fea_{\text{static}})$
13: $\mathbf{y} =$ MKGNN$(\boldsymbol{X}, \mathcal{G})$
14: Connect nodes to restore the trajectory: $\tau_{\text{rec}} \leftarrow \mathbf{y}$

Algorithm 2: MKGNN.

Input: Trajectory graph $\mathcal{G}$ and its related features $\boldsymbol{X}$, the graph filter $\mathbf{g}_{s,\theta}$, the weight parameters $\boldsymbol{W}$;

Output: Nodes through which the trajectory passes: $\mathbf{y}$.

1: **for** the layer $l \in [L]$ **do**
2: $\boldsymbol{A}', \boldsymbol{D} \leftarrow \mathcal{G}$
3: $\hat{\mathcal{L}} = \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}'\boldsymbol{D}^{-\frac{1}{2}}$
4: Define the multi-kernel graph Fourier Transform operator: $\psi_s(\hat{\mathcal{L}})$
5: Define the graph convolution: $\mathbf{x} *_G \mathbf{g}_\theta = \frac{1}{S}\sum_{i=1}^{S} \psi_{s_i}(\hat{\mathcal{L}})\mathbf{g}_{s_i,\theta}\psi_{s_i}^{-1}(\hat{\mathcal{L}})\mathbf{x}^l$
6: Approximate the graph covolution: $P_p(g)(\hat{\mathcal{L}})$
7: Output of the l-th layer: $\mathbf{X}_{:,j}^{l+1} = \sigma(\sum_{i=1}^{d_l} w_{i,j}\frac{1}{S}\sum_{s=1}^{S} P_p(\psi_s)(\hat{\mathcal{L}})\mathbf{g}^l_{s,\theta_{(j,i)}} P_p(\psi_s^{-1})(\hat{\mathcal{L}})\mathbf{X}^l_{:,i})$
8: **end for**
9: $\mathbf{y} =$ softmax(Linear$(\boldsymbol{X}^L)$)

CNN network to map the three-dimensional data into a two-dimensional representation. Concurrently, the Yolov5 structure identifies and extracts target features, as outlined in lines 4–8. The DeepSORT algorithm introduced in line 9 utilizes the determined video (or LiDAR) position coordinates for recalibration to create separate time-series video (or LiDAR) track segments for vehicles with the same vehicle ID. Subsequently, the CA model in line 10 normalizes the trajectory data to conform to the road network graph. Following this, the graph feature matrix is computed as shown in line 11. Additionally, line 12 demonstrates how the MKGNN module is employed to identify nodes that intersect with the trajectory. Finally, line 13 connects those nodes to restore the missing trajectory.

The MKGNN module, further described in Algorithm 2, processes the trajectory graph and its associated features to output a list of nodes traversed by the trajectory. Initially, line 2 involves extracting the adjacency and degree matrices from the trajectory graph. Then, the normalized graph Laplacian matrix is computed using these matrices, as depicted in line 3. Lines 4–5 give the definition of the multi-kernel graph Fourier Transformer and the graph convolution and line 6 deploys Chebyshev polynomial to approximate the graph convolution defined before. Then, the output of l-th MKGNN is calculated by line 7. Finally, the nodes traversed by the trajectory are extracted by line 9.

IV. Experiments

A. Experimental Settings

1) Dataset: We use one simulation dataset generated by Carla with version 0.9.8 and one real-world dataset located in Hangzhou, China.

In the simulated environment, there are about 120 vehicles running in a 10 km^2 road network, with 100 networked equipped with 4 in-vehicle cameras with different orientations, a LiDAR, a GPS detector, and a gyroscope. The networked vehicles collect traffic data about their location and surrounding environment every 100 ms. In addition, there are static traffic volume and speed detectors installed on both roadsides every 200 meters and intersections, which can generate traffic volume and average speed on different road segments. Fig. 1(a) is an example of this simulation environment, and in this environment, our goal is to restore the trajectories of the remaining 20 vehicles without any detectors accurately. In the real-world dataset, we collected data from all passing vehicles at a four-way, eight-lane intersection over a one-day range. It contains the GPS data of each networked vehicle and the second-level traffic data (video, speed, acceleration, volume, point cloud) monitored by static roadside detectors, and our target is to recover all obscured vehicle trajectories efficiently.

