# UrbanMotion: Visual Analysis of Metropolitan-Scale Sparse Trajectories

Lei Shi, Congcong Huang, Meijun Liu, Jia Yan, Tao Jiang, *Student Member, IEEE*, Zhihao Tan, Yifan Hu, Wei Chen, and Xiatian Zhang

**Abstract**—Visualizing massive scale human movement in cities plays an important role in solving many of the problems that modern cities face (e.g., traffic optimization, business site configuration). In this article, we study a big mobile location dataset that covers millions of city residents, but is temporally sparse on the trajectory of individual user. Mapping sparse trajectories to illustrate population movement poses several challenges from both analysis and visualization perspectives. In the literature, there are a few techniques designed for sparse trajectory visualization; yet they do not consider trajectories collected from mobile apps that possess long-tailed sparsity with record intervals as long as hours. This article introduces UrbanMotion, a visual analytics system that extends the original wind map design by supporting map-matched local movements, multi-directional population flows, and population distributions. Effective methods are proposed to extract and aggregate population movements from dense parts of the trajectories leveraging their long-tailed sparsity. Both characteristic and anomalous patterns are discovered and visualized. We conducted three case studies, one comparative experiment, and collected expert feedback in the application domains of commuting analysis, event detection, and business site configuration. The study result demonstrates the significance and effectiveness of our system in helping to complete key analytics tasks for urban users.

**Index Terms**—Movement visualization, sparse trajectory, wind map

## 1 Introduction

Mapping the movement of people has long been a research topic in GIS and visualization communities [1], [2], [3], [4]. In modern cities, human movements are measured via advanced sensing technology, e.g., GPS [5], road-side sensors [6], and participatory crowdsourcing [7]. Visualizations of massive scale movement data, as an important method for urban analytics, pave the way for the success of many real-world applications [8], e.g., traffic optimization [9], urban planning [10], [11], and business site configuration [11].

The human movement data in cities is often sparse in both space and time. Spatially, each type of measurements or sensors could capture the movement of only one group of people or under a particular circumstance. For example, road-side sensors record the movement of people in cars, predominately close to intersections [6]; tourism mobile apps mostly track the tourist's movement around places of interests [12]. Temporally, the trajectory of individual people is seldom measured in real time because of the constraint on communication cost. The intra-trajectory record intervals are often seconds or minutes [13], [14], [15].

This paper studies a movement dataset containing trajectories of millions of mobile users, which are synthesized from location records of hundreds of thousands of mobile apps (refer to Section 3.1 for details). Our dataset covers a majority of population groups and spatial locations in a city, thus avoiding the spatial sparsity issue. The dataset allows us to construct a population-level movement visualization to reveal the overall human movement pattern in cities. Nevertheless, as a trade-off of the comprehensive spatial coverage, the reporting frequency of each individual people's locations in our dataset becomes much lower, on average 0.4 records per hour, due to energy and privacy concerns of mobile apps. Therefore, our dataset is called (temporally) sparse trajectories, which are also observed in many other scenarios (e.g., geo-tagged social media data [16]).

Mapping sparse trajectories raises several questions from both analysis and visualization perspectives. On the analysis side, can we extract human movements from sparse trajectories when the average record interval ($\sim$2.5 hours) is longer than the elapsed time of a single trip in cities ($<2$ hours)? How to aggregate extracted individual movements for population-level visualization? Existing literature have proposed many aggregation methods [17], e.g., Origin-Destination (OD) based [18], [19], route based [20], [21], and spatiotemporal (ST) aggregations [22], [23]. These methods either demand the trajectory information and characteristics that are not available in sparse data (OD and route), or are not designed to illustrate global movement patterns (ST


- *L. Shi and M. Liu are with SKLSDE and Beijing Advanced Innovation Center for Big Data and Brain Computing, School of Computer Science and Engineering, Beihang University, Beijing 100191, China. E-mail: leishi@buaa.edu.cn, lmj199804@gmail.com.*
- *C. Huang, J. Yan, T. Jiang, and Z. Tan are with SKLCS, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China. E-mail: {huangcc, jiangt, tanzh}@ios.ac.cn, yanjia@iscas.ac.cn.*
- *Y. Hu is with Yahoo Labs, Sunnyvale, CA 94089 USA. E-mail: yifanhu@yahoo.com.*
- *W. Chen is with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310027, China. E-mail: chenwei@cad.zju.edu.cn.*
- *X. Zhang is with Beijing Tendcloud Tianxia Technology Co., Ltd, Beijing 100027, China. E-mail: xiatian.zhang@tendcloud.com.*


*Manuscript received 20 Aug. 2019; revised 12 Apr. 2020; accepted 19 Apr. 2020. Date of publication 4 May 2020; date of current version 2 Sept. 2021. (Corresponding author: Xiatian Zhang.) Recommended for acceptance by Natalia Andrienko. Digital Object Identifier no. 10.1109/TVCG.2020.2992200*

aggregation). On the visualization side, a number of designs have been proposed to represent ODs [19], [24], [25], [26], trajectories [6], [27], and routes [20]. Most of them can not reveal global and local movements simultaneously, as well as their multivariate patterns (direction, volume, speed, etc.). Meanwhile, few existing methods are designed to visualize huge amount of temporally sparse trajectories in a city.

This work discovers a long-tailed sparsity pattern in the trajectories collected from mobile apps. This pattern allows us to effectively extract dense trips from trajectory data and aggregate them into stochastic vector field on the spatiotemporal grids of the city. On visualization, we apply the wind map design [28] because of the analogy between the movement of wind and the movement of people: 1) they all flow continuously in space and time; 2) both of them can be abstracted as vector fields from the underlying data. In summary, we make following contributions in this work.

- We propose a framework to compute and analyze the vector field of urban population movements from dense parts of sparse trajectories. (Section 4).
- We improve the wind map visualization design in recognition of the differences between wind and population in their movements. (Section 5).
- Qualitative evaluation, case studies and expert interviews are conducted on a system implementation called UrbanMotion in the mobile trajectory dataset of Chinese cities. (Section 6).

## 2 Related Work

### 2.1 Aggregation of Movement Data for Visualization

Visualization of movement data has been classified into two types of views according to the exploratory analysis task they support [17]: the situation-oriented view and the trajectory-oriented view. As aggregation methods are generally introduced to deal with huge amount of movement data for effective visualization, they must be tailored to the specific type of view and exploratory task their visualizations target to serve. Hence, separate aggregation methods are applied in situation-oriented and trajectory-oriented views respectively, though some methods work for both views.

In situation-oriented views, spatial, temporal, and spatiotemporal aggregation methods [20], [22], [23], [29] are often employed. These methods treat each position of an entity as an independent discrete event and aggregate all these positions according to time intervals, spatial compartments (e.g., cells), or both. For example, Dykes and Mountain developed several basic ST aggregation methods for movement data and introduced corresponding visual representations [29]. Andrienko *et al.* improved these methods by aggregating movement trips according to their directions and distances [30]. Clustering is used to form time intervals according to the similarity of flow situations. Willems *et al.* developed a kernel density estimation method [31] to aggregate vessel movements based on their attributes (e.g., speed and acceleration) in addition to space and time information. Scheepens *et al.* further proposed to combine several density fields into a density map to visually explore multiple movement attributes [32], [33].

ST aggregation methods, though effective in abstracting spatial and temporal dimensions of movement data, are not sufficient in our analysis scenario. Apart from serving the overview task for population distribution, our techniques are mainly designed to illustrate global population movements, which are not considered or processed locally in ST aggregations.

In trajectory-oriented views, movements of individual entities are considered. In addition to ST aggregations, aggregation by origin-destinations [18], [19] and detailed trajectory routes [20], [21] are also introduced. Guo *et al.* [19] conducted an OD aggregation to compose a matrix representation for migration of US companies. Wood *et al.* constructed OD maps over aggregations to study bicycle hire scheme and commuting behavior in cities [18], [25]. MobilityGraphs applied graph visualizations over aggregated OD flows to reveal the movement pattern of massive scale population [34]. On route aggregations, Rinzivillo *et al.* [21] implemented four kinds of distance functions between trajectories. They proposed a progressive clustering method which applies each distance function in tandem to produce easily interpretable trajectory clusters. Andrienko *et al.* [20] generalized spatial locations into a few areas by Voronoi tessellation. Routes of massive movements are then aggregated into flows among these areas.

Compared with ST aggregations, it is also hard to apply OD or route aggregations in our scenario. Because of the sparsity of our trajectory data, either accurate origins/destinations or detailed routes are not available in trajectory data for aggregation. In this work, we proposed a hybrid aggregation approach over existing methods to serve two major visual analysis tasks. First, to reveal the overview of movement population, a S×T×D (direction) aggregation method similar to the work by Andrienko *et al.* [17], [30] is introduced. To deal with huge data volume, our method applies ordinary griddings on space and time instead of elaborate abstractions. The focus is on movement directions where clustering or map-matching based aggregations are employed. Second, to discover global population flows, a flow tracing algorithm over locally aggregated movements is proposed.

### 2.2 Visual Representation of Movement

The movement of objects (e.g., people, vehicle) is generally visualized as trajectories on a geospatial map [1], [3]. When a huge amount of diversified trajectories are displayed together, the visualization quickly becomes cluttered. This is the main problem faced by most trajectory visualization techniques.

The first kind of techniques solve the problem by visualizing the output of aggregation methods in Section 2.1. Based on aggregation methods used, these visualizations can be classified into density-based, OD-based, and trajectory-cluster-based techniques. Andrienko *et al.* applied a density-based aggregation to display thematic city patterns in space and time from Twitter data [35]. Guo *et al.* proposed VIS-STAMP which applies a map of matrix representation to display OD interaction data in migrations [19]. Flowstrates allowed users to visually query OD flows by regions of interests and analyze their temporal changes through the heatmap-based flow ordering, filtering, and aggregation [24]. Wood *et al.* studied bicycle hire patterns

using OD maps and flow maps [25]. Yang *et al.* presented and evaluated MapTrix, an OD data visualization technique connecting the matrix metaphor with origin and destination maps [36]. When time information is considered, the concept of temporal OD flows can be aggregated from individual OD trips [37], [38]. Based on temporal OD flows, Ferreira *et al.* modeled a wide range of spatiotemporal queries to explore taxi trajectories in cities [26]. Kohonen map applied a Self-Organizing Map (SOM) based analysis on trajectory data, which combines automatic trajectory clustering with human interaction methods [27].

Despite the versatility of aggregation-based trajectory visualizations, our analysis scenario in this work makes it impossible to apply existing methods. For example, density-based visualizations (e.g., [35]) are good for showing the spatiotemporal distribution, but fail to reveal population movement patterns. OD-based visualizations, notably through matrix [19], [36] and heatmap [24] representations, work best for a high-level abstraction of global movements, but are not designed to illustrate and discover local movement patterns. We propose a trajectory-cluster-based visualization similar to Jankowski *et al.*'s design [37]. Compared with the original design, we introduce multivariate visual encodings to display movement direction, volume, and speed simultaneously. A wind-map like visual metaphor is proposed to augment these movement patterns.

The second kind of techniques explore the 3D design space to encode spatiotemporal movements. Kraak [39] summarized the classical space-time cube visualization designed by Hedley [40] and Kwan [41]. The main idea is to stretch a $z$-axis to represent time while keeping geospatial semantics in the x-y plane intact. A product called GeoTime [42] was developed with the same idea, which focuses on complex event analyses in the spatiotemporal context. Tominski *et al.* extended the space-time cube by introducing a display wall metaphor which stacks trajectory bands on the $z$-axis [43]. Attribute information on trajectories can be shown by the display wall. 3D visualizations, though appealing to end users, may introduce occlusions in the simultaneous display of population distribution and their movements.

Third, there are a few techniques designed for sparse trajectory visualization [6], [16], [44]. The work by Wang *et al.* [44] and Guo *et al.* [6] studied spatially sparse traffic trajectories collected at a number of transportation cells in a city. Specially, TripVista [6] is designed to explore microscopic vehicle trajectories at a given cell (e.g., road intersections), but can not be applied to visualize population movements of the whole city. In the meanwhile, Chen *et al.* proposed a visual analytics system to discover movement patterns from temporally sparse trajectories collected from geo-tagged social media [16]. Because time intervals between geo-tagged records are uniformly long (e.g., days), only inter-city movements of travelers are visualized. In summary, there is currently few existing movement visualization method that considers huge amount of temporally sparse trajectory data within a city.

Finally, the proposal by Poco *et al.* [45] on the visualization of NYC taxi trips comes closest to our work. They also considered temporally sparse trajectory data and applied vector field visualizations in their design. In comparison, the techniques in this paper have three key differences on input data and visualization goal. First, NYC data are all taxi trips and there is no need for travel detection as in our data with $>50$ percent stays. There are only start and stop locations in each trip of NYC data, an additional closest path computation is necessary to recover full travel trips from partial trajectories. Second, on trajectory aggregations, Poco *et al.* grouped taxi trips on each road segment together, similar to the map-matching approach. As discussed in Section 7, map-matching and local clustering methods optimize different visualization goals. Third, on movement visualizations, Poco *et al.* focused more on speed-related traffic patterns, e.g., slowest or fastest flows, while our work put more emphasis on directional features using a pattern-based flow seeding algorithm.

