# Calibrating Trajectory Data for Similarity-based Analysis

Han Su[†] Kai Zheng[†] Haozhou Wang[†] Jiamin Huang[‡] Xiaofang Zhou[†]
[†]University of Queensland, Australia [‡]Nanjing University, China
[†]{h.su1,uqkzheng, h.wang16,uqxzhou}@uq.edu.au [‡]hjm10@software.nju.edu.cn

## ABSTRACT

Due to the prevalence of GPS-enabled devices and wireless communications technologies, spatial trajectories that describe the movement history of moving objects are being generated and accumulated at an unprecedented pace. Trajectory data in a database are intrinsically *heterogeneous*, as they represent discrete approximations of original continuous paths derived using different sampling strategies and different sampling rates. Such heterogeneity can have a negative impact on the effectiveness of trajectory similarity measures, which are the basis of many crucial trajectory processing tasks. In this paper, we pioneer a systematic approach to *trajectory calibration* that is a process to transform a heterogeneous trajectory dataset to one with (almost) unified sampling strategies. Specifically, we propose an anchor-based calibration system that aligns trajectories to a set of anchor points, which are fixed locations independent of trajectory data. After examining four different types of anchor points for the purpose of building a stable reference system, we propose a geometry-based calibration approach that considers the spatial relationship between anchor points and trajectories. Then a more advanced model-based calibration method is presented, which exploits the power of machine learning techniques to train inference models from historical trajectory data to improve calibration effectiveness. Finally, we conduct extensive experiments using real trajectory datasets to demonstrate the effectiveness and efficiency of the proposed calibration system.

## Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Spatial databases and GIS

## Keywords

similarity measure, trajectory database, trajectory calibration

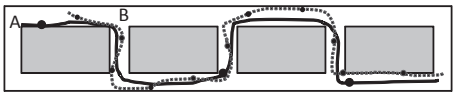
## 1. INTRODUCTION

Driven by major advances in sensor technology, GPS-enabled mobile devices and wireless communications, a large amount of data recording the motion history of moving objects, known as *trajectories*, are currently generated and managed in scores of application domains. This inspires a tremendous amount of research effort on analyzing large scale trajectory data from a variety of aspects in the last decade. Representative work includes designing effective trajectory indexing structures [28, 1, 25, 2, 8], efficient trajectory query processing [32, 5, 10], and mining knowledge/patterns from trajectories [20, 17, 16, 22], to name a few.

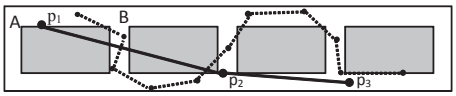
In theory, a trajectory can be modelled by a continuous function mapping time to space; in practice, however, a trajectory can only be represented by a discrete sequence of locations sampled from the continuous movement of the moving object, due to limitations of location acquisition technologies. In other words, when a raw trajectory is reported to the server and stored in the database, it is just a *sample* of the original travel history. The sampling strategies used to generate trajectory data can vary significantly for several reasons. First of all, the sampling methods can be different, such as distance-based methods (e.g., report every 100m), time-based methods (e.g., report every 30s) and predication-based methods (e.g., report when the actual location exceeds a certain distance from the predicted location). Secondly, different parameters may be used even for the same sampling strategy. For example, based on the time-based sampling strategy a geologist equipped with specialized GPS-devices can report her locations with high frequency (say every 5 seconds) while a casual mobile phone user may only provide one location record every couple of hours or even days (via, for example, a Web check-in service). Such variations can also be imposed by external factors (such as availability of on-device battery and wireless signal) and may change at owner's discretion.

As such, trajectory data in real world database applications are *heterogeneous* by nature. This, however, can be problematic when these heterogeneous trajectories are processed directly. For example, as we shall illustrate later, it does not make much sense to compare two trajectories obtained using different sampling strategies by directly applying existing trajectory similarity measures like Euclidean distance, DTW [19], LCSS [18] or EDR [4]. This is because these measures are all based on the spatial proximity between sampled locations, and hence easily affected by the sampling strategies adopted. Consider in Figure 1(a) that two moving objects follow highly similar routes in an urban area, but adopt different sampling strategies. As a result, the raw trajectory (denoted by the solid line) of object $A$ has fewer sample points than that of $B$ (denoted by the dashed line). Figure 1(b) illustrates the actual trajectory data stored in the database. It is easy to observe that the two trajectories may have a greater distance (than they are supposed to be) based on most trajectory similarity measures. A system relying on trajectory similarity search may produce misleading results to the users if these trajectories are processed without the awareness of this issue. Therefore, this issue of trajectory heterogeneity