2) Baselines: We compare DTRF with baselines from both conventional and multi-modal trajectory prediction and recovery methods, including M-LSTM [31], BiTraP [32], DGAN [33], MBT [34], TrajBERT [35], TransFuser [36], TrajGAT [14]:

- **M-LSTM**: It presents a novel LSTM model for predicting the vehicle trajectory. The model assigns confidence values to maneuvers being performed by vehicles and outputs a multi-modal distribution over future motion based on them.
- **BiTraP**: It uses a conditional variational autoencoder (CVAE) with recurrent neural networks (RNNs) to encode observed trajectories and decode multi-modal future trajectories, and it uses a bi-directional decoder to improve the accuracy of long-term predictions.
- **DGAN**: It introduces a unique dynamic graph attention network that models the dynamic social interactions among agents and conforms to traffic rules with a semantic map.

- **MBT**: It uses an LSTM-based architecture that incorporates persons' preferences and predicts multiple trajectories and their probabilities. Besides, it uses a multi-modal loss function that updates the best trajectories.
- **TrajBERT**: It uses a BERT-based model that learns the mobility patterns of users from both forward and backward directions and refines the trajectory predictions with spatial and temporal constraints. In addition, it introduces a new loss function that incorporates spatial-temporal awareness and multi-task learning.
- **TransFuser**: It is a multi-modal fusion transformer model that learns to attend to the relevant features from different modalities and outputs a driving policy.
- **TrajGAT**: It introduces a map-embedded graph attention network (TrajGAT) that extracts and fuses temporal features following an encoder-decoder architecture and models dynamic spatial patterns using a sparse heterogeneous graph constructed from vectorized lane-level map and a rule-based graph attention network, which aims to impute missing trajectory data for vehicles that are lost from the view of roadside sensors due to dynamic occlusions.

3) Evaluation Metrics: In this paper, the average displacement error and the accuracy of the trajectory recovery are used to evaluate the performance of a model. The definitions of these two metrics are demonstrated as follows:

- *Average displacement error (ADE):* It is the average Euclidean distance difference between each predicted location and each ground truth location.
- *The accuracy of the trajectory recovery (Acc):* It is the ratio of the area where the vehicle's predicted trajectory overlaps with the area where the actual trajectory passes.

4) Implementation Details: DTRF can be divided into three components: a data processing model, which processes the multi-modal data to a structured one for trajectory recovery; a CA-based trajectory representation model, which maps trajectories into lane-level graphs and extracts microscopic traffic flow features of vehicles; and a multi-kernel graph neural network (MKGNN), which restores discrete and incomplete trajectory points into continuous trajectory paths. The data processing model deploys YOLOv5,[1] combined with the DeepSORT algorithm[2] [37]. The data processing model deploys YOLOv5, combined with the DeepSORT algorithm. First, we used YOLOv5 to identify every vehicle in each frame of the video, including its license plate, color, and vehicle type. At the same time, the same neural network structure is used to calculate the exact geographic coordinates of each unobstructed object in the radar data. Then, we associate the precise coordinates of the objects in the radar data with the vehicle information using the stop line of the intersection as a reference and use the DEEPSORT algorithm to associate the trajectories of the same vehicle between the front and back frames and generate multiple trajectory segments. Noting that the trajectory segments generated by the preprocessing method alone can only guarantee the continuity of the trajectories within the segments, the next two modules in our proposed framework then present these trajectory segments properly and merge the trajectory segments that will have the same driving behavior to generate vehicle trajectories with high confidence. In the trajectory representation model, we set $\gamma = 0.85$, the number of nodes $N = 10273$ that is the number of grids in the real-world road network. Besides, the number of node features is 7, i.e., timestamp, hourly traffic flow, dwell time, and traffic speed & acceleration (both x and y-axis). In MKGNN, we set the number of layers $K = 3$, the kernel size $S = 10$.

During the training process, the whole procedure is similar to the test process illustrated in Algorithm 1, where the graph filter $\mathbf{g}_{s,\theta}$ and the weight parameters $\boldsymbol{W}$ in the MKGNN module are learnable and can be updated by the gradient descent algorithm. Here, the adaptive Moment Estimation with Weight Decay (AdamW) [38] is utilized for gradient descent to update the learnable graph filter $\mathbf{g}_{s,\theta}$.