TABLE 1
The Metadata of Each Location Record

| Field | Description | Sample |
|---|---|---|
| Time | Timestamp of record | 2016-07-12 18:02:41 |
| Lon. | Longitude of location | 116.523625 |
| Lat. | Latitude of location | 39.792935 |
| Mid | Unique device ID | 1370021020431 |
| Src | Localization method | GPS |

## 3 System Overview

### 3.1 Urban Trajectory Data and Its Long-Tailed Sparsity

Our data is provided by a mobile analytics company which keeps track of billions of smart devices in China, including mobile phones, tablets, wearable devices, etc. The dataset is collected by registering third-party APIs inside more than 100,000 kinds of mobile apps. When a registered mobile app is activated on a device, an API will report the location of that device to the company server. The metadata of each location record is shown in Table 1. There are five fields in each record: timestamp, location (longitude and latitude), unique device ID (anonymized), and localization method (GPS or Wi-Fi). The location records in the dataset have a spatial resolution finer than 100 meters due to the use of GPS and Wi-Fi for localization.

To conduct this work, we extracted sample datasets at Beijing, Tianjin, and Tangshan separately (three major cities of China), from July to September, 2016 (90 days). Take the Beijing dataset as an example, it is composed of trajectories from 31.85 million devices. Each device is included if it has at least one location record within the administrative boundary of Beijing during the 90-day period. The Beijing dataset has 8.41 billion records and reaches a size of 738.1G byte. The raw trajectory of each device has been pre-processed to remove conflicting and duplicate records reported by multiple apps.

Our trajectory data has two distinctive features which make this work feasible: a high volume/coverage and a long-tailed sparsity. First, the number of devices measured exceeds a half of population in Beijing. The data is synthesized from trajectories collected from mobile apps in various domains, including entertainment, education, and information. In comparison, the taxi trajectory/pick-up data widely used in previous researches [9], [46] only represents a small share and a single type of population movement. Second, trajectories in our data are sparse over time, i.e., 0.4 records per

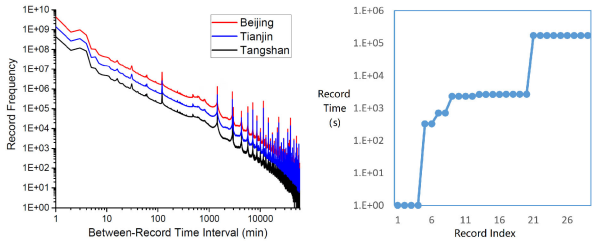



Fig. 1. The long-tailed sparsity pattern in our dataset: (a) distributions of between-record time intervals; (b) an example trajectory (X∼record index, Y∼ record timestamp).

device×hour. This may prohibit us from extracting movement information. A normal urban trip with an elapsed time shorter than two hours can be measured only once, due to the long average between-record interval in each trajectory (∼2.5 hours). In our data, instead of having uniformly sampled location records, the temporal distribution of trajectory records is highly unbalanced. We call it a long-tailed sparsity.

Fig. 1a depicts the distribution of between-record time intervals in our datasets. All three distributions follow long-tailed power-law like decays, which are indicated by the straight lines in a log-log scale. For Beijing dataset, there are 89.0 percent intervals shorter than 30 minutes, and in the meantime, 1.93 percent intervals longer than 24 hours. In other words, long-tailed sparse trajectories are mostly composed of several densely measured trips with short intra-trip intervals (e.g., minutes). An example trajectory is shown in Fig. 1b. Such a long-tailed sparsity allows us to effectively detect movements of people from their trajectory data (Section 4.1).

### 3.2 Task Characterization

The trajectory data described above allows better understanding of urban mobility due to its high volume and city coverage. The visual analytics of such urban data could suggest solutions to urban problems such as traffic optimization, urban planning, etc. To identify key user tasks associated with the analysis of our urban data, we set up pilot interviews with three domain experts in related urban sectors, i.e., urban planners (UP), public safety officers (SO), and business analysts (BA). The objective of the interview was to understand their everyday job responsibilities that are related to population movements in the city, and the current technical practice in handling their jobs. The detailed interview records are given in Appendix B, hich can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2020.2992200. From the feedback of these interviews, we summarize four user tasks that are important to the job performance of these targeted urban users. The task characterizations and key examples from experts are listed below.

- *T1. Overview of global population movements in a city: How do population movements distribute over the city in a given time period? What are the characteristics of these movement flows, including volume, speed, and direction? How do these movements interact with city infrastructure, e.g., road networks and POIs?* Urban planners study the overall movement pattern in a city to understand busy regions in daily commutes of city residents (UP). Public safety officers increase their situation awareness of the city through visualization of the overall population movement in the city (SO).
- *T2. Examination of local movements and pattern discovery: How many people (and how fast) are there moving on the roads of a local city region during a given time period? Are there local hotspots, including population movement hubs and channels?* In commute analyses, urban planners understand the detailed spatiotemporal utilization of a local road and compare the detected hubs/channels with the designed function in a city region to optimize urban planning. Business analysts help customers to make decisions on new commercial sites by understanding local population movements around the candidate and existing sites. Locations close to a hub region might be preferred (BA).
- *T3. Correlation analysis with population distributions: How do population movements correlate with the distribution of population and their statistics, e.g., the number of residents, the ratio of incoming/outgoing people, and the average moving speed?* Public safety officers identify the regions having a high number of incoming people but a low number of outgoing people, together with movement patterns of these regions, to alert a potential event (SO). Business analysts predict potential buyers of a real estate project by synthesizing movement data around the project and the metadata of these movements if available, e.g., gender, profession, purchasing power (BA).
- *T4. Detection of temporal movement dynamics and anomalies: Where and when do certain city regions have a larger volume of movement flows than their long-term average? How anomalous are these incidents? How to reason these anomalies?* Urban planners study the temporal dynamics of net incoming/outgoing movements to identify working and living regions in a city. The movement flows among these regions are further analyzed to evaluate the degree of work-home balance in certain city districts (UP). Identifying movement anomalies can help public safety officials maintain awareness of low-risk events, e.g., traffic jams; early intervene on high-risk developing events, e.g., floods, blizzards; and evaluate the impact of planned future events on the city, e.g., city marathons (SO).

### 3.3 Technical Challenge and System Pipeline

Fulfilling the above user tasks in this work faces several challenges due to the characteristics of the urban trajectory data. First, different from existing data sets with dense or uniformly sparse location records over time, our data processes a unique long-tailed temporal sparsity. How to extract and process such sparse trajectory data for movement visualization remains unknown. Second, in the movement visualization literature (Section 2.2), most existing techniques for big trajectory data apply spatiotemporal aggregations, which focus on revealing high-level movement distributions. They are not designed to illustrate global and local movement flows simultaneously (T1 & T2). Meanwhile, they do not natively support temporally sparse trajectory data. Finally, as our trajectory data measures both movements and stays of

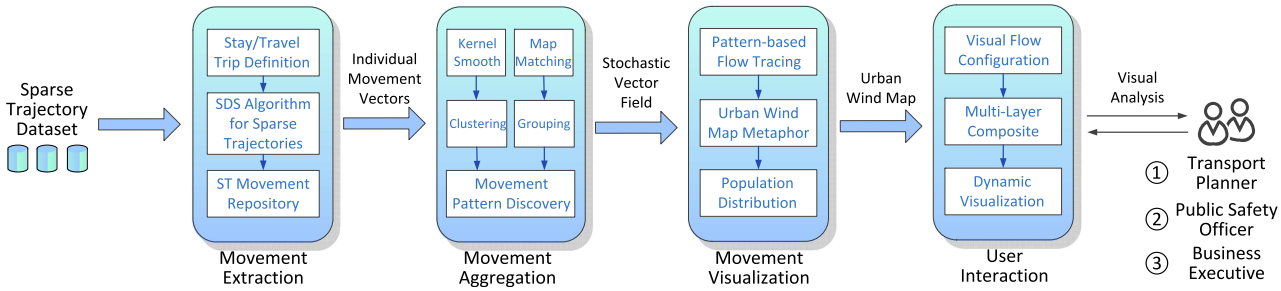



Fig. 2. The visual analytics pipeline of UrbanMotion.

urban population, customized visualizations should be designed to correlate population movements with their static, dynamic, and anomalous distributions.

In this work, we propose a suite of visual analytics techniques to tackle these challenges, which are implemented in a system called UrbanMotion. Fig. 2 illustrates the pipeline of the UrbanMotion system. It takes a set of sparse trajectories in a time period as input. In the first stage, movement trips are extracted from dense parts of each individual trajectory by an inference algorithm. A trip is further decomposed into movement vectors in spatiotemporal cells, which are stored in movement repositories. In the second stage, movement vectors are clustered to form local movement flows, after smoothing by a kernel-based method. Alternatively, a map-matching algorithm is used to adapt movement vectors onto road networks, which are further grouped into local movement flows. To this end, a stochastic movement vector field is constructed in the selected time period. Both characteristic and anomalous patterns are discovered from this vector field. In the third stage, a flow tracing algorithm is applied to the vector field to compute global movement flows using a pattern-based seed generation algorithm. Global flows are displayed by a urban wind map visualization, which is improved from the original design with adaptations to population movements. Users work with this visualization through customized interactions.

## 4 Urban Trajectory Analysis

In this section, we describe the trajectory analysis methods to support the user tasks of UrbanMotion. They include movement extraction to separate travel trips from all the mobile data (Section 4.1), movement clustering and map-matching to generate global and local movements (Sections 4.2, 4.3), and movement pattern discovery to identify temporal and anomalous movement patterns (Section 4.4).

### 4.1 Movement Extraction

We first consider a continuously measured trajectory $\Gamma$ of a mobile user during a time period $T$. $\Gamma$ is defined by the set of spatiotemporal records in $T$: $\Gamma = \bigcup_{t \in T} < t, \ell(t) >$ where $\ell(t)$ denotes the location of the user at time $t$. Any continuous sub-trajectories of $\Gamma$ are defined as the trips of $\Gamma$. For example, the trip $\gamma$ during the time period $\tau \subseteq T$ is defined as $\gamma = \bigcup_{t \in \tau} < t, \ell(t) >$.

**Definition 1.** *Stay/Travel Trip*

For any trip $\gamma$ of a trajectory $\Gamma$ during the time period $\tau$:

(a) $\gamma$ is a *stay trip* if: $|\tau| \geq \Delta T$ and $||\ell(t_1) - \ell(t_2)|| < \Delta S$ $(\forall t_1, t_2 \in \tau)$;

(b) $\gamma$ is a *travel trip* if: $\gamma$ does not overlap with any stay trip satisfying (a).

where $|\cdot|$ denotes the length of a time period, $||\cdot||$ is the $L_2$ norm that represents the spatial distance between two locations. $\Delta T$ and $\Delta S$ are the temporal and spatial thresholds in the definition.

In essence, Definition 1(a) models the stay trip as a sufficiently long time period ($\geq \Delta T$) when the trajectory is kept within a circular region of diameter $\Delta S$. This definition is consistent among all the previous literature on the stay point detection [47], [48]. Note that the stay trips of a trajectory by definition can overlap with each other in space and time. Their enclosure is called the maximal stay trip. On the other hand, based on the ground truth that a user can either stay or travel in any time point, the trip not overlapped with any stay trip is determined as a travel trip (Definition 1(b)).

The real-world human trajectory is hardly measured continuously, but consists of a list of discretely sampled records on certain time points ($t_1 < \cdots < t_L$): $\Gamma = \bigcup_{t \in \{t_1, \ldots, t_L\}} < t, \ell(t) >$, where $L$ is the number of records in trajectory $\Gamma$. It has been shown in our concurrent work [49] that if the sampling process is dense, i.e., $\forall i \in [1, L), ||t_i - t_{i+1}|| << \Delta T/2$, any travel trip extracted from the discretely sampled trajectory by Definition 1(b) is very close to the travel trip detected on the underlying continuous trajectory. The extraction of travel trips from dense trajectories can be computed by an iteration-based algorithm (see Appendix A), available in the online supplemental material.

Nevertheless, in this work we are given temporally sparse trajectories with an average record interval of 2.5 hours (compared with a default parameter of $\Delta T = 30\ min$). The extraction of travel trips on sparse trajectories is challenging. Take a uniformly sampled trajectory as an example whose record intervals are constantly longer than $\Delta T$. No travel trip can be confidently inferred. Before or after a given record, there is an unobserved time period longer than $\Delta T$. Obtaining continuously measured records in this time period could infer the given record as stay if Definition 1(a) is met.

Fortunately, we have found that the trajectories in our dataset have a long-tailed sparsity. This allows us to apply a new inference algorithm called Slice & Doubly Sliding (SDS) [49] to detect stay and travel trips from long-tailed sparse trajectories. The algorithm first slices each trajectory into multiple dense segments at all the intervals larger than $\Delta T$. Because of the long-tailed sparsity, only 2.3 percent records are in length-one segments and dropped under the default parameter of $\Delta T = 30\ min$. On each remaining dense segment, the travel trips are detected following Theorem 1 in Appendix A, available in the online supplemental

material. The pseudocode of the algorithm is also given in Appendix A, available in the online supplemental material. More details can be found in the work by the same group of authors [49].

We apply the SDS algorithm to the dataset of Beijing. The percentage of records detected as in the stay and travel trips are 50.2 and 0.83 percent respectively, with the other records undecided by the algorithm due to the sparsity of data. The spatial/temporal thresholds are set to $\Delta T = 30\ min$ and $\Delta S = 800\ m$. We studied the impact of these thresholds. It is shown in Appendix A, available in the online supplemental material, that the extraction of stay trips is less affected by the thresholds and the percentages of detected travel trips vary a lot in different thresholds. The threshold setting objective is to detect more travels while keeping Definition 1 reasonable. Finally, we pick $\Delta T = 30\ min$ because the detected ratio of travels does not increase much when switching to $\Delta T = 45\ min$ and it does not impose a strict stay definition. Similarly, $\Delta S = 800m$ is picked which maximizes the recall of travels and allows a mild stay definition compared with $\Delta S = 400m$. The thresholds of $\Delta T = 30\ min$ and $\Delta S = 800m$ are also consistent with empirical settings in recent literature [47], [48].