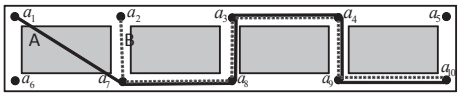

*SIGMOD'13,* June 22–27, 2013, New York, New York, USA.




(a) Original routes




(b) Raw trajectories




(c) After calibration

**Figure 1: Motivation of calibration**

must be dealt with in order to make meaningful similarity-based trajectory processing.

## 1.1 Problem Analysis: A Case Study

Now let us examine the impact of sampling strategies on trajectory similarity analysis through a case study. We test with four commonly used trajectory distance measures: Euclidean Distance, DTW [19], LCSS [18] and EDR [4]. These distance measures perform reasonably well according to the reported results. However, whether it is explicitly mentioned or not, a prerequisite for these measures to be effective is that the sampling strategies of all trajectory data must be compatible (that is, very similar). In the sequel, we will demonstrate that the effectiveness of these distance measures are highly sensitive to how trajectory data are sampled.

In this experiment, we first select 500 densely sampled trajectories on a road network as the original routes. For each of them, we adopt a time-based sampling method with variable sampling rates of 10, 20, 30, 60 and 100 seconds (denoted by $T_{10}$, $T_{20}$, $T_{30}$, $T_{60}$ and $T_{100}$, respectively). Then we choose $T_{30}$ as the baseline trajectory and calculate the distance between $T_{30}$ and other variants using these four trajectory distance measures. The average measured normalized distance values are reported in Table 1. One can see that, although all the trajectories re-sampled using different sampling rates refer to exactly the same original trajectory, the reported distance values vary widely no matter which distance measure is adopted. Consequently, all the data analysis tasks relying on such distance measures can be ineffective as similar trajectories may not be properly identified as such. The root cause of this phenomenon is that all these distance measures, as well as many other trajectory processing techniques, are merely sample-based. In other words, all the distance evaluations are performed between sample points. These distance measures can work only based on some assumptions such as very dense point sampling. As we discussed earlier, these assumptions may no longer hold for many real-life trajectory datasets. This case study also illustrates the severity of the trajectory heterogeneity problem.

**Table 1: Effect of sampling rates**

| Rate \ Distance | ED | DTW | LCSS | EDR |
|---|---|---|---|---|
| 10 | 0.35 | 0.21 | 0.71 | 0.75 |
| 20 | 0.21 | 0.09 | 0.27 | 0.37 |
| 30 | 0 | 0 | 0 | 0 |
| 60 | 0.24 | 0.15 | 0.47 | 0.53 |
| 100 | 0.25 | 0.21 | 0.75 | 0.68 |

## 1.2 Challenges and Contributions

With the observation and awareness of this heterogeneity problem for trajectory data, a *calibration* process is necessary before raw trajectory data can be used for subsequent data analytics to transform a set of heterogeneous trajectories into one with more unified sampling strategies. The goal of this calibration processing is to reduce or even eliminate the negative impact of the sampling strategies on measuring trajectory similarity. In other words, all trajectories after calibration should better resemble their original continuous routes thus can be more accurately compared with each other regardless of the sampling strategies used in generating the raw trajectories. In order to achieve this goal, we need to construct a reference system that is trajectory-independent and then *rewrite* raw trajectories based on the same reference system.

It is a non-trivial task to perform trajectory calibration. First, building a good reference set is the stepping stone for the entire system. Since our goal is to rewrite the trajectory data using the reference set, we expect a good reference set to be stable, independent of data sources, and have a strong association with the trajectory data. The first and second properties are essential for producing trajectories in a unified form, while the third property ensures that the calibration process will not introduce a large deviation from the original routes. Trajectory calibration may encounter three circumstances when rewriting a trajectory with the reference set: 1) a trajectory point may need to be shifted and aligned onto the reference; 2) some trajectory points may need to be removed or merged (when the sampling rate is higher than necessary); 3) some new trajectory points may need to be inserted (when the sampling rate is too low), all in the context of the chosen reference system. Further, the criteria to judge the goodness of the calibration results need to be established, for the system to enforce efficiently and effectively and for the users to understand to what extent the calibration can improve the data analysis results.