[1][Online]. Available: https://github.com/ultralytics/yolov5

[2][Online]. Available: https://github.com/nwojke/deep_sort

TABLE I
EXPERIMENTAL RESULTS OF DIFFERENT BASELINES

	Simulation		Real-world Data	
	ADE (m)	Acc (%)	ADE (m)	Acc (%)
M-LSTM	2.1584	15.3368	2.7102	9.3560
BiTraP	1.4059	32.2665	1.8529	18.8309
DGAN	1.4506	28.6776	1.7198	18.2018
MBT	1.6277	22.749	1.9646	17.4211
TransFuser	1.2365	42.3618	1.5734	21.2838
TrajBERT	1.3821	33.4858	1.8040	19.8117
TrajGAT	0.9454	59.1139	1.2115	43.1343
DTRF	**0.6943***	**77.7375***	**0.8295***	**69.2823***

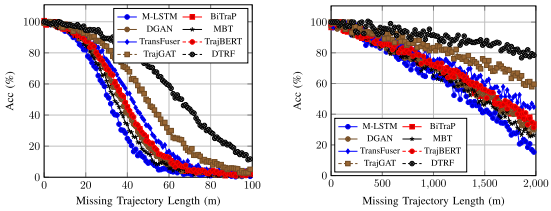
"*" indicates the statistically significant improvements (i.e., two-sided t-test with $p < 0.05$) over the best baseline. For all metrics: the lower, the better.

B. Overall Performance

The performance of all the baselines in two datasets is shown in Table I, in terms of the two metrics, i.e., *ADE* and *Acc*. The performance of each method is measured by the average of the last 10,000 runs in a total of 60,000 runs. We can see that the proposed method has better performance compared to all baselines.

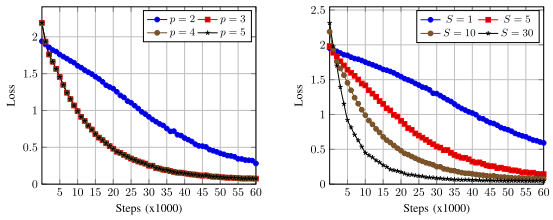
Among these baselines, M-LSTM, BiTraP, MBT, TrajBERT, and TransFuser are grounded in sequence modeling techniques, which are particularly suited for predicting vehicle trajectories that can be represented as time-ordered sequences of GPS coordinates. These models excel at handling temporal dynamics and can effectively utilize various forms of traffic data, including speeds, volumes, and signal timings, to predict future locations accurately. However, these methods struggle to effectively characterize vehicle trajectories that exhibit varied driving behaviors, leading to performances that is inferior compared to our methods.

DGAN and TrajGAT leverage graph neural networks to exploit the spatial structure of road networks. These models are particularly adept at ensuring that the predicted trajectories adhere to the physical road layouts, an essential factor for realistic trajectory planning in urban environments. By incorporating



(a) Intersection (b) Road segment

Fig. 5. The experimental results under different missing lengths of trajectories. (a) Intersection. (b) Road segment.



(a) Training Loss for different p (b) Train loss for different S

Fig. 6. Experimental results under different parameters. (a) Training Loss for different p. (b) Train loss for different S.

road network graphs, these models can prevent the prediction errors that occur when generated trajectories do not correspond to actual roadways, thus enhancing both the feasibility and reliability of the recovery results. However, these methods fall short of capturing feature differences among trajectory segments from different vehicles, leading to inaccuracies such as the incorrect merging of trajectory segments from different vehicles within the same time period. Consequently, the experimental results of these methods are still marginally inferior to our proposed approach.

In summary, our method can thoroughly analyze the multimodal data and capture the refined driving behaviors of different vehicle trajectories, which leads to a high precision of trajectory recovery.

In addition, compared to the simulation environment, the data collected from the real-world environment has more complex features, which are more difficult to capture by neural networks. Therefore, all models perform worse in this dataset. However, in the proposed DTRF, the key module, MKGNN, treats each trajectory as a continuous graph signal and defines a filter to filter out noise information to improve the precision and accuracy of trajectory recovery. Thus, our model still performs the best.

C. Ablation Study

Impact of Missing Trajectory Length

Noting that the incompleteness of a trajectory can have a significant impact on the performance of trajectory recovery, we divide the training data according to the length of the missing trajectory. Moreover, the traffic environment at the intersection is different compared to the one in the road segment. Therefore, we selected vehicle trajectories located at intersections with a width of about 100 meters and road sections with a length greater than 2 km for experiments, as shown in Fig. 5. We can see that the traffic conditions at intersections are more complex than those at road sections: when the path missing rate is higher than 60%, the recovery accuracy of each method drops significantly. Besides, our model outperforms the other baselines in all settings, which demonstrates the robustness of the proposed framework.