A potential issue in movement extraction is that only a small amount of records are kept for visualization (0.83 percent overall). This is reasonable as the trajectory data is collected from many kinds of mobile apps. Most often these apps are used in a stay mode except a few such as mobile navigation apps. In Fig. 1(a)(c) of Appendix A, available in the online supplemental material, it can be found that at most 98 percent records are detected as stays in the trajectories with certain global sparsity. This indicates the dominance of stays in our data assuming small correlation between the sampling rate of trajectories and its stay/travel ratio. To increase the percentage of travel records in use, the subset of trajectory with a global sparsity of 5∼10 minutes can be selected, where up to 2.8 percent records can be extracted as travels. Another method is to apply more advanced mobility inference algorithms (e.g., those in Ref. [49]), but comes at the cost of reduced accuracy. The SDS algorithm guarantees a 100 percent accuracy of detected travel trips. Because of the scale of our data (up to 7 million extracted travel records in Beijing), we do not adopt these improvements. We caution that in case the data size is small, the result of UrbanMotion may not faithfully represent the overall movement in a city. A longer time period should be selected to ensure there are enough travel records for movement visualization.

### 4.2 Movement Clustering

On the travel trips detected from each trajectory, we connect pairs of adjacent records to form multiple movement vectors, as shown by the blue arrow lines in Fig. 3a. From our dataset, there are billions of movement vectors extracted. It is impossible to visualize all of them on the same map. In the first method, we propose to cluster these vectors into significant local movement flows according to directional affinity and space/time context. The detailed steps of our method are illustrated in Fig. 3b, 3c, and 3d.

The first step is to sort movement vectors in space and time dimensions. We partition the land of a city into square cells of 500 meters wide and time into hourly slots. Note that different space/time granularity settings are possible. They can be specified according to user's analysis requirement. For example, if the user would like to visualize more accurate movement flows geographically, s/he can switch to a smaller grid setting. When s/he would like to track movement flows in a shorter time period, a smaller time slot than one hour could be set. For longer time periods, our system supports automatic merging of movement vectors in consecutive hours. After the spatiotemporal partitioning, all the movement vectors are categorized and stored in multiple spatiotemporal movement repositories. Each repository holds all the movement vectors starting or ending in a particular cell during a given time slot, as shown in Fig. 3b.

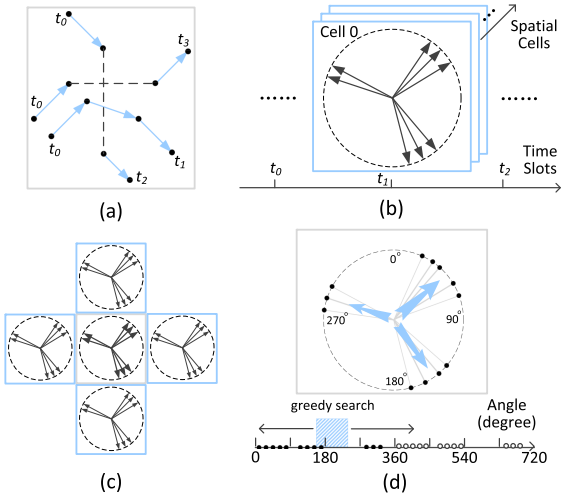



Fig. 3. The analysis of long-tailed sparse trajectories to generate local movement flows (clustering approach): (a) detecting local movement vectors from travel trips; (b) sorting vectors in the spatiotemporal movement repository; (c) applying vector KDE to smooth the spatial distribution of local movements; (d) clustering local movement vectors into flows.

In the second step, we introduce an optional smoothing process to emphasize major movement directions. The goal is to alleviate the potential error of localization records and movement vectors. Our method is a variant of the Kernel Density Estimation (KDE) over movement vectors, which is called the vector KDE. As shown in Fig. 3c, each vector in a cell is duplicated into surrounding cells, with the weight of duplicated vectors decayed by the distance to the original vector according to a Gaussian kernel function. Formally, the movement vectors in each cell can be computed by

$$V(x_i) \leftarrow \bigcup_{v_0(x_j)\in V_0(x_j), |x_i - x_j| \le R} \left\{ v_0(x_j), \frac{1}{\sqrt{2\pi}h} e^{-\frac{(x_j - x_i)^2}{2h^2}} \right\}, \tag{1}$$

where $v_0(x_j) \in V_0(x_j)$ is a vector in the initial set of movements in cell $j$ centered at $x_j$. $\{v_0(x_j), w\}$ is the weighted vector. $V(x_i)$ is the vector set in cell $i$ after applying KDE. The cell $i$ will be influenced by surrounding cells within a range of $R$. $h > 0$ is the KDE bandwidth that controls the degree of smoothing.

In a final step, the set of movement vectors in each cell-time pair are clustered to generate significant local movement flows. A baseline method is to apply the density-based spatial clustering over intersection points between vectors

and their surrounding circle (black dots in the upper part of Fig. 3d). Such a algorithm, e.g., DBSCAN [50], has a worst-case complexity of $O(n^2)$ due to the computation of distance matrix, where $n$ is the number of points. As we will execute clustering on millions of cell-time pairs, an $O(n^2)$ complexity is almost infeasible. Noticing that movement vectors in our case only have one degree of freedom, i.e., their directions, we introduce an 1-D DBSCAN algorithm, which has an $O(n)$ computational complexity.

As shown in the lower part of Fig. 3d, each movement vector is depicted as a point in the linear axis of [0°, 360°). To allow a movement cluster crossing 0° (360°), all these points are duplicated in an extended axis of [360°, 720°), as indicated by the hollow points. The 1-D DBSCAN algorithm starts with an initial window of [180°, 181°). The window expands on the axis in both directions if the point density in the current window exceeds a minimal threshold, i.e., the density condition of DBSCAN. When the window can not expand any more, all the points in the window become a cluster if the number of points exceeds the minimal threshold, i.e., the support condition of DBSCAN. The clustered points, as well as their duplications, are then removed from the axis. The next window starts from the right border of the previous window and detects remaining clusters until no cluster can be found any more. Each detected cluster corresponds to a local movement flow, denoted as $f = \{v, w\}$. The flow direction ($v \in [0, 360)$) is set as the weighted circular mean of vector directions in the cluster. The volume ($w$) is set as the sum of all the vector weights.

After aggregating local movements, a vector field is constructed in a given time period (e.g., an hour), which can be visualized to meet user task T1 in Section 3.2. Unlike classical vector fields, there can be multiple local movement flows in a cell. Thus, it is called a stochastic vector field.

### 4.3 Map-Matching

The clustering approach aggregates multiple movement directions into a single flow which may not be on the underlying roads of urban map any more. The visualization result can obscure the detailed movement pattern on road networks. In an alternative approach to clustering, we propose to apply map-matching [51], a well-studied technique in intelligent transport system (ITS) and GIS research fields, to calibrate each movement vector onto urban roads. Local movement flows are naturally grouped at each urban road bidirectionally. Note that from the result in Section 5, both clustering and map-matching methods have pros and cons, and can be favored in different application scenarios. The comparison and usage of the two methods are further discussed in Section 7.

As shown in Fig. 4a, the initial inputs of map-matching are movement vectors detected in each cell and the city's road network. In our implementation, OpenStreetMap data is used due to its openness and the popularity of the data interface, while other map data is also compatible with our framework. From the OpenStreetMap data, any type of "ways" representing a kind of road is extracted and duplicated in both directions. The geometry of each road is further partitioned into several segments that are close to straight lines in each segment, as shown by orange lines in Fig. 4b. Next, each movement vector is matched to a road segment

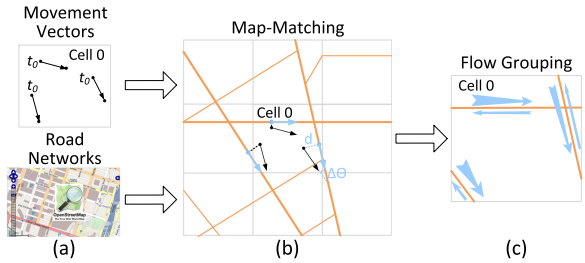



Fig. 4. Movement aggregation through the map-matching approach: (a) movement vectors and road networks as input; (b) map-matching process; (c) movement flows grouped on road networks.

by the map-matching algorithm. In the literature, many algorithms have been proposed for map-matching, e.g., geometric [52], topological [53], probabilistic [54], and model-based algorithms [55], [56], [57], [58], [59]. We applied a weighted topological algorithm proposed by Velaga *et al.* [60], which is scalable and has been successfully used for sparse data [14].

In Velaga *et al.*'s algorithm to match each movement vector, a candidate set of road segments is first computed, which consists of all the segments overlapping with spatial cells close to the movement vector. By default, 9 cells including the vector's cell and 8 adjacent cells are considered, as shown in Fig. 4b. Between the vector ($v$) and each candidate road segment ($r$), a matching score ($MS$) is calculated which is summed from two weighted components: the proximity ($\mathscr{D}$) and the heading difference ($\mathscr{H}$).

$$MS(v, r) = W_d \times \mathscr{D} + W_h \times \mathscr{H}, \quad \mathscr{D} = 1 - \frac{d}{\eta}, \quad \mathscr{H} = cos(\Delta\theta), \tag{2}$$

where $d$ denotes the distance from the starting point of the vector to its projection on the road segment, $\eta$ is the maximally allowable position error in the matching (100 meters by default according to Sanaullah *et al.* [14]), $\Delta\theta$ denotes the angle difference between the movement vector and the road segment. $W_d$ and $W_h$ denote weights for proximity/heading differences and are set to 0.5 respectively.

Among all candidate segments, the segment having the highest positive matching score with the vector is selected. The movement vector is then calibrated onto the segment, using the projected point on the segment as the new start point and the segment's direction as the new movement direction. If no positive matching score is found in all candidates, the vector is classified as an outlier and dropped in the analysis. After applying map-matching to all movement vectors in a cell, local movement flows are grouped naturally on the direction of road segments (Fig. 4c). Note that in the map-matching approach, we do not introduce KDE or clustering before/after the matching process in order to ensure that the aggregated local flows are still moving on road networks.

We note that the map-matching algorithm applied here is a simplistic one mainly for the demonstration of our visual analytics approach. In real world cases, each trajectory from the mobile dataset could contain multiple sparsely sampled records that actually locate on separate road segments. Location records can also be biased by data noise due to the limitation of measurement methods. The current point-by-point nearest road mapping algorithm [60] does not consider the

feasibility of matched paths for trajectories and is not optimized for noisy input data. In a real application, we highly recommend to apply advanced map-matching algorithms, though the computation cost could be larger. For example, the Hidden Markov Model (HMM) based algorithm by Newson and Krumm [57] leverages the context in trajectory routes and the connectivity/constraint of underlying road networks. HMM approaches are shown to be robust to data noise and sparseness according to the result of existing researches [57], [58], [59].

### 4.4 Pattern Discovery

From stochastic vector fields generated by the trajectory data, we discover both spatial and temporal patterns, which will be displayed in the visualization to meet the user tasks in T2 and T4.

For *spatial patterns*, we identify two types of spatial cells that are characteristic in the population movement: hub cells and channel cells. A hub cell indicates the place where people move in multiple directions. A channel cell indicates major roads where population move primarily in two directions. We adopt the metric of Bimodality Coefficient (BC) recently suggested by Freeman and Dale [61] to detect channel cells that have bimodal movement direction distributions. The BC metric also reveals a cell's similarity to the uniform movement direction distribution which helps to detect hub cells simultaneously. Formally, consider a spatial cell where $k$ local movement vectors are extracted, denoted as $V = \bigcup_{i=1,\dots,k}\{v_i\}$ where $v_i \in [0, 360)$ is the angular direction of the $i$th vector, the BC metric is computed by

$$BC(V) = \frac{\gamma(V)^2 + 1}{\kappa(V) - 3 + 3 \cdot \frac{(k-1)^2}{(k-2)(k-3)}}, \tag{3}$$

$\kappa(V) = \frac{1}{k}\sum_{i=1,\dots,k}(\frac{||v_i-\mu(V)||_a}{\sigma(V)})^4$, $\gamma(V) = \frac{1}{k}\sum_{i=1,\dots,k}(\frac{||v_i-\mu(V)||_a}{\sigma(V)})^3$, $\sigma(V) = \sqrt{\frac{1}{k}\sum_{i=1,\dots,k}||v_i - \mu(V)||_a^2}$ are the kurtosis, skewness, and standard deviation of the distribution of $V$. $||x||_a = (x + 180)\%360 - 180$ is the angular distance.

Note that the average of vector directions is computed by the mean of circular quantities [62]

$$\mu(V) = \left(atan2\left(\frac{\sum_{i=1,\dots,k}\sin(v_i)}{k}, \frac{\sum_{i=1,\dots,k}\cos(v_i)}{k}\right) + 360\right)\%360. \tag{4}$$

The BC metric of a standard bimodal distribution is 1.0 whereas that of a uniform distribution is $\frac{5}{9}$. We compute the BC metrics of all the cells in a given time period. The cells with their metrics closest to 1.0 are detected as the channel cells and the cells with their metrics closest to $\frac{5}{9}$ are detected as the hub cells. The number of detected cells can be configured in the visualization. Note that all these cells need to have a considerably large total flow volume.