In this paper, we propose an *anchor-based calibration system* for heterogeneous trajectory data. It comprises two components: a reference system and a calibration method. For the first component, we present several reference systems by defining different types of anchor points (space-based, data-based, POI-based and feature-based), which are fixed small regions in the underlying space. A series of strategies are designed for the calibration component, including the methods to insert anchor points to trajectories in order to make them more complete without scarifying geometric resemblance to the original routs. To this end, we first derive the transfer relationship among anchor points by learning from a historical trajectory dataset, and then infer the most probable alignment sequence and complement points with high accuracy by exploiting the power of the Hidden Markov Model and Baysian Network. We also perform an empirical study to examine the effect of calibration process, using the trajectories with a very high sampling rate as the ground truth data (the original route) and generate raw trajectories using a different sampling rate in a controlled way. Then we measure the similarities between the raw trajectories, with a set of commonly used distance functions, before and after the calibration process. We will show in the experiment that while the similarities between the raw trajectories heavily depend on the sampling rates, with calibrated trajectories their similarities always highly resemble those of the original routes for a wide range of sampling rates.

Continuing with the previous example in Figure 1, one possible approach is to use the turning points $a_1, ..., a_{10}$ as the reference system as shown in Figure 1(c), and rewrite both trajectories with these points. Since trajectory $B$ has enough samples to describe its route, it is fairly simple to calibrate – just align each sample to its nearest turning point. However, there is so much information

lost in trajectory $A$ that simply aligning each sample to its nearest turning point (i.e., $a_1, a_8, a_9$) still results in a low quality trajectory. A good calibration system should help to infer that $a_7, a_3, a_4$ are very likely (indicated by a confidence value) to be passed by the routes from $a_1 \rightarrow a_8$, and $a_8 \rightarrow a_9$. After both trajectories have been calibrated, they can become similar again.

To sum up, we make the following major contributions.

- We make a key observation that widely existing heterogeneity in trajectory data caused by different sampling strategy can harm the effectiveness of trajectory data analysis, thus calls for a calibration system to reduce or eliminate the impact of the sampling heterogeneity.
- We design an anchor-based calibration system in two phases: building a reference system and performing calibration. As a comprehensive solution, we present four types of anchor points which are suitable for building a stable reference system. On this basis, we propose two kinds of approaches, geometry-based and model-based, to effectively align and complement trajectories using the anchor points.
- We conduct extensive experiments based on large-scale real trajectory dataset, which empirically demonstrates that the calibration system can significantly improve the effectiveness of most popular similarity measures for heterogeneous trajectories.

The remainder of this paper is organized as follows. Section 2 introduces the preliminary concepts and overviews the calibration system. We discuss the reference systems and calibration methods in Section 3 and Section 4 respectively. The experimental observations are presented in Section 5, followed by a brief review of related work in Section 6. Section 7 concludes the paper and outlines some future work.

# 2. PROBLEM STATEMENT

In this section, we present some preliminary concepts and give an overview of the proposed calibration system. Table 2 summarizes the major notations used in the rest of the paper.

**Table 2: Summarize of notations**

| Notation | Definition |
|---|---|
| $T$ | a raw trajectory |
| $\overline{T}$ | a calibrated trajectory |
| $p$ | a sample point of a trajectory |
| $a$ | anchor point |
| $\mathcal{A}$ | the set of anchor points in a reference system |
| $T(a_i \rightarrow a_j)$ | a trajectory travelling from $a_i$ to $a_j$ |
| $d(a_i, a_j)$ | distance between anchor points $a_i$ and $a_j$ |
| $d(T_i, T_j)$ | distance between trajectories $T_i$ and $T_j$ |

## 2.1 Preliminary Concepts

DEFINITION 1 (ORIGINAL ROUTE). *An original route of a moving object is a continuous mapping from time domain to spatial coordinates (i.e., longitude and latitude), indicating the exact path travelled by the object.*

The original route does not exist in a practical database since no positioning technique can acquire location records continuously. Instead, only a set of samples from the original route can be obtained and stored in the database.