Parameter Analysis

Refer to (24), we have two parameters that need to be defined manually: p and S. In order to explore the influence of these two parameters on the experimental results, we use the method of controlling variables to conduct experiments on each parameter in turn.

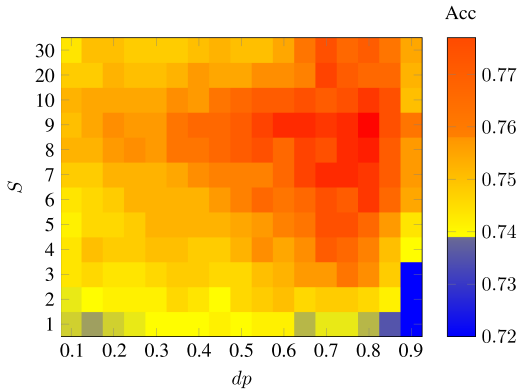
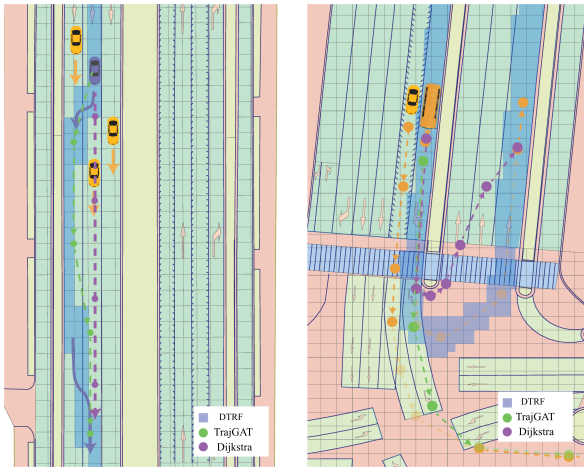


Fig. 7. Results under different settings of S and dp.

First, we verify the effect of p on the metric Acc of the proposed framework. It can be seen from Theorem 1 that as p increases, the approximation error using Chebyshev polynomials decreases exponentially, that is, a small value of p can still achieve better performance. Taking the Simulation dataset as an example, we take $p = 2, 3, 4$, and 5 to conduct experiments, and the experimental results are shown in Fig. 6(a). We can find that when $p >= 3$, the model loss are not affected by the Chebyshev polynomial order p, that is, we can think that the approximation method can reach the optimum when $p = 3$. This also verifies the proposed theorem 1.

We then conduct experiments for different values of S. We select $S = 1, 5, 10$ and 30 in turn, and the experimental results are shown in Fig. 6(b). It can be seen intuitively that when S is larger, the model converges faster, and the loss value is also lower after the model converges. One possible reason is that different kernels can capture different driving characteristics, which enriches the ability of model expression.

When we observe the relationship between Acc and the value of S, the Acc generally shows a trend of improvement as S increases until a certain level. But when S continues to increase, the Acc of the model begins to decline. This is mainly because the redundancy in the model is too high, which increases the difficulty of model training. In order to solve this problem, a simple idea is to increase the dropout rate of MKGNN. In order to further explore the relationship between the hyperparameters dp and S, a series of more detailed experiments are conducted under different settings, and the experimental results are shown in Fig. 7. The redder area in the figure indicates higher Acc, and the yellow and blue areas indicate lower test Acc. From a



(a) Road segment (b) Intersection

Fig. 8. Visualization of trajectory recovery results using different algorithms. (a) Road segment. (b) Intersection.

horizontal point of view, taking $S = 9$ as an example, when the dropout rate gradually increases from 0.3 to 0.8, the accuracy of the model is also gradually improved, but a too high model dropout rate ($dp > 0.8$) results in a decrease in accuracy due to too much sample information is manually masked during the training process. From a vertical perspective, when S gradually increases, there is a similar phenomenon. This is because the redundancy in the model needs to be maintained in a reasonable range, which is also consistent with our experience.

D. Case Study

In the case study, we select one road segment and an intersection in the real-world road network and conduct experiments. We select vehicles randomly at the peak hour and draw the possible trajectory generated by the Dijkstra algorithm (purple line), the TrajGAT model (green line), and DTRF (blue area).