For *temporal patterns*, we apply the anomaly detection method on the time series of the movement statistics in a given cell (e.g., the summed flow volume, the stay/travel record number). Three types of anomalies are detected for each cell-hour pair: the hourly, the daily, and the weekly anomalies. The hourly anomaly compares the statistics in the current cell-hour pair with those of the other 23 hours in the same day and cell. The daily anomaly compares the current hour with the same hour in all the other days. The weekly anomaly compares the current hour with the same hour in the same weekday/weekend (e.g., Monday). Take the hourly anomaly as an example, for each day and cell, there are 24 values in each statistic, denoted as $m = 24$, which is assumed to follow a normal distribution with an average of $\mu$ and a variance of $\sigma$. Consider the $i$th hour, whose statistic is denoted as $x_i$, its anomaly score is computed by the Extreme Value Theory (EVT) [63].

First, the normalized Mahalanobis distance between $x_i$ and the average $\mu$ is computed by

$$y_m = \frac{\frac{|x_i-\mu|}{\sigma} - \mu_m}{\sigma_m} \quad where \tag{5}$$

$$\mu_m = \sqrt{2\ln m} - \frac{\ln\ln m + \ln 2\pi}{2\sqrt{2\ln m}}, \quad \sigma_m = \frac{1}{\sqrt{2\ln m}}, \quad m = 24.$$

Second, the probability for the statistic to deviate from the normal distribution, denoted as $p$, is computed. The probability is finally translated to an anomaly score, denoted as $\alpha(x_i)$.

$$p = e^{-e^{-y_m}}, \quad \alpha(x_i) = \min\left(\frac{-\ln(1-p)}{\Phi}, 1\right), \tag{6}$$

where $\Phi$ is the expected highest anomaly score for normalization.

### 4.5 Implementation

For efficiency consideration, all trajectory analyses are preprocessed in the backend by Python. Most algorithms are natively parallel: movement extraction in the trajectory level, movement clustering, map-matching, and BC/anomaly metric computation in the cell level. These parallelisms are exploited to speed up the trajectory analysis process. In this process, trajectory data and results are stored in files which are demultiplexed by groups of trajectories or time intervals for the ease of analysis. For example, raw location records are partitioned into ~3000 chucks, with each chunk composed of 10K trajectories. In each stage of analysis, input data is directly loaded from relevant file chunks and processed in parallel, without the need of querying from database. This boosts the processing speed by eliminating the cost of DBMS. The analysis results are provisioned to the front-end through a Koa middleware framework of the Node.js server [64].

Table 2 lists the running time of analytics carried out in the back-end, which are measured in a Linux server with two 14-core 2.8 GHz Intel Xeon 5117 CPU and 128 GB of memory. By default, 20 threads are used for parallel processing. Note that the movement extraction time is for processing all the 31.85M trajectories (8.4 billion records) in the 90-day dataset of Beijing, thus the long running time (only once) should not be concerned. Vector KDE, movement clustering, map-matching, pattern discovery, and flow tracing times are for processing the dataset of a typical hour (0.35M local movement vectors) using default algorithm parameters. Besides the optional KDE and map-matching

TABLE 2
The Running Time of UrbanMotion Trajectory Analysis and Visualization Algorithms

| Analysis Stage | | Input Data | Time (s) |
|---|---|---|---|
| Trajectory Analysis | Extraction | 8.4B records (90 days) | 21201.2 |
| | Vector KDE | 0.35M local movement vectors in 29.2K cells (1 hour) | 312.6 |
| | Clustering | | 11.1 |
| | Map-Matching | | 554.5 |
| | Pattern discovery | | 1.5 |
| Visualization | Flow Tracing | 40.1K clusters in 10K cells (1 hour) | 1.6 |

process, the standard back-end analytics pipeline can be pre-computed very fast for interactive online visualization.

## 5 Visualization

### 5.1 Rationale

We reviewed related work on movement visualizations (Section 2.2). Most existing designs focus on the display of trajectories, routes, OD flows, and trajectory clusters, but can not reveal global and local movement patterns simultaneously (T1 and T2). Because of its temporally sparse nature, our data does not have full semantics in each trajectory (e.g., origins/destinations, routes). Instead, the trajectory analysis method in our system leverages the long-tailed sparsity pattern to extract local movements and then abstracts them into a stochastic vector field for visualization (Section 4).

A direct visualization of vector field fails to illustrate the continuous nature of human movements in the population scale. We consider the design of wind map [28], which is an implementation of state-of-the-art integration-based flow visualization [65]. In general, wind map is composed of two types of techniques: a visual representation of wind flows using animated streamlines, and the generation of these streamlines from the underlying wind field through flow seeding and tracing.

We apply the wind map design because of the similarity between movements of wind and movements of population in their visual representation. Both wind and population move in groups of particles and form continuous trajectories. The animated streamlines can well illustrate global population movements (the overview task of T1). Multiple visual channels available in streamlines (shape, fill color, etc.) can faithfully represent the multivariate nature of both wind and population movements (direction, volume, speed, etc., see Section 5.3 for details). Both wind and movement data are processed into vector field for visualization.

On the other hand, the generation process of wind and human movements in cities vary a lot due to their different formation mechanism, demanding new analysis methods for wind map visualization of population movements. First, people move in cities along road networks while wind can take any directions by nature. In recognizing this difference, an alternative map-matching stage is applied for the visualization of local movement patterns (Section 4.3 for the local pattern examination task of T2). Second, population movements can be multi-directional in places such as road intersections, while wind is single directional in any place. Instead of the original random flow seeding in wind map, we introduce a pattern-based flow seeding and tracing algorithm that adapts to the multi-directional nature of population movements and better reveal their temporal and anomalous patterns (Section 5.4 for task T4). Third, because the underlying population is highly related to their movements, we propose a multi-layer map design that overlays the distribution of population with the movement visualization to unveil their correlation (Section 5.5 for task T3).

### 5.2 Overall Interface Design

An overview of the UrbanMotion interface is shown in Fig. 5. The widget in the middle presents the main visualization for urban movements (Fig. 5a). It is composed of multiple layers. In the top layer, all the movement flows are visualized in an animated urban wind map (Section 5.3), which is computed by flow generation algorithm (Section 5.4). Beneath, population heatmaps can be turned on to correlate movements with the statistics of underlying population, including their movement source, destination, speed, and temporal anomalies (Section 5.5). In the background, multiple types of base maps can be used as location references, including road network map, terrain map, and satellite imaging map.

Besides the main widget, UrbanMotion designs several supplemental widgets to help user interact with the movement visualization to complete analysis tasks. As shown in Fig. 5b, two time/date selection panels on the bottom-left of the interface draw the distribution of movement data over time. The lower part shows the number of raw travel trips per day by a bar chart. The upper part details the number of trips on each hour of a particular day by a clock-shaped heat map. Users can click on the bar chart to select one day and then click on the hour wedges of the heat map to specify a time window of one or more hours on that day. The population movement during the selected time window will be visualized in the main view of UrbanMotion. Several functional time/date windows, e.g., morning (7AM~10AM) and weekend (Saturday and Sunday), can be directly selected by one-click buttons on the left part of the panels. Two play buttons are also available on the panels, which can be used to trigger animations on the main movement visualization. These animations will show the dynamics of urban movements over days (for the selected hours) or over hours of a day.

On top of the main widget, an algorithm configuration panel is designed to let user set the parameters of the flow generation algorithm (Fig. 5c). Users can adjust these parameters and get realtime feedback on the main visualization to better understand the urban movements.

On the right of the main widget, there is a control panel to set up movement visualizations, including the urban wind map and the population heat map (Fig. 5d). On top of the control panel, a clock-shaped movement legend is introduced. The legend is composed of 24 sectors, each corresponding to an angular range of 15 degrees. The color lightness of each sector indicates the total volume of local movement flows within the corresponding angular range. Users can select one or more sectors on the legend to highlight the global movement flows starting from the corresponding angular ranges. The unselected flows will fade out in the background. This interaction helps users to interactively demultiplex population movements in the overview

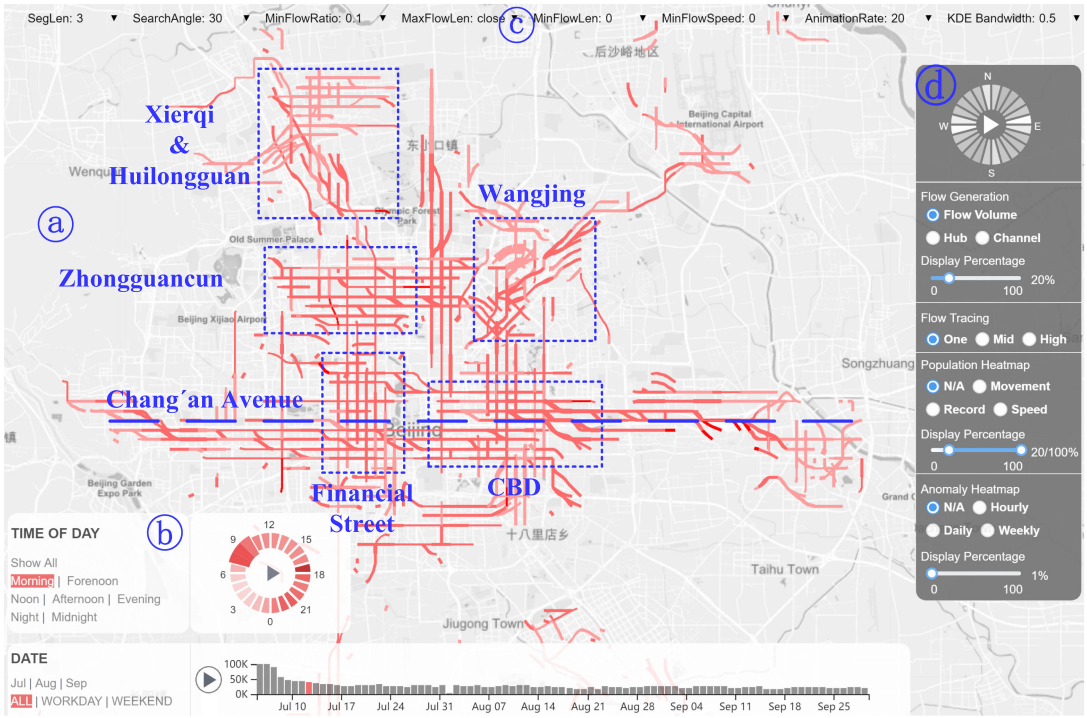



Fig. 5. UrbanMotion user interface: (a) movement visualization showing five major regions in Beijing during commute time; (b) time/date selection panel, the morning (7AM-10AM) of July 12th is selected; (c) algorithm configuration menu for flow generation; (d) visualization control panel.

visualization. At the center of the movement legend, there is a play/pause button, which can be clicked to stop the animation of movement flows in the main view for analysis and then restore their animation. The volume and speed of movement flows is better perceived in the pause-mode visualization. Beneath the movement legend, there are several sections to control the flow visualization, the population heatmap, and the anomaly heatmap. These sections are generally used to switch between different visualization patterns, e.g.,hub/channel/large-volume flows, source/destination/speed heatmaps, and hourly/daily/weekly anomalies. More details of these panels are introduced in the following subsections together with the visualization to be controlled.

The UrbanMotion interface in the frontend is implemented using Vue.js framework [66] for rendering, view composition, and interactions. Leaflet JavaScript package [67] is used for mapping functions and the display of map layers. Visualization algorithms (flow seeding and tracing) are implemented by Python in the back-end for efficiency consideration. Data requests for flow visualization are generated in the user interface, computed on the fly in the back-end, and feed to the front-end for interactive visualization.

### 5.3 Urban Wind Map

Fig. 5a gives an example of urban wind map, the key layer for movement visualization in UrbanMotion. The map is composed of multiple global movement flows. Each flow is drawn by a cubic spline using the monotone cubic interpolation [68]. In a default white-background style, all the flows are visualized in red colors; in another dark-background style, the flows are visualized in white colors (Fig. 8). The saturation of red colors in white-background and the lightness of white colors in dark-background represent the speed of movement, with redder/lighter flows indicating faster movements. The flow thickness represents the volume of movement, with thicker flows indicating larger population. This multivariate visual encoding is consistent with the evaluation result obtained by Perin *et al.* [69] which suggests using color to represent speed. The remaining visual channel of flow size is used to represent volume. Note that users can switch between the two color themes based on their preference.

On the map view, the display of the movement flows are animated to represent their directions. The drawing is done in a tick by tick manner. The default time span of a tick is 20 milliseconds and it can be adjusted to control the animation speed. In each tick, all the flows move ahead a variable length along the flow direction from its current location. This is called the head of the flow. The length of the head is computed by the flow speed multiplying one tick's time span. To enable the animation, we use a semi-transparent mask to cover all the older parts of the flow except the head drawn in the current tick. The drawing of the head and the masking of the older flow continues until the head reaches the end of the flow. The same flow will start again after the animation cycle of the flow. The animation cycle of a flow can be shorter than the time to draw the entire flow, so that there can be multiple heads moving on a flow at the same time.

To drill-down to the underlying trajectory data, interactions are introduced on the urban wind map. When users click on a cell on the main map view, the number of movement devices extracted from the raw data on this cell is displayed in a statistical chart (Fig. 11d). The chart depicts the dynamics of the number of devices in the currently selected time interval and several adjacent intervals before and after. When the anomaly heat map is activated, the chart will show the time series of movement statistics used for anomaly detection (Fig. 10).