DEFINITION 2 (RAW TRAJECTORY). *A raw trajectory $T$ is a finite sequence of locations sampled from the original route of a moving object, i.e., $T = [p_1, p_2, ..., p_n]$.*

Please note that a time-stamp is usually associated with each location record, but we will only concentrate on the spatial feature of the trajectory in this paper. Simply speaking, the raw trajectory of a moving object is only *one possible sample* from its original route by using a specific *sampling strategy*. A sampling strategy is the mechanism based on which the object decides to report its location. Time-based sampling, distance-based sampling, turning-based sampling and prediction-based sampling are among the most widely used sampling strategies. Besides, the object can also adopt different sampling rates which is the frequency of reporting the location depending on the sampling strategy (e.g., every 500 meters or 30 seconds, etc). In the rest of the paper, we will use *trajectory* and *raw trajectory* interchangeably when the context is clear.

DEFINITION 3 (ANCHOR POINT). *An anchor point is a fixed spatial location in the space, which is stable and independent of the trajectory data source.*

An anchor point can either refer to a geographical object that physically exists such as a Point of Interest (POI), or can be virtually defined such as the centroid of a grid. Actually any kind of entities in space can serve as the anchor points as long as they are stable and not affected by the trajectory data input.

DEFINITION 4 (REFERENCE SYSTEM). *A reference system is a homogeneous set of anchor points.*

All the anchor points that form a reference system must belong to the same type. The homogeneity of anchor points in a reference system guarantees the consistency of the trajectory calibration process, i.e., all the trajectories are calibrated to the same type of anchor points. For example, all the POIs in a city can constitute a reference system, while a union set of POIs and road intersections cannot since they are not of the same type.

DEFINITION 5 (TRAJECTORY CALIBRATION). *Given a reference system with anchor point set $\mathcal{A}$, trajectory calibration for $T = [p_1, p_2, ..., p_n]$ is a process that transforms $T$ into another trajectory $\overline{T} = [a_1, a_2, ..., a_m]$ where $a_i \in \mathcal{A}$ $(1 < i < m)$. $\overline{T}$ is called the calibrated trajectory of $T$.*

We expect the new trajectory $\overline{T}$ after calibration to preserve the original route of $T$ as much as possible, which is critical to reduce the erroneous adjustments to the original route. Ideally, the trajectories that share the same original route will have the same calibrated trajectory no matter what sampling strategies they adopt. Therefore an evaluation criterion is how well the calibrated trajectory resembles the original route.

## 2.2 System Overview

Figure 2 shows the overview of the proposed calibration system, which basically comprises two parts: reference system generation and trajectory calibration. In this work, we define four types of anchor points for constructing a reference system, i.e., space-based, data-based, POI-based and feature-based anchor points. The reference system can be built offline since it is independent of data input. The calibration process is divided into two phases: *alignment* and *complement*. Generally, the alignment phase maps existing sample points of a trajectory to some anchor points. The complement phase inserts additional anchor points to make the trajectory more complete and similar to its original route, which is especially important for trajectories with low sampling rates. The calibration process can be either online or offline depending on the application requirement (e.g., an on-the-fly process or a batch process). We will detail the discussion for each part in the next two sections.

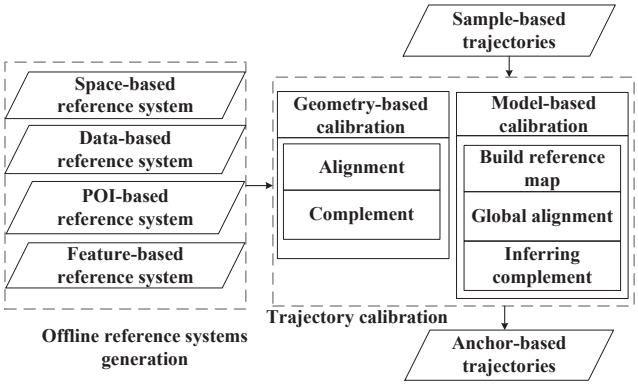



**Figure 2: System overview**

# 3. REFERENCE SYSTEMS

In this section, we will define several different types of anchor points for building a reference system. Although any fixed entity in the space can be an anchor point, not all of them are suitable for calibration. First, a reference system should have a sufficient number of anchor points in order to describe any given trajectory with high quality. For example, if we simply use all cinemas in a city as the reference system, a trajectory may have too few points after calibration. As such, most information in the raw trajectories will be lost. Second, a reference system should be stable and does not require substantial changes to calibrate new data. This property is crucial as it ensures that most new trajectories can be calibrated without refurbishing the reference system. Based on this guideline, we propose four types of anchor points that are expected to be suitable for building a reference system. We will study their effects on the calibration process with experiments later in this paper.