Trajectory Recovery Within The Road Segment

In Fig. 8(a), three networked yellow vehicles equipped with detection devices only monitor the positions of a blue vehicle before and after it overtakes, and our goal is to recover the trajectory of this blue vehicle completely. The traditional Dijkstra algorithm, while treating each vehicle as an individual point, overlooks the physical space occupied by each vehicle within its lane. This oversight renders it ineffective at modeling the crucial lane-changing behavior needed to avoid collisions with the vehicle in front of the blue vehicle during the trajectory recovery process. On the other hand, the graph-based algorithm TrajGAT, along with our proposed method, successfully captures this behavior, demonstrating a more effective performance in this setting.

Trajectory Recovery Within An Intersection

Conversely, the TrajGAT algorithm struggles to accurately model the driving behavior characteristics specific to different trajectory segments, leading to the minivan's trajectory at the intersection being erroneously merged with that of the bus, resulting in significant recovery inaccuracies. Our proposed method, DTRF, effectively addresses these two major challenges simultaneously, thereby accurately reconstructing the trajectory of buses.

Trajectory recovery at intersections becomes more complex, as illustrated in Fig. 8(b). There are two cars in this figure, and the trajectories in orange of each vehicle are their real trajectories, whereas the light-colored part is the real trajectory segments that are obscured and need to be recovered. For clarity, we only show the trajectories of the buses after trajectory recovery by various algorithms. Dijkstra calculates the shortest path between two ordered trajectory segments but still ignores the space occupied by the vehicles. Practically, the bus cannot navigate the path indicated by the purple line using just a minor steering adjustment to execute a U-turn, as this defies the laws of physics related to time and motion. Conversely, the TrajGAT algorithm struggles to accurately model the driving behavior characteristics specific to different trajectory segments, leading to the trajectory of the car at the intersection being erroneously merged with that of the bus, resulting in significant recovery inaccuracies. Our proposed method, DTRF, effectively addresses both two major challenges simultaneously, thereby accurately reconstructing the trajectory of buses.

V. Conclusion

In this paper, we introduce a novel framework, DTRF, designed to tackle the complex challenges of multi-modal trajectory recovery. This framework utilizes Contextual Analysis (CA) to uniformly represent diverse vehicle trajectories and integrates a Multi-Kernel Graph Neural Network (MKGNN) to accurately reconstruct missing trajectory segments. The CA effectively standardizes and interprets multi-modal traffic data, which vary in structure, sampling rates, and meanings, thereby enhancing the framework's ability to leverage these diverse data types. The MKGNN, grounded in spectral graph theory, employs distinct kernels to model various driving behaviors and efficiently extracts continuous vehicle trajectories.

Looking ahead, we aim to enhance the generalizability of DTRF across a broader range of real-world scenarios involving larger datasets. This includes plans to explore the integration of larger, more complex models such as LLMs to further improve the accuracy and reliability of trajectory recovery. Additionally, we plan to explore the development of adaptive algorithms that can dynamically adjust to varying traffic conditions and vehicle behaviors. This initiative aims to significantly enhance the robustness and flexibility of our framework, particularly in diverse environments where there is a need to simultaneously recover a large number of trajectories from the same period and location. This capability is crucial for ensuring accurate and efficient trajectory recovery in complex traffic scenarios.

References

[1] L.-M. Ang, K. P. Seng, G. K. Ijemaru, and A. M. Zungeru, "Deployment of IoV for smart cities: Applications, architecture, and challenges," *IEEE Access*, vol. 7, pp. 6473–6492, 2019.

[2] J. Sun, G. Xu, T. Zhang, M. Alazab, and R. H. Deng, "A practical fog-based privacy-preserving online car-hailing service system," *IEEE Trans. Inf. Forensics Secur.*, vol. 17, pp. 2862–2877, 2022.