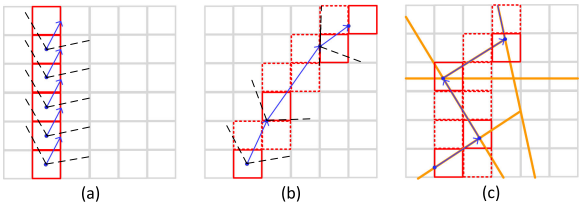



Fig. 6. Flow tracing algorithms: (a) a baseline flow tracing algorithm starting from cell centers, black dashed lines indicate an angular range of search space for the next local flow; (b) a deterministic flow tracing with larger than one search depth (cells in red dotted outlines) and through exact intersection points (blue points on cells in red solid outlines); (c) flow tracing over a map-matched vector field.

### 5.4 Global Flow Generation

To obtain the global movement flows from the stochastic vector field, we propose a pattern-based flow tracing algorithm that works in two steps. In the first step, a list of local movement flows are selected from the vector field of movements and sorted as seed flows. Unlike the original wind map approach that randomly selects seed locations, we extract local flows as seeds which form the spatial patterns discovered in Section 4.4. In the second step, seed flows are traced in the sorted order to generate global movement flows. For each seed flow, the flow tracing algorithm is applied using the seed as the starting direction.

The seed flow selection and sorting considers three spatial patterns of the movement: hub, channel, and large flow. These patterns can be revealed on the map by setting the "Flow Generation" section of the control panel (Fig. 5d). For hub/channel patterns, the top cells having these patterns are detected by the method in Section 4.4 and all the local flows in these cells are added to the selected flow list, sorted by the order of cells. Among flows in the same cell, they are sorted by their flow volume. For the large flow pattern, local flows detected in all the cells are sorted in a same list by their flow volume. Users are allowed to specify a percentage threshold for the top cells/flows displayed as seeds.

From each seed flow direction, we apply a deterministic flow tracing algorithm [70] to detect global movement flows traversing the cells. As shown in Fig. 6b, the algorithm starts from the seed flow and stretches along the flow direction to connect to a few cells ahead, which are called the search space (the angular range indicated by black dashed lines which includes cells in red dotted outlines). The number of such cells is called the search depth (3 by default). All the local movement flows in the cells within the search space are extracted and compared. By default, the flow with the largest volume and satisfying several flow tracing conditions is selected. The selected flow will be connected back to the seed flow at the intersection point on the outline of the selected cell (cells with solid red outlines). The search restarts with the selected flow as the seed and is repeated until no flow satisfying all the flow tracing conditions can be found. The algorithm produces accurate flows as it traverses through the exact intersection points with each cell the flow passes (Fig. 6b). In comparison, the baseline algorithm will go through the centers of cells it passes (Fig. 6a), which is less accurate. In the implementation, four flow tracing conditions are applied: 1) the *angle* between the seed flow and the next flow should be less than a threshold (60 degrees by default);

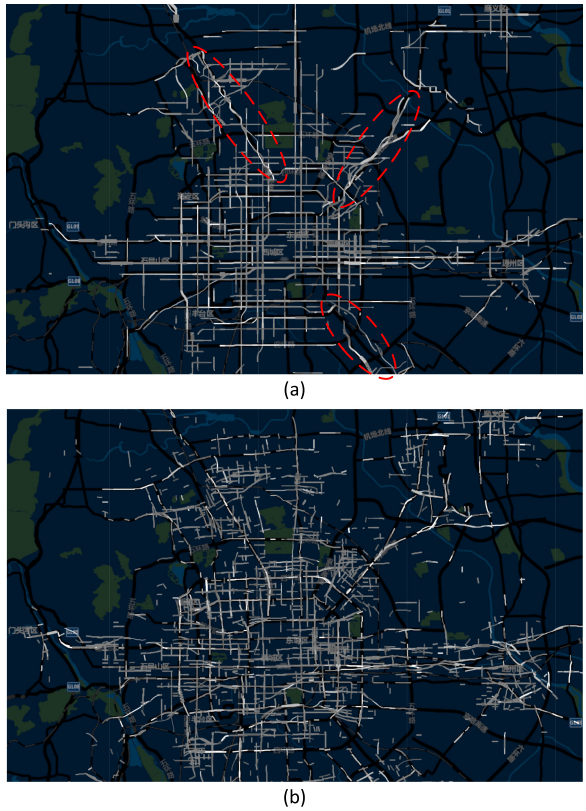



Fig. 7. Visualization of flow tracing result over different movement aggregation methods: (a) local clustering, which better illustrates global movement patterns than (b) map-matching, as shown by three visually detected major channels connecting to the metro of Beijing (red dashed circles in (a)).

2) the *volume* of the next flow should be larger than a ratio of the seed flow (0.1 by default); 3) the accumulative *curvature* on all the selected flows should be upper bounded (120 degrees by default); 4) the maximal *length* of the flow could be upper bounded (no limit by default). These conditions and the search depth can be adjusted by the user in the algorithm configuration panel (Fig. 5c). In addition, we support tree-based flow tracing in that more than one large flows can be traced from each single seed. In the "Flow Tracing" section of the control panel (Fig. 5d), users can switch to a mild (3 flows to trace) or a high (5 flows to trace) degree of branching. We caution that the modification of flow tracing parameters from the default setting will lead to different abstraction of global movement flows. For example, specifying a larger angular range, a deeper search depth, and a smaller volume ratio will assemble less yet longer movement flows, which could reduce the visual clutter in the abstraction, but also introduce more errors in the overall visual summarization. Tuning these parameters in the opposite direction will generate more accurate flow abstractions, though the overall coverage rate on movement data could drop as less local movement flows are connected. User should be aware of these effects in interacting with UrbanMotion visualization.

Note that when the map-matching process is applied, the flow tracing algorithm should allow relaxed conditions to detect possible turns at the intersection of road networks. By default, the *angle* threshold is set to 90 degrees and the *curvature* threshold is set to 180 degrees. An example of flow tracing over the map-matched vector field is given in Fig. 6c. In Fig. 7, the flow tracing results over local clustering

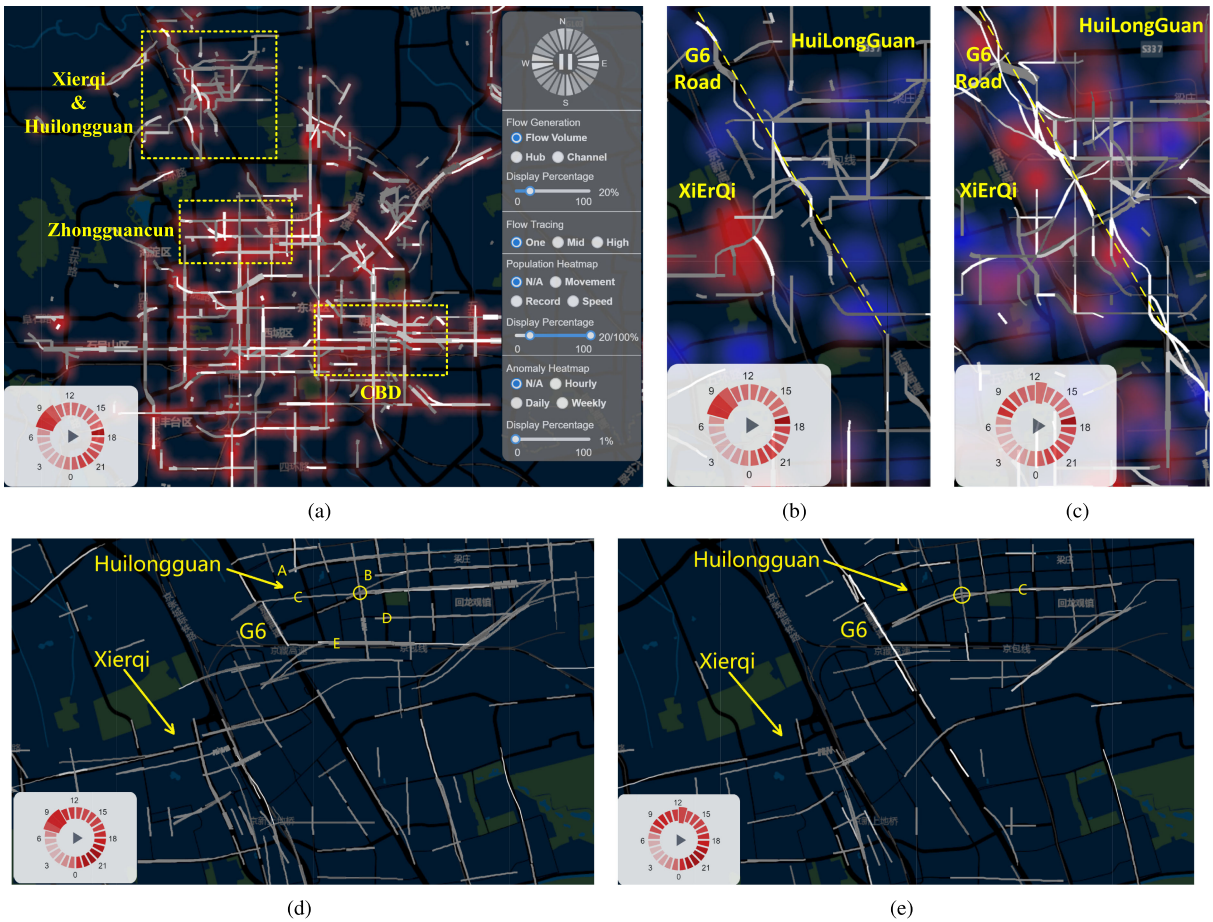



Fig. 8. Commuting analysis of Xierqi&Huilonguan regions on July 12th, 2016: (a) movement visualization on top of speed distributions in the morning (7AM-10AM); (b) using movement distributions in the morning as background heatmap, red indicates destination regions, blue indicates source regions; (c) movement visualizations in 12AM-1PM (noon); (d) switch to the map-matching approach for movement visualization in the morning (7AM-10AM); (e) the map-matching approach in 12AM-1PM (noon).

and map-matching approaches are presented and compared. Because the map-matching process adapts local movements in each cell into separate road segments, the volume of local flows becomes smaller than those out of the clustering approach. The flow tracing over map-matched vector fields tends to detect shorter movement flows on local roads and reveal more detailed movement patterns (Fig. 7b). In comparison, the flow tracing result over local clustering better illustrates global movement patterns in an abstract manner, which can slightly deviate from the actual local movements. As shown in Fig. 7a, by the local clustering approach, it is easier to detect three major transportation channels connecting to the city metro from northwest, northeast, and southeast (red dashed circles).

### 5.5 Population Heatmap

By the multi-layer map design of UrbanMotion, beneath the urban wind map there is a population heatmap correlating movements with the underlying population. The heatmap visualizes the distribution of the statistics of the population related to the movement. As shown in the "Population Heatmap" section of the control panel, three statistics can be visualized. The movement category (Mov.) is selected to show the distribution of net incoming/outgoing population, i.e., source and destination regions (Figs. 8b and 8c), which are computed for each cell by analyzing the trajectory data [49]. The red color indicates the net incoming cell and the blue color indicates the net outgoing cell. The color darkness indicates the size of the net incoming/outgoing population. The record category is selected to show the distribution of the raw location records (Fig. 8a). In the speed category, the heat map shows the distribution of the movement speed in each cell. The more saturated red color indicates a higher speed (Figs. 9a and 9b).

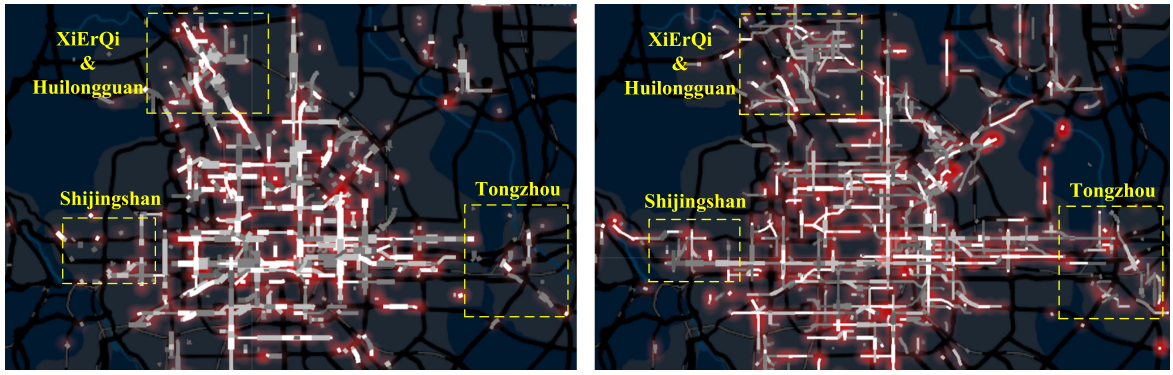
The anomalous temporal patterns detected by the method in Section 4.4 are also drawn as an optional map layer. In the "Anomaly Heatmap" section of the control panel, users can switch among hourly, daily, and weekly anomalies. The anomalous cells are depicted in red, with color saturation indicating the anomaly score (Fig. 10). Only positive anomalies having more movements than usual are displayed.

We caution that when population/anomalous heatmaps are applied, the dark background may be selected. The flow visualization in red under the white background can interfere with heatmaps in the same color and downgrade the display effect.

## 6 Evaluation

### 6.1 Methodology and Experiment

We mainly evaluate UrbanMotion through three case studies, together with experts from three application domains




(a) (b)

Fig. 9. Events detected during the heavy rainstorm in Beijing, July 19th, 2016: (a) movement visualization on top of a speed heatmap, 8AM-9AM in the morning; (b) on the same hour one week ago.

(i.e., urban planners, public safety officers, and business analysts). These domain experts were invited to use our system in a trial to support their everyday job responsibility. During the study, we stood by these experts to further explain the functionality of UrbanMotion and sometimes offered suggestions on its usage. In a sense, the case studies were carried out jointly by both experts and us (system designer). After experts completed their trial with UrbanMotion, a discussion session was held in which experts gave verbal feedback on positive and negative aspects of the system regarding their factual requirements in the everyday job. Experts also advised on possible extensions of the system to better meet their requirements.