It is worth noting that road intersections and segments are natural choices for anchor points if a digital road map is available. But we will not adopt it in this work for two main reasons. First we attempt to make the proposed methods general enough to fit both constrained and unconstrained trajectories (e.g., traces of hiking, boating, walking, and many out-door activities); and second, most digital maps actually have legal or technical restrictions on their use [26, 14], which hold back people from using them in creating new applications. Therefore in this work, we will build reference systems based on resources that are easier to acquire, and our framework can be easily extended to accommodate the network-based anchor points in future work.

## 3.1 Space-based Anchors

The most straightforward idea is to divide the entire space into uniform grid cells, and use the *centroids* of the cells as the anchor points. An obvious advantage of using grid centroids is that we can easily build a reference system for any trajectory dataset without extra resources or information. The idea of using grid to adjust trajectories is inspired by the Realm method [13], but their purpose is to represent spatial objects in a database with predefined precision.

## 3.2 Data-based Anchors

A space-based reference system, despite its simplicity, may not capture the distribution of the trajectory data. In other words, the space partition may be too fine-grained for a set of sparsely distributed trajectories but too coarse for another with dense distribution. Another option is to select a large enough set of historical trajectories and use their sample points as the anchor points. Since these samples, called archived samples, represent the travel history of moving objects in the past, it is more reasonable to rewrite a given trajectory based on this type of anchor point. Besides, each anchor point is guaranteed to be a reachable location for a new trajectory. But using archived samples also has down sides. First, we must have a sufficient number of historical trajectories that locate in the same region with the input trajectories. Second, the calibrated trajectory may have a high degree of redundancy when the number of archived samples is large. For instance, in our experiments, we observe there can be more than 300 archived samples along a street one hundred meters long. Third, the effectiveness of the reference system may be affected by the noises residing in the archived data.

## 3.3 POI-based Anchors

A Point of Interest (POI) refers to a semantic location such as a restaurant, hotel, shopping centre, etc. POIs are stable in terms of their locations since a business or facility can usually last for a long period. Besides, POIs have a consistent distribution with trajectories in the same area, since in most cases people travel from/to some POIs to perform certain activities. Due to this property, we can use a POI dataset to build a reference system for the same area (e.g., within a county/city). However, we observe that directly using the POIs can be problematic. Since POIs can be densely distributed in a small area, each trajectory point can be rewritten with many possible candidate anchor points within close proximity. As such, the trajectories after calibration may still have quite different sample strategies. In order to remedy this problem, we pre-process the POI dataset by applying a density-based clustering method (e.g., DBSCAN [9]) to generate a smaller number of clusters, which will be used as the anchor points. By this means, POIs in densely populated areas can be merged into clusters and a cluster becomes an anchor point. As such, trajectories with similar routes, but different samples, will have a better chance to be re-synchronized by mapping their sample points to POI clusters.

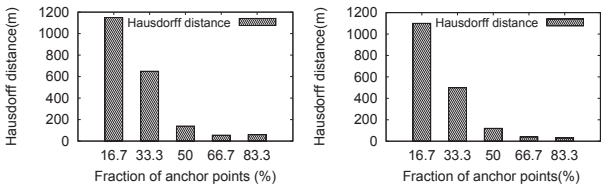
## 3.4 Feature-based Anchors

The data-based reference system utilizes the historical archived trajectory points as the anchor points, which can have a high degree of redundancy. To remedy this issue, we can use only some important points in trajectory data, called *features*, as the anchor points. Moving objects usually travel in a constrained space such as road networks, tracks or waterways. Therefore an important feature that can well characterize a trajectory is the *turning points*, at which a moving object changes its direction significantly. In other words, the main shape of a trajectory can be described by a few turning points regardless how many samples it has originally. So intuitively, if we can rewrite all the trajectories based on turning points, their shapes can be well preserved and the samples are also synchronized. We can adopt the algorithm in [6], which detects point clusters that satisfy both the density requirement and the direction change condition, to extract turning points.