[3] Y. Wu, H.-N. Dai, H. Wang, Z. Xiong, and S. Guo, "A survey of intelligent network slicing management for industrial IoT: Integrated approaches for smart transportation, smart energy, and smart factory," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 2, pp. 1175–1211, Secondquarter 2022.
[4] X. Xu et al., "Game theory for distributed IoV task offloading with fuzzy neural network in edge computing," *IEEE Trans. Fuzzy Syst.*, vol. 30, no. 11, pp. 4593–4604, Nov. 2022.
[5] J. Sun et al., "A tamper-resistant broadcasting scheme for secure communication in Internet of Autonomous Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 3, pp. 2837–2846, Mar. 2024.
[6] J. Xing, W. Wu, Q. Cheng, and R. Liu, "Traffic state estimation of urban road networks by multi-source data fusion: Review and new insights," *Physica A: Stat. Mechanics Appl.*, vol. 595, 2022, Art. no. 127079.
[7] H. Shao, L. Wang, R. Chen, H. Li, and Y. Liu, "Safety-enhanced autonomous driving using interpretable sensor fusion transformer," in *Proc. Conf. Robot Learn.*, 2023, pp. 726–737.
[8] X. Kong, W. Zhou, G. Shen, W. Zhang, N. Liu, and Y. Yang, "Dynamic graph convolutional recurrent imputation network for spatiotemporal traffic missing data," *Knowl.-Based Syst.*, vol. 261, 2023, Art. no. 110188.
[9] L. Liao, Y. Lin, W. Li, F. Zou, and L. Luo, "Traj2Traj: A road network constrained spatiotemporal interpolation model for traffic trajectory restoration," *Trans. GIS*, vol. 27, no. 4, pp. 1021–1042, 2023.
[10] L. Liu, J. Feng, X. Mu, Q. Pei, D. Lan, and M. Xiao, "Asynchronous deep reinforcement learning for collaborative task computing and on-demand resource allocation in vehicular edge computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 15513–15526, Dec. 2023.
[11] R. W. Liu, M. Liang, J. Nie, W. Y. B. Lim, Y. Zhang, and M. Guizani, "Deep learning-powered vessel trajectory prediction for improving smart traffic services in maritime Internet of Things," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 5, pp. 3080–3094, Sep./Oct. 2022.
[12] X. Kong, Q. Chen, M. Hou, A. Rahim, K. Ma, and F. Xia, "RMGen: A tri-layer vehicular trajectory data generation model exploring urban region division and mobility pattern," *IEEE Trans. Veh. Technol.*, vol. 71, no. 9, pp. 9225–9238, Sep. 2022.
[13] Y. Guo, R. W. Liu, J. Qu, Y. Lu, F. Zhu, and Y. Lv, "Asynchronous trajectory matching-based multimodal maritime data fusion for vessel traffic surveillance in inland waterways," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 11, pp. 12779–12792, Nov. 2023.
[14] C. Zhao, A. Song, Y. Du, and B. Yang, "TrajGAT: A map-embedded graph attention network for real-time vehicle trajectory imputation of roadside perception," *Transp. Res. Part C: Emerg. Technol.*, vol. 142, 2022, Art. no. 103787. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X22002157
[15] C. Luo, L. Sun, D. Dabiri, and A. Yuille, "Probabilistic multi-modal trajectory prediction with lane attention for autonomous vehicles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 2370–2376.
[16] Z. Yin, R. Liu, Z. Xiong, and Z. Yuan, "Multimodal transformer networks for pedestrian trajectory prediction," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 1259–1265.
[17] Y. Hu, W. Zhan, L. Sun, and M. Tomizuka, "Multi-modal probabilistic prediction of interactive behavior via an interpretable model," in *Proc. IEEE Intell. Veh. Symp.*, 2019, pp. 557–563.
[18] J. Wang, N. Wu, X. Lu, W. X. Zhao, and K. Feng, "Deep trajectory recovery with fine-grained calibration using Kalman filter," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 3, pp. 921–934, Mar. 2021.
[19] T. Xia et al., "Attnmove: History enhanced trajectory recovery via attentional network," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 5, pp. 4494–4502.
[20] H. Ren et al., "MTrajRec: Map-constrained trajectory recovery via Seq2Seq multi-task learning," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 1410–1419.
[21] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 2778–2788.
[22] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *Proc. IEEE Int. Conf. Image Process.*, 2017, pp. 3645–3649.
[23] F. R. Chung, *Spectral Graph Theory*, vol. 92. Providence, R.I., USA: American Mathematical Soc., 1997.
[24] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Representations*, Toulon, France, Apr. 2017. [Online]. Available: https://openreview.net/forum?id=SJU4ayYgl
[25] S. Smale and D.-X. Zhou, "Geometry on probability spaces," *Constructive Approximation*, vol. 30, pp. 311–323, 2009.
[26] I. H. Sloan and W. E. Smith, "Product integration with the Clenshaw-Curtis points: Implementation and error estimates," *Numerische Mathematik*, vol. 34, no. 4, pp. 387–401, 1980.
[27] L. N. Trefethen, "Is Gauss quadrature better than Clenshaw–Curtis?," *SIAM Rev.*, vol. 50, no. 1, pp. 67–87, 2008.
[28] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, 2nd ed. New York, USA: Dover Publications, 2001.
[29] S. Xiang, X. Chen, and H. Wang, "Error bounds for approximation in Chebyshev points," *Numerische Mathematik*, vol. 116, no. 3, pp. 463–491, 2010.
[30] Z. Zhang, L. Shi, and D.-X. Zhou, "Classification with deep neural networks and logistic loss," *J. Mach. Learn. Res.*, vol. 25, no. 125, pp. 1–117, 2024.
[31] N. Deo and M. M. Trivedi, "Multi-modal trajectory prediction of surrounding vehicles with maneuver based LSTMs," in *Proc. IEEE Intell. Veh. Symp.*, 2018, pp. 1179–1184.
[32] Y. Yao, E. Atkins, M. Johnson-Roberson, R. Vasudevan, and X. Du, "BiTraP: Bi-directional pedestrian trajectory prediction with multi-modal goal estimation," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 1463–1470, Apr. 2021.
[33] Y. Wang, Z. Jiang, Y. Lin, and J. C. Niebles "Multimodal trajectory prediction for autonomous driving with semantic map and dynamic graph attention network," in *Proc. NeurIPS Virtual Workshop on Traffic and Mobility Challenges (NeurIPS-TMC)*, 2020. [Online]. Available: https://arxiv.org/abs/2007.06241
[34] S. Hauri, N. Djuric, V. Radosavljevic, and S. Vucetic, "Multi-modal trajectory prediction of NBA players," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2021, pp. 1640–1649.
[35] J. Si et al., "TrajBERT: BERT-based trajectory recovery with spatial-temporal refinement for implicit sparse trajectories," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 4849–4860, May 2024.
[36] A. Prakash, K. Chitta, and A. Geiger, "Multi-modal fusion transformer for end-to-end autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 7077–7087.
[37] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan, "POI: Multiple object tracking with high performance detection and appearance feature," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 36–42.
[38] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. 7th Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=Bkg6RiCqY7