In addition to case studies, experiments comparing UrbanMotion with possible alternatives might provide valuable result on system performance. However, these quantitative comparison could be unfair as UrbanMotion is customized for the trajectory data with a long-tailed temporal sparsity. To this end, a qualitative lab experiment was conducted, in which UrbanMotion is compared with another well-known aggregation-based movement visualization technique. This technique was proposed by Andrienko and Andrienko [20] and generally called the cartographic flow map visualization (FlowMap in short). To enable the comparison, the FlowMap visualization was implemented according to Ref. [20] and displayed on top of the UrbanMotion interface. Certain adaptations were made to allow FlowMap to take the new trajectory data. For example, travel trips are detected by the same method in UrbanMotion (Section 4.1), processed by an optional map-matching stage (Section 4.3), and then used as the input trajectory data of FlowMap. A default setting of $MaxRadius = 3\ km$ is applied according to the recommendation in Ref. [20].

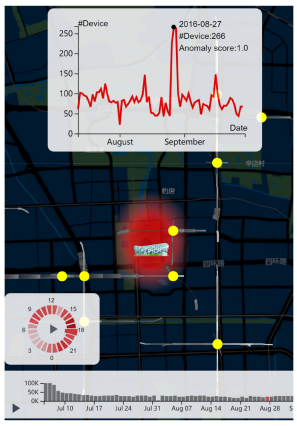



Fig. 10. The movement hotspot on a daily anomaly heatmap, 6-7PM, Aug. 27th, 2016. Movement hubs/statistics are displayed.

The full details of the comparative experiment can be found in Appendix C, available in the online supplemental material. Here we only report key findings from the experiment. On the positive side for UrbanMotion, there are two main reasons to apply UrbanMotion on the long-tailed sparse trajectory data. First, though FlowMap could lead to a higher level of abstraction for urban movements, it only represents a small share of raw data due to the long-tailed data sparsity. For each trajectory in the data, multiple travel trips are extracted, most of which are extremely short in time and lie within the boundary of a single Voronoi cell. Under the default parameter of $MaxRadius = 3\ km$, only 5.42 percent of the raw travel trips extracted are between different cells and visualized in the FlowMap display. In comparison, UrbanMotion applies a default cell size of 500m (approximately $MaxRadius = 250m$), which ensures a much higher ratio of cross-cell travel trips. The direction of within-cell movements are also considered in the trajectory analysis. Using a finer granularity in the FlowMap method leads to an increased visual complexity that disrupts the overview display. From the visualization result, the output of UrbanMotion better corresponds to the actual road network with respect to the output of FlowMap. The second advantage, by integration with the map-matching technique, UrbanMotion can be enhanced to examine local movement patterns in focused areas of a city, e.g., the utilization of major roads. The FlowMap method can also take map-matched movement vectors as input, but the updated visualization has a similar overall movement pattern. This is because the measurement error of location records are generally much smaller than the parameter of $MaxRadius$ in FlowMap (e.g., 3 km). The movement aggregation is only slightly affected after the map-matching, except the Voronoi tessellation. By the aggregation-based design, FlowMap with map-matching still connects the generating point of Voronoi cells and remains a global movement visualization.

Through the experiment, we also identify two major limitations of UrbanMotion in comparison to FlowMap. First,

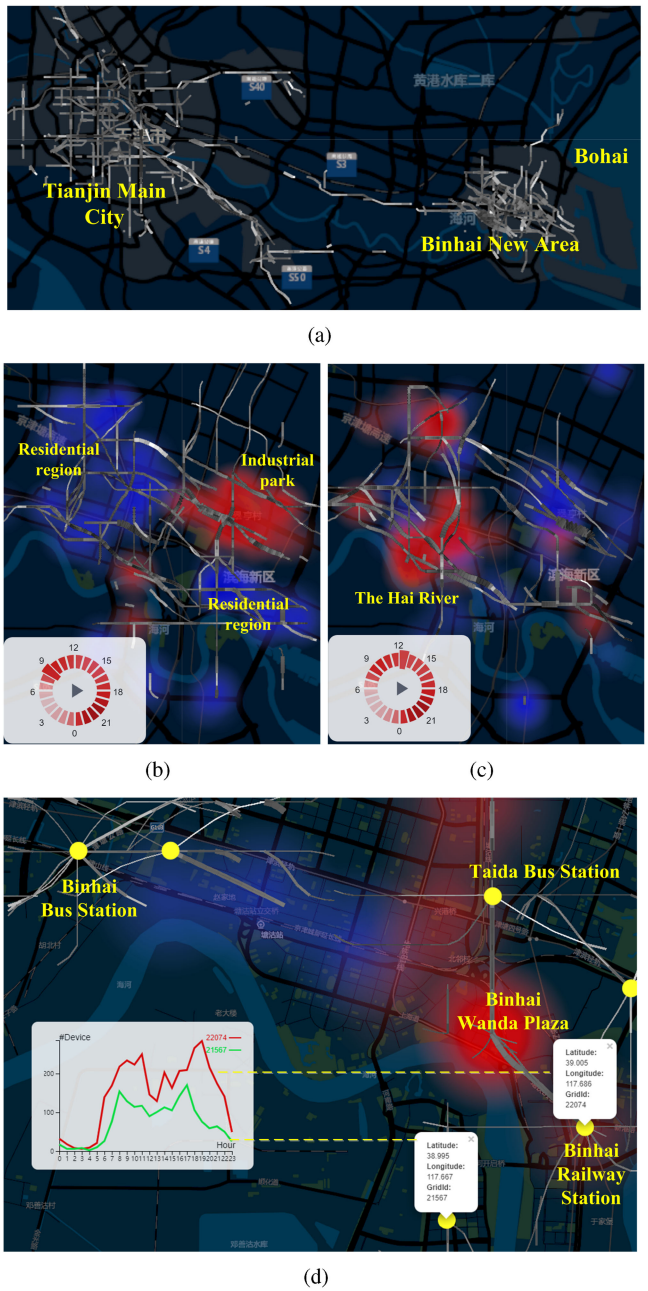



Fig. 11. Commercial site configuration using movement visualization of the Binhai New Area in Tianjin.

UrbanMotion adopts a fix-sized gridding of the city map with relatively small cells. In areas with few movements, the technique is less efficient as the same number of cells need to be processed. FlowMap resolves this issue by introducing Voronoi tessellation. The spatial data aggregation is more efficient under uneven movement distribution. Second, FlowMap is more suitable for online movement visualization over streaming data. The Voronoi tessellation can be computed offline using historical data. The online processing only needs to map each characteristic point into a corresponding Voronoi cell. In comparison, UrbanMotion computes movement clustering on each cell or map-matching on each location record. Because of the limitation in computational complexity, UrbanMotion is mainly used for the replay of existing trajectory data.

## 6.2 Commuting Analysis

Two urban planning experts applied UrbanMotion to study of the commuting pattern in Beijing. The experts pointed out that long commuting times, caused by traffic jams and a high real estate price, are among the most critical urban problems in Beijing. They first picked the day of July 12th in our system, a typical weekday. They chose a time period in the morning (7AM-10AM) to retrieve the population movement of commuting to work. The corresponding movement map is shown in Fig. 5a. In the visualization, a mild KDE smoothing with a bandwidth of $h = 0.5$ is applied to accentuate the overview of movement. The experts found that the result is consistent with their domain knowledge on the road network of Beijing. The city is divided into a northern part and a southern part by the Chang'an avenue, as shown by the blue dashed line in Fig. 5a. This avenue correlates with a bundle of movement flows on the map. By examining movement flows with high volume, the experts discovered five busiest regions during the commuting time: CBD, Financial Street, Zhongguancun, Xierqi&Huilongguan, and Wangjing. These five regions correspond to financial, industrial, high-tech, and IT centers of Beijing. From the animated visualization, the experts noticed that most north-south commutes are from north/south to the center of the city. For east-west movements, both directions to/from the city center are significant. This suggests that the work area of Beijing is close to a rectangle in the city center, with a larger width than height.

Among the busiest city regions, Xierqi&Huilongguan outside the city center appears to be different in commuting patterns. The experts switched to the dark background and activated the population heatmap layer to show the speed distribution. As shown in Fig. 8a, the Xierqi&Huilongguan region suffers from low commuting speed indicated by dimmer flows and a darker underlying heat map, compared with the other regions such as Zhongguancun and CBD. The experts commented that it is known for a while that traffic jams in Xierqi&Huilongguan happen more frequently than the other regions in Beijing. The experts zoomed into this region to analyze local movement patterns. They switched the population heatmap to show the movement source/destination distribution. As depicted in Fig. 8b, during the commute hours of 7AM-10AM, the Xierqi region in the west has a salient red area in the center, i.e., a destination of population. This area is headquarter for major IT companies in Beijing (e.g., Baidu). The Huilongguan region lies to the east of Xierqi and is home to many IT people. The region is mostly in a blue color, which indicates a source of population. The experts pointed out that a key concept in the commute analysis is the work-home balance. From the visualization result, the Xierqi&Huilongguan region has a poor work-home balance because the working center and the living center locate in separate areas.

The experts then selected the hour of 12AM-1PM in the noon as a contrast to the commuting pattern. In Fig. 8c, it is shown that the source and destination of population now distribute more evenly. This pattern again indicates the commute issue in the Xierqi&Huilongguan region. The experts followed up to identify the root cause of the commute issue in this region. They noticed that there are less commuting flows connecting between Xierqi and Huilongguan regions than the commuting flows inside each of the

two regions (Fig. 8b). As they switched the base layer to the road network, it can be observed that Xierqi and Huilongguan regions are cut by G6 (the dashed line in Figs. 8b and 8c), a major incoming road from northern suburbs to the metro of Beijing. The experts told us, currently there are only a few streets crossing G6 and connecting Xierqi & Huilongguan regions. The visual analysis has demonstrated that these streets are the key bottleneck for the local transportation.

In a more detailed study, the experts applied the map-matching approach to understand the actual road network utilization in the Xierqi&Huilongguan region. As given in Fig. 8d, the overall movement patterns shown by map-matching are quite similar to those by the clustering approach (Fig. 8a), which validates the correctness of both approaches for high-level movement visualization. Meanwhile, the map-matched population movements mostly follow the direction of underlying roads, while the clustering of movements do not guarantee a good matching. By map-matched movement visualization, there are five high-utilization west-to-east roads at the Huilongguan region in the morning (annotated from A to E outside the 5th ring road of Beijing, Fig. 8d). After the commute time, only one west-to-east road in the center has a high utilization (road C in Fig. 8e). The population movements during 7AM-10AM and 12AM-1PM also share some common pattern in that there is a same busy intersection as annotated by the yellow circle in Figs. 8d and 8e. This pattern suggests that the same intersection could be the center of transportation in the Huilongguan region for both commute and other trips. By similar analysis, the experts discovered more patterns in this region. For example, the movements on the G6 road, the major incoming channel from the region to the city center, are much faster after the commute time, which validates the severity of traffic jam in this region.

### 6.3 Event Detection

In another study, we worked with a public safety expert responsible for developing the emergency management system for the city government. The daily task of his job is to detect the events occurred in the city range that could have security concerns, e.g., large fires, floods, and severe traffic jams.

On 8AM-9AM, July 19th 2016, some anomalous patterns were observed in our system, as shown in Fig. 9a. The color of most movement flows seemed to be much brighter than they were previously. This indicates a higher commuting speed. The expert quickly retrieved the movement map on the same hour one week ago and turned on the heatmap of population speed for comparison. As shown by the dimmer heat map in Fig. 9b, his finding is confirmed that the movement speed on July 19th is much higher than usual. He also noticed that, in the suburbs of Beijing (e.g., Xierqi & Huilongguan, Shijingshan, Tongzhou, as annotated in Figs. 9a and 9b), there are fewer local movements except high-speed commutes to the center of the city (Fig. 9a). The expert quickly linked this observation to the rainstorm started earlier on that day, which was later known as the longest for Beijing during the recent decade. After contacting the transportation office, he was told that the subway system of the city was having a record-high number of passengers on July 19th, the first day during the rainstorm. It is inferred that, most citizens previously commuting by bus, bike or other ground vehicles, had switched to commute with the subway system. This explains the pattern of a higher commuting speed and less local movement.

In a second trial, the official activated the anomaly heatmap layer in UrbanMotion to better analyze noteworthy events during a long period of time. On 6PM-7PM, Aug. 27th 2016, he observed a hotspot at the Olympic park of Beijing under the daily anomaly setting, as shown in Fig. 10. This indicates an overhigh concentration of movements on 6PM-7PM that day compared with the same hour in the other weekdays. Switching to flow generation by hub cells, several hub areas close to the Olympic park are detected and people are moving to these areas. The official proceeded to click on the anomaly to check details. The statistical chart on top of the area reveals that the number of movement devices in this hotspot is at least four times that of the same hour in normal weekdays. In communication with the management of the Olympic park, the official was told that a concert by a famous band ("May Days") was scheduled to begin on 8:30PM in the Beijing National Stadium (Bird's Nest).

### 6.4 Commercial Site Configuration

We also worked with a business analyst from our data provider. The analyst's key responsibility is to solve customer's problems using the mobile data collection. He mentioned that one frequent customer problem is commercial site configuration, in which customers would like to select an optimal location in the city for their next branch of business or advertisement (e.g., real estates, malls). The analyst applied UrbanMotion to his assigned city of Tianjin. Fig. 11a gives an overview of population movement on a typical weekday of July 5th 2016. It is shown in the visualization that the city can be divided into two parts by urban movements: the main city in the west and the Binhai New Area in the east (harbor of the Bohai Sea). The analyst is especially interested in the Binhai New Area as the district has a rapidly growing economy and accounts for more than a half of the city's GDP. Customer inquiries about site configuration in the new area are increasing.