## 3.5 Stability Test of Reference Systems

As we mentioned, reference system stability is a highly desired property, which is crucial to the performance of calibration. As such, in the last part of this section we conduct a quantitative analysis about the stability of the proposed reference systems. Since both grid centroids and POI clusters are fixed locations and thus inherently stable, we only evaluate the stability of the other two reference systems, i.e., the ones based on archived samples and turning points. We use a trajectory dataset that is collected from the taxicabs in a big city for a period of three months (more detailed description for this dataset will be provided in Section 5). This dataset is divided into six equal parts according to the log time, so

each part corresponds to about two weeks of the trajectory data. Then we use the first part to construct the reference systems. After that, we superimpose the second part on the first one and compute their Hausdorff distance [15] to measure the similarity between two sets of points. Next we merge the second and the first dataset into one, and progressively append the remaining parts in a similar manner. The Hausdorff distances between each appended part and the existing reference system are reported in Figure 3. We can see that for both methods the distances decrease quickly when the second (1/6) and the third part (2/6) of the dataset are appended to the reference systems, and then stabilize thereafter. It means the reference systems built upon the first half of the dataset have highly similar distribution with the data in the other half. In other words, only a part of the historical trajectory data can lead to a stable reference system, which will nicely capture the distribution of new data.




(a) Archived samples (b) Turning points

**Figure 3: Stability of reference systems**

# 4. TRAJECTORY CALIBRATION

In this section, we discuss in detail about the calibration process based on a reference system built offline. Specifically, the calibration process can be divided into two phases: *alignment* and *complement*. The alignment phase maps a trajectory to some anchor points. The number of sample points in a trajectory may be kept unchanged or reduced since multiple samples in close proximity can be merged into the same anchor point. However a low-sampling-rate trajectory cannot benefit from this phase alone since the calibrated trajectory will be still low-sampled. The complement phase inserts some anchor points in-between the trajectory points after alignment, by estimating those important but missing anchor points that the object may pass by.

## 4.1 Geometry-based Calibration

In this part we present a geometry-based calibration method, which simply explores the spatial relationship between trajectories and anchor points in space when choosing the anchor points for alignment and complement.

### *4.1.1 Alignment*

The geometry-based alignment is based on the simple idea of finding the nearest anchor point for each sample point of a given trajectory and then mapping the original sample point to its nearest anchor point. More precisely, each sample point in a trajectory will be aligned to a nearest anchor point. In order to avoid the case that a sample point will be aligned to a faraway anchor point, we can map a sample point to some anchor points within a distance threshold $\eta_{dist}$ and the sample points far away from any anchor point will be removed. Besides, if several consecutive sample points, i.e., $p_i, p_{i+1} \cdots, p_j$, of trajectory $T$ are all close with each other thus can be aligned to the same anchor point $a$, we will only record one copy of $a$ in the aligned trajectory. By this means, we can reduce the unnecessary redundant samples and outliers in some trajectories. The alignment process involves a constrained nearest neighbor search against the anchor point set for each trajectory point, which has a logarithmic-scale complexity with respect to the number of anchor points ($O(\log |\mathcal{A}|)$) [24] when some space partition or tree-based index is used. Thus the complexity of the alignment is $O(N_T \cdot \log |\mathcal{A}|)$, where $N_T$ is the size of the input trajectory.