Xiao Han received the B.Sc. and M.Sc. degrees from Zhejiang University of Technology, Hangzhou, China. He is currently working toward the Ph.D. degree with the City University of Hong Kong, Hong Kong. In 2017, he participated in the "City Brain" Project. His research interests include Big Data analytics and intelligent transportation systems.

Ding-Xuan Zhou received the B.Sc. and Ph.D. degrees in applied mathematics from Zhejiang University, Hangzhou, China, in 1988 and 1991, respectively. He was with the City University of Hong Kong, Hong Kong, in various roles from 1996 to 2022, including a Chair Professorship during 2009–2022. He joined the University of Sydney, Sydney, NSW, Australia, in 2022 as a Professor and the Head of School of Mathematics and Statistics. He was rated as a highly-cited researcher during 2014–2017. His research interests include theory of deep learning, neural networks, statistical learning theory, wavelet analysis, and approximation theory. He is serving on the editorial board of more than ten international journals, and is also the Editor-in-Chief of the journals *Analysis and Application and Mathematical Foundations of Computing*.

Guojiang Shen received the B.Sc. degree in control theory and control engineering and the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 1999 and 2004, respectively. He is currently a Professor with the College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou. His research interests include artificial intelligence theory, Big Data analytics, and intelligent transportation systems.

Xiangjie Kong (Senior Member, IEEE) received the B.Sc. and Ph.D. degrees from Zhejiang University, Hangzhou, China. He was an Associate Professor with the School of Software, Dalian University of Technology, Dalian, China. He is currently a Full Professor with College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou. He has authored or coauthored more than 180 scientific papers in international journals and conferences (with over 150 indexed by ISI SCIE). His research interests include network science, mobile computing, and urban computing.

Yulong Zhao received the B.Sc. degree in mathematics from Beijing Normal University, Beijing, China, in 2007, and the MPhil. and Ph.D. degrees in applied mathematics from City University of Hong Kong, Hong Kong, in 2009 and 2013, respectively. He is currently a Senior Lead Engineer with Hong Kong Applied Science and Technology Research Institute, Hong Kong. His research interests include learning theory, computer vision, natural language processing, biomedical analysis, quantitative investment, and intelligent systems.