The analyst first selected the hour of 8AM-9AM. He activated the population heatmap to trace major source and destination regions of the new area during the commuting time. It is found in Fig. 11b that a top destination in the morning is the industrial park of the new area (red hotspot in Fig. 11b), which is probably workplaces for many people. Meanwhile, a lot of citizens travel from southern and western regions of the industrial park, which appear to be residential regions. This pattern provides valuable insights for real estate customers. More apartments can be built in the northern region of the park where there is smaller outgoing commute volume now, considering that the eastern region has already been occupied as the harbor of the new area.

Switching to 12AM-1PM on the same day, as shown in Fig. 11c, top movement destinations change to the areas besides the Hai River where many commercial centers locate. The Binhai New Area is a famous site of tourism, so that not only company employees from the industrial park but also visitors from other cities, move to these commercial centers in the noon. Deploying business or commercial

branches along the Hai River can be advantageous based on the trajectory data and the time for sale can be scheduled to noon for an optimized revenue.

The analyst then switched to 6PM-7PM, a typical hour for commuting to home. As a lot of his customers are commercial brands, he wanted to know where the billboard for brands should be placed to maximize influence. He configured the movement visualization to show top hubs in Binhai New Area. It can be found in Fig. 11d that three of movement hubs co-locate with key transportation stations, including Binhai Railway Station, Binhai Bus Station, and Taida Bus Station. There are multi-directional movement flows from each of these hubs, in which billboard placements can draw the attention of diversified population. As he clicked one of these hubs, the statistical chart in the left shows that the number of moving devices in the hub cell (red curve) is more than four times that of a nearby cell (green curve) in the evening. He activated the population heatmap to show their sources and destinations. The only destination hotspot in the evening lies close to Binhai Wanda Plaza, a major commercial center of the district. Population are moving from the stations to the plaza.

### 6.5 Expert Feedback

Qualitative feedback were collected from experts in both functional and technical perspectives. First, a lot of experts commented that they had rarely looked at an urban movement visualization built over such a large amount of fine-grained data. Local government could benefit from commuting patterns discovered in their districts. Based on these patterns, governments can optimize their public transportation system to attract incoming residents and investments to promote their economic performance. An expert noted that: "The work-home balance is a subtle concept in urban planning that the community have not yet reached a consensus on how to define it. This is probably the tool that can help us in the process to finally solve the problem". Yet, UrbanMotion was initially designed as a generic movement visualization tool for long-tailed sparse trajectory data. Customizations are necessary to upgrade the system for work-home balance analysis. In the analysis side, methods must be introduced to discover work and home locations from each user's sparse trajectory. In the visualization side, beyond a preliminary display of source/destination distributions, land usage information could be combined with discovered work/home locations to construct an integrated work/home map layer. Meanwhile, visualization of movement flows in UrbanMotion by default favors large-volume flows. To better reveal commuting patterns between work and home, new flow seeding algorithms and visual metaphors could be designed to visualize asymmetric movement flows salient during the commute time.

Second, on visualization techniques, most experts found UrbanMotion to be quite different from previous OD-based visualizations (for which domain experts called the "jump wire"). They pointed out, movement flows go through road networks so that it is possible to analyze the detailed route taken by a specific population group. Governments could refer to this information in designing subway blueprints. Real estate agents could take this map to a presentation to demonstrate the convenience of commuting around their housing projects. A large trunk of business site selection tasks could also benefit from UrbanMotion visualization.

Third, some experts stated that they need more statistical charts and map layers presented together with the urban wind map. Demographics of population can be shown to complement commuting analysis. For example, the gender of each trajectory could be quite helpful if displayed, though it seems hard to acquire such information due to the privacy concern. In future, UrbanMotion could be enhanced by adding typical features of a base map [71] and an abstracted view of movements, e.g., POI distributions, histogram of movement speeds, and node-link diagram of movement networks.

## 7 Discussion

UrbanMotion supports two types of aggregations to abstract huge amount of trajectory data: movement clustering and map-matching. While the result in Fig. 7 and our case study have demonstrated that clustering and map-matching approaches are more suitable for global and local movement visualizations respectively, a more detailed discussion on the trade-off of each approach could be helpful. First, comparing visualizations in Fig. 7a and 7b, the overall movement patterns are quite similar, which validates the correctness of both approaches for high-level movement visualization. The clustering approach groups multiple local movements into the same aggregated movement flow, trading off some accuracy in local movement visualization for a better abstraction of overall movements. On the other hand, the map-matching approach calibrates each local direction for a more accurate representation of movements, while the overall visualization tends to be more cluttered with a larger number of shorter movement flows. Second, from the perspective of performance and cost, both methods have a computational complexity of $O(n)$ ($n$ is the number of local movement vectors), which is scalable to support a huge amount of trajectory data. The clustering approach has a much lower cost in that it only needs to update two window statistics in clustering each vector, while the map-matching approach needs to compute matching scores from each vector to $k$ nearby road segments, and then compare these $k$ scores. In our implementation, the running time of map-matching is about 50 times longer than that of the movement clustering (Section 4.5). Last, it is impossible to combine the two approaches to enjoy the best of both worlds. Clustering on map-matched movement vectors again makes aggregated flows deviate from road networks and are not accurate for local movement visualization. Map-matching clustered local flows, on the other hand, introduces more errors to the abstract visualization.

On visualization, the urban wind map applies animations to illustrate movement directions of global flows on the map, in addition to multivariate visual encodings to represent flow volume, speed, etc. Previous literature by Robertson *et al.* [72] has concluded that animation is not as effective as static displays, such as small multiples, regarding analytics. In UrbanMotion, a pause mode is introduced in which users could stop the animation and access the static visualization for analysis. Most case studies and screenshots in this paper are obtained in the pause mode.

Animations may only be used as a supplementary option for presentation. We also note that the animated presentation can be cluttered because of the bidirectional or closely intersected movements. Users could use the flow demultiplex interaction to only show the movements in certain ranges of directions. The resulting visualization becomes a less cluttered overview of selected movement flows.

The source code of this work is available at: https://github.com/visdata/UrbanMotion/.

## 8 Conclusion and Future Work

In this work, we study the visual analytics of population movements from billions of mobile location records in modern cities. Motivated by the long-tailed sparsity of underlying trajectory, we proposed an integrated technique called UrbanMotion. The technique is composed of movement extraction, aggregation, and pattern discovery algorithms in the analysis side. In the visualization side, a flow generation algorithm, an improved wind map metaphor, and a multilayer visualization interface are designed and implemented. The interface supports multiple customized interactions and is complemented by several control and information panels for flexible usage. We evaluate UrbanMotion in three real-world case studies on commuting analysis, event detection, and commercial site configuration, as well as a qualitative lab experiment comparing with classical aggregation-based movement visualization methods. Both case study result and expert feedback demonstrate the effectiveness of our system in visually analyzing population movements in modern cities.

We plan to extend this work in three dimensions. First, on visualizing local movement patterns, we want to explore the possibility to incorporate trajectory context. In the map-matching approach, a series of local movement vectors belonging to the same trajectory can be map-matched together in a new path-matching paradigm. Additional constraints such as turn restrictions can be considered in this trajectory-aware flow generation. Second, while this work supports movement visualization in an initial scenario over historical trajectory data, a more realistic scenario of online visualization over live trajectory streams poses multifaceted challenges. Incremental flow computation mechanisms should be developed in the back-end to support the streaming of trajectory data. In the visualization side, appropriate animated transitions could be designed to deliver changes of population movement in a smooth and effective manner. For online visualization of live trajectory streams, another useful addition is to automatically detect emerging population movement patterns. This involves collecting relevant movement patterns from application domains (e.g., traffic jams, road constructions), developing fast online pattern detection algorithms, and designing feature visualization methods. Third, on visualizing global movement patterns, UrbanMotion allows us to derive many findings as indicated in case studies. Yet, the visual complexity is still considerably high due to the use of small cells in aggregation. Applying much bigger grid settings could reduce the visual complexity, but will also introduce distortions to the actual global movement pattern. In the next step, we will study the possibility of deriving an adaptive, hierarchical gridding approach to cope with uneven movement distribution and different geographic scales.

## Acknowledgments

This work was supported by NSFC Grants 61772504, U1609217, the Fundamental Research Funds for the Central Universities, and SKLSDE. Lei Shi and Congcong Huang contribute equally in this work.

## References

[1] A. M. MacEachren, *How Maps Work: Representation, Visualization, and Design*. New York, NY, USA: Guilford Press, 2004.
[2] D. Guo, "Flow mapping and multivariate visualization of large spatial interaction data," *IEEE Trans. Vis. Comput. Graph.*, vol. 15, no. 6, pp. 1041–1048, Nov./Dec. 2009.
[3] G. Andrienko, N. Andrienko, P. Bak, D. Keim, and S. Wrobel, *Visual Analytics of Movement*. Berlin, Germany: Springer, 2013.
[4] N. Andrienko and G. Andrienko, "Visual analytics of movement: An overview of methods, tools and procedures," *Inf. Vis.*, vol. 12, no. 1, pp. 3–24, 2013.
[5] H. Liu, Y. Gao, L. Lu, S. Liu, L. Ni, and H. Qu, "Visual analysis of route diversity," in *Proc. IEEE Conf. Vis. Analytics Sci. Technol.*, 2011, pp. 171–180.
[6] H. Guo, Z. Wang, B. Yu, H. Zhao, and X. Yuan, "TripVista: Triple perspective visual trajectory analytics and its application on microscopic traffic data at a road intersection," in *Proc. IEEE Pacific Vis. Symp.*, 2011, pp. 163–170.
[7] F. Miranda *et al.*, "Urban pulse: Capturing the rhythm of cities," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 1, pp. 791–800, Jan. 2017.
[8] Y. Zheng, W. Wu, Y. Chen, H. Qu, and L. M. Ni, "Visual analytics in urban computing: An overview," *IEEE Trans. Big Data*, vol. 2, no. 3, pp. 276–296, Sep. 2016.
[9] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "T-drive: Enhancing driving directions with taxi drivers' intelligence," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 1, pp. 220–232, Jan. 2013.
[10] C. Ratti, R. Pulselli, S. Williams, and D. Frenchman, "Mobile landscapes: using location data from cellphones for urban analysis," *Environ. Planning B, Planning Des.*, vol. 33, no. 5, pp. 727–748, 2006.
[11] D. Liu *et al.*, "SmartAdP: Visual analytics of large-scale taxi trajectories for selecting billboard locations," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 1, pp. 1–10, Jan. 2017.
[12] N. Shoval, "Tracking technologies and urban analysis," *Cities*, vol. 25, no. 1, pp. 21–28, 2008.
[13] M. Rahmani and H. N. Koutsopoulos, "Path inference from sparse floating car data for urban networks," *Transp. Res. Part C: Emerg. Technol.*, vol. 30, pp. 41–54, 2013.
[14] I. Sanaullah, M. Quddus, and M. Enoch, "Developing travel time estimation methods using sparse GPS data," *J. Intell. Transp. Syst.*, vol. 20, no. 6, pp. 532–544, 2016.
[15] W. Li, D. Nie, D. Wilkie, and M. C. Lin, "Citywide estimation of traffic dynamics via sparse GPS traces," *IEEE Intell. Transp. Syst. Mag.*, vol. 9, no. 3, pp. 100–113, Fall 2017.
[16] S. Chen *et al.*, "Interactive visual discovering of movement patterns from sparsely sampled geo-tagged social media data," *IEEE Trans. Vis. Comput. Graphics*, vol. 22, no. 1, pp. 270–279, Jan. 2016.
[17] G. Andrienko and N. Andrienko, "A general framework for using aggregation in visual exploration of movement data," *The Cartographic J.*, vol. 47, no. 1, pp. 22–40, 2010.
[18] J. Wood, J. Dykes, and A. Slingsby, "Visualisation of origins, destinations and flows with OD maps," *Cartogr. J.*, vol. 47, no. 2, pp. 117–129, 2010.
[19] D. Guo, J. Chen, A. M. MacEachren, and K. Liao, "A visualization system for space-time and multivariate patterns (VIS-STAMP)," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 6, pp. 1461–1474, Nov./Dec. 2006.
[20] N. Andrienko and G. Andrienko, "Spatial generalization and aggregation of massive movement data," *IEEE Trans. Vis. Comput. Graphics*, vol. 17, no. 2, pp. 205–219, Feb. 2011.
[21] S. Rinzivillo, D. Pedreschi, M. Nanni, F. Giannotti, N. Andrienko, and G. Andrienko, "Visually driven analysis of movement data by progressive clustering," *Inf. Vis.*, vol. 7, no. 3/4, pp. 225–239, 2008.

[22] G. Andrienko *et al.*, "Space-in-time and time-in-space self-organizing maps for exploring spatiotemporal patterns," *Comput. Graph. Forum*, vol. 29, no. 3, pp. 913–922, 2010.

[23] U. Demšar and K. Virrantaus, "Space–time density of trajectories: exploring spatio-temporal patterns in movement data," *Int. J. Geogr. Inf. Sci.*, vol. 24, no. 10, pp. 1527–1542, 2010.

[24] I. Boyandin, E. Bertini, P. Bak, and D. Lalanne, "Flowstrates: An approach for visual exploration of temporal origin-destination data," *Comput. Graph. Forum*, vol. 30, no. 3, pp. 971–980, 2011.

[25] J. Wood, A. Slingsby, and J. Dykes, "Visualizing the dynamics of london's bicycle hire scheme," *Cartographica*, vol. 46, no. 4, pp. 239–251, 2011.

[26] N. Ferreira, J. Poco, H. T. Vo, J. Freire, and C. T. Silva, "Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips," *IEEE Trans. Vis. Comput. Graphics*, vol. 19, no. 12, pp. 2149–2158, Dec. 2013.

[27] T. Schreck, J. Bernard, T. Von Landesberger , and J. Kohlhammer, "Visual cluster analysis of trajectory data with interactive kohonen maps," *Inf. Vis.*, vol. 8, no. 1, pp. 14–29, 2009.

[28] M. Wattenberg and F. B. Viégas, "Wind map - hint.fm," 2012. [Online]. Available: http://hint.fm/wind/

[29] J. A. Dykes and D. M. Mountain, "Seeking structure in records of spatio-temporal behaviour: Visualization issues, efforts and applications," *Comput. Statist. Data Anal.*, vol. 43, no. 4, pp. 581–603, 2003.

[30] G. Andrienko, N. Andrienko, G. Fuchs, and J. Wood, "Revealing patterns and trends of mass mobility through spatial and temporal abstraction of origin-destination movement data," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 9, pp. 2120–2136, Sep. 2017.

[31] N. Willems, H. V. D. Wetering, and J. J. V. Wijk, "Visualization of vessel movements," *Comput. Graph. Forum*, vol. 28, pp. 959–966, 2009.

[32] R. Scheepens, N. Willems, H. Van de Wetering, G. Andrienko, N. Andrienko, and J. J. Van Wijk , "Composite density maps for multivariate trajectories," *IEEE Trans. Vis. Comput. Graphics*, vol. 17, no. 12, pp. 2518–2527, Dec. 2011.

[33] R. Scheepens, N. Willems, H. van de Wetering, and J. J. Van Wijk, "Interactive visualization of multivariate trajectory data with density maps," in *Proc. IEEE Pacific Vis. Symp.*, 2011, pp. 147–154.

[34] T. Von Landesberger, F. Brodkorb, P. Roskosch, N. Andrienko, G. Andrienko, and A. Kerren, "MobilityGraphs: Visual analysis of mass mobility dynamics via spatio-temporal graphs and clustering," *IEEE Trans. Vis. Comput. Graphics*, vol. 22, no. 1, pp. 11–20, Jan. 2016.

[35] G. Andrienko *et al.*, "Thematic patterns in georeferenced tweets through space-time visual analytics," *Comput. Sci. Eng.*, vol. 15, no. 3, pp. 72–82, 2013.

[36] Y. Yang, T. Dwyer, S. Goodwin, and K. Marriott, "Many-to-many geographically-embedded flow visualisation: An evaluation," *IEEE Trans. Vis. Comput. Graphics*, vol. 23, no. 1, pp. 411–420, Jan. 2017.

[37] P. Jankowski, N. Andrienko, G. Andrienko, and S. Kisilevich, "Discovering landmark preferences and movement patterns from photo postings," *Trans. GIS*, vol. 4, no. 6, pp. 833–852, 2010.

[38] Y. Ma, T. Lin, Z. Cao, C. Li, F. Wang, and W. Chen, "Mobility viewer: An eulerian approach for studying urban crowd flow," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 96, pp. 2627–2636, Sep. 2016.

[39] M.-J. Kraak, "The space-time cube revisited from a geovisualization perspective," in *Proc. 21st Int. Cartographic Conf.*, 2003, pp. 1988–1996.

[40] N. R. Hedley, C. H. Drew, E. A. Arfin, and A. Lee, "Hagerstrand revisited: Interactive space-time visualizations of complex spatial data," *Informatica: Int. J. Comput. Informat.*, vol. 23, pp. 155–168, 1999.

[41] M.-P. Kwan, "Interactive geovisualization of activity-travel patterns using three-dimensional geographical information systems: A methodological exploration with a large data set," *Transp. Res. Part C, Emerg. Technol.*, vol. 8, no. 1/6, pp. 185–203, 2000.

[42] T. Kapler and W. Wright, "Geo time information visualization," *Inf. Vis.*, vol. 4, no. 2, pp. 136–146, 2005.

[43] C. Tominski, H. Schumann, G. Andrienko, and N. Andrienko, "Stacking-based visualization of trajectory attribute data," *IEEE Trans. Vis. Comput. Graphics*, vol. 18, no. 12, pp. 2565–2574, Dec. 2012.

[44] Z. Wang *et al.*, "Visual exploration of sparse traffic trajectory data," *IEEE Trans. Vis. Comput. Graphics*, vol. 20, no. 12, pp. 1813–1822, Dec. 2014.

[45] J. Poco, H. Doraiswamy, H. T. Vo, J. L. Comba, J. Freire, and C. T. Silva, "Exploring traffic dynamics in urban environments using vector-valued functions," *Comput. Graph. Forum*, vol. 34, no. 3, pp. 161–170, 2015.

[46] X. Li *et al.*, "Prediction of urban human mobility using large-scale taxi traces and its applications," *Front. Comput. Sci.*, vol. 6, no. 1, pp. 111–121, 2012.

[47] F. Calabrese, F. C. Pereira, G. Di Lorenzo, L. Liu, and C. Ratti, "The geography of taste: Analyzing cell-phone mobility and social events," in *Proc. Int. Conf. Pervasive Comput.*, 2010, pp. 22–37.

[48] S. Jiang, G. A. Fiore, Y. Yang, J. Ferreira Jr , E. Frazzoli, and M. C. González, "A review of urban computing for mobile phone traces: Current methods, challenges and opportunities," in *Proc. 2nd ACM SIGKDD Int. Workshop Urban Comput.*, 2013, Art. no. 2.

[49] L. Shi, "Mobility inference from long-tailed sparse trajectories," 2020, *arXiv: 2001.07636*.

[50] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discov. Data Mining*, 1996, pp. 226–231.

[51] M. A. Quddus, W. Y. Ochieng, and R. B. Noland, "Current map-matching algorithms for transport applications: State-of-the art and future research directions," *Transp. Res. Part C, Emerg. Technol.*, vol. 15, no. 5, pp. 312–328, 2007.

[52] D. Bernstein *et al.*, *An Introduction to Map Matching for Personal Navigation Assistants*, Tech. Rep., New Jersey TIDE Center, 1996.

[53] J. S. Greenfeld, "Matching GPS observations to locations on a digital map," in *Proc. 81th Annu. Meeting Transp. Res. Board*, 2002, vol. 1, pp. 164–173.

[54] W. Y. Ochieng, M. A. Quddus, and R. B. Noland, "Map-matching in complex urban road networks," vol. 55, no. 2, pp. 1–14, 2003.

[55] D. Obradovic, H. Lenz, and M. Schupfner, "Fusion of map and sensor data in a modern car navigation system," *J. VLSI Signal Process. Syst. Signal Image Video Technol.*, vol. 45, no. 1/2, pp. 111–122, 2006.

[56] M. A. Quddus, R. B. Noland, and W. Y. Ochieng, "A high accuracy fuzzy logic based map matching algorithm for road transport," *J. Intell. Transp. Syst.*, vol. 10, no. 3, pp. 103–115, 2006.

[57] P. Newson and J. Krumm, "Hidden Markov map matching through noise and sparseness," in *Proc. 17th ACM SIGSPATIAL Int. Conf. Advances Geographic Inf. Syst.*, 2009, pp. 336–343.

[58] A. Thiagarajan, L. S. Ravindranath, H. Balakrishnan, S. Madden, and L. Girod, "Accurate, low-energy trajectory mapping for mobile devices," in *Proc. 8th USENIX Conf. Netw. Syst. Des. Implementation*, 2011, pp. 267–280.

[59] C. Y. Goh, J. Dauwels, N. Mitrovic, M. T. Asif, A. Oran, and P. Jaillet, "Online map-matching based on hidden Markov model for real-time traffic sensing applications," in *Proc. 15th Int. IEEE Conf. Intell. Transp. Syst.*, 2012, pp. 776–781.

[60] N. R. Velaga, M. A. Quddus, and A. L. Bristow, "Developing an enhanced weight-based topological map-matching algorithm for intelligent transport systems," *Transp. Res. Part C, Emerg. Technol.*, vol. 17, no. 6, pp. 672–683, 2009.

[61] J. B. Freeman and R. Dale, "Assessing bimodality to detect the presence of a dual cognitive process," *Behav. Res. Methods*, vol. 45, no. 1, pp. 83–97, 2013.

[62] C. M. Bishop, *Pattern Recognition and Machine Learning*. Berlin, Germany: Springer, 2006.

[63] S. J. Roberts, "Novelty detection using extreme value statistics," *IEE Proc. Vis. Image Signal Process.*, vol. 146, no. 3, pp. 124–129, Jan. 1999.

[64] Node.js, 2019. [Online]. Available: https://nodejs.org/

[65] T. McLoughlin, R. S. Laramee, R. Peikert, F. H. Post, and M. Chen, "Over two decades of integration-based, geometric flow visualization," *Comput. Graph. Forum*, vol. 29, no. 6, pp. 1807–1829, 2010.

[66] Vue.js javascript framework, 2019. [Online]. Available: https://vuejs.org/

[67] Leaflet: A javascript library for interactive maps, 2019. [Online]. Available: https://leafletjs.com/

[68] M. Steffen, "A simple method for monotonic interpolation in one dimension," *Astron. Astrophys.*, vol. 239, 1990, Art. no. 443.

[69] C. Perin, T. Wun, R. Pusch, and S. Carpendale, "Assessing the graphical perception of time and speed on 2D+ time trajectories," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 1, pp. 698–708, Jan. 2018.

[70] S. Mori, B. J. Crain, V. P. Chacko, and P. Van Zijl, "Three-dimensional tracking of axonal projections in the brain by magnetic resonance imaging," *Ann. Neurol.*, vol. 45, no. 2, pp. 265–269, 1999.

[71] Esri arcgis software products, 2019. [Online]. Available: https://www.arcgis.com/

[72] G. Robertson, R. Fernandez, D. Fisher, B. Lee, and J. Stasko, "Effectiveness of animation in trend visualization," *IEEE Trans. Vis. Comput. Graphics*, vol. 14, no. 6, pp. 1325–1332, Nov./Dec. 2008.

**Lei Shi** received the BS, MS, and PhD degrees from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, in 2003, 2006, 2008, respectively. He is currently a professor with ACT & SKLSDE, School of Computer Science and Engineering, Beihang University, Beijing, China. Previously, he was a professor in SKLCS, Institute of Software, Chinese Academy of Sciences, Beijing, China. His research interests include information visualization, visual analytics, and data mining. He is the recipient of the VAST Challenge Award twice, in 2010 and 2012.

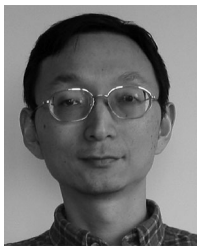
**Congcong Huang** received the BS degree from the Department of Computer Science, Sichuan University, Chengdu, China, and the MS degree from SKLCS, Institute of Software, Chinese Academy of Sciences, Beijing, China. She is currently working toward the graduate degree at SKLCS, Institute of Software, Chinese Academy of Sciences, Beijing, China. Her research interests include data visualization and visual analytics.

**Meijun Liu** received the BS degree from the School of Computer Science and Engineering, Northeastern University, Boston, Massachusetts. She is currently working toward the graduate degree in ACT & SKLSDE, School of Computer Science and Engineering, Beihang University, Beijing, China. Her research interest include spatial data visualization.

**Jia Yan** received the PhD degrees from the University of Chinese Academy of Sciences, Beijing, China, in 2015. He is currently an associate professor in TCA/SKLCS, Institute of Software, Chinese Academy of Sciences, Beijing, China. His research interests span visualization for security, malware analysis and software security.

**Tao Jiang** (Student Member, IEEE) received the BS degree from the School of Software, Beijing Institute of Technology, Beijing, China, in 2015. He is currently working toward the graduate degree at SKLCS, Institute of Software, Chinese Academy of Sciences, Beijing, China. His research interests include computer graphics, data visualization, visual analytics and web development.

**Zhihao Tan** received the BS degree from Beijing Jiaotong University, Beijing, China. He is currently working toward the graduate degree in Institute of Software, Chinese Academy of Sciences, Beijing, China. His research interests include visual analytics and data mining.



**Yifan Hu** received the BS and MS degrees in applied mathematics from Shanghai Jiao-Tong University, Shanghai, China, and the PhD degree in optimization from Loughborough University, United Kingdom. He is currently a principal research scientist at Yahoo Labs, having previously worked at AT&T Labs. He is a contributor to the Graphviz graph drawing system. His research interests include data mining, information visualization, and algorithms.

**Wei Chen** is currently a professor in State Key Lab of CAD&CG at Zhejiang University, Hangzhou, China. His current research interests include visualization and visual analytics. He has published more than 40 IEEE/ACM Transactions and IEEE VIS papers. He served in many leading conferences and journals (PacificVis, ChinaVis steering committee, IEEE LDAV and ACM SIGGRAPH Asia VisSym). He is also the associate EIC of JVLC, an associate editor of the *IEEE Computer Graphics and Applications* and the *Journal of Vision*.

**Xiatian Zhang** is currently the chief data scientist of TalkingData. He has long engaged in machine learning research and has dozens of research papers in publication. He also has much of experience of the applications of machine learning, such as recommender systems, and computing advertising. He is managing the Data Science Center of TalkingData, which develop new technology and application to support the core business, and explore future direction of TalkingData.