### *4.1.2 Complement*

**Algorithm 1:** Geometry-based Complement

**Input**: Anchor point datset $\mathcal{A}$, aligned trajectory $\overline{T}$, $\eta_{dist}$
**Output**: Complemnted trajectory $\overline{T}$

1 $S_l \leftarrow$ line segments $l_i$ connecting consecutive anchor points $a_i$ and $a_{i+1}$ of $\overline{T}$;
2 **for** *each* $l_i \in S_l$ **do**
3  Initialize an empty list $\mathcal{L} \leftarrow \emptyset$;
4  Initialize a candidate complement anchor set $C_i \leftarrow \emptyset$;
5  $C_i \leftarrow$ all anchor points $a \in \mathcal{A}$ satisfying $d(a, l_i) \leq \eta_{dist}$;
6  $a' \leftarrow a_i$;
7  **while** *true* **do**
8  Find $a^* = \arg\min_{a \in C_i}\{d(a, a_i)\}$;
9  **if** *the angle between* $\overrightarrow{a', a^*}$ *and* $\overrightarrow{a_i, a_{i+1}} < \frac{\pi}{2}$ **then**
10  Insert $a^*$ to $\mathcal{L}$;
11  $a' \leftarrow a^*$;
12  Remove $a^*$ from $C_i$;
13  **if** $C_i$ *is empty* **then**
14  break ;
15  Insert the points in $\mathcal{L}$ into $\overline{T}$ in-between $a_i$ and $a_{i+1}$;
16 **return** $\overline{T}$

The main idea of the geometry-based complement method is to add the anchor points around the line segment in-between any two consecutive samples into the calibrated trajectory, based on the intuition that a moving object rarely changes its direction significantly between two consecutive sampled locations. The main difficulty of this method lies in that, after the anchor points nearby the line segments are selected, how to decide the right insertion order for these points. Algorithm 1 illustrates the main structure of our proposed method. Basically, given an aligned trajectory $\overline{T}$, the geometry-based method consists four steps. (1) Connect each two adjacent anchor points $a_i$ and $a_{i+1}$ by a line segment $l_i$ of $\overline{T}$ (line 1). (2) Build an anchor point set $C_i$ for each line segment that keeps all the anchored points $a$ whose distance to $l_i$ is less than a threshold $\eta_{dist}$ (line 4-5). $C_i$ holds all the candidate anchor points that are potential to be used. (3) Then we iteratively find the next anchor point $a^*$ from $C_i$ to be inserted which has the minimum distance to the $a_i$ (line 8), and insert $a^*$ in-between $a_i$ and $a_{i+1}$ if it does not change the moving trend of $l_i$ (line 9-10). (4) Repeat step 2 and 3 until $C_i$ becomes empty.

The example in Figure 4 demonstrates how the anchor points are selected and complemented into the trajectory segment in-between $a_i$ and $a_{i+1}$ by using the geometry-based complement algorithm. First we find five candidate anchor points ($a_1$, $a_2$, $a_5$, $a_7$ and $a_8$) whose distances with the line segment $l_i$ are less than $\eta_{dist}$. Then these points are sequentially connected in the order of their distances with $a_i$, and none of them conflicts with the major direction from $a_i$ to $a_{i+1}$.

The complexity of the above algorithm is dependent on two factors: the length of aligned trajectory $\overline{T}$ (i.e., the sum of the lengths of the line segments) and the number of anchor points "close" to $\overline{T}$. Let $L$ denote the average length of a trajectory segment and $\rho$ the average density of anchor points in the given reference system. Then the average number of anchor points that are close to each

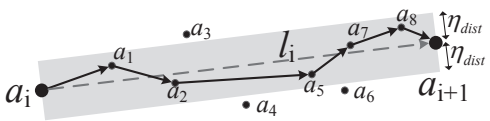



**Figure 4: An example of the geometry-based complement**

line segment is $N_a = 2L \cdot \rho \cdot \eta_{dist}$. Since these anchor points need to be sorted based on their distances with $a_i$, the overall complexity is $O(N_T \cdot N_a \log N_a)$, where $N_T$ is the average size of the aligned trajectory.

The geometry-based calibration has two major drawbacks. First, it takes a greedy strategy to align each trajectory point in an isolated manner, which ignores the relationship between anchor points. Second, the anchor points inserted in the complement step are all around the trajectory segments, which means it can only increase the sampling rate of the trajectory while keep the shape unchanged. But sometimes the shape of a trajectory has changed due to the loss of some descriptive samples (e.g., the one at turning points), in which case we need to complement the shape of the trajectory.

## 4.2 Model-based Calibration

To further improve the calibration performance, we propose a more advanced model-based calibration approach, which explores the correlations between anchor points that are learnt from a historical trajectory archive, and leverages the power of the Hidden Markov Model (HMM) and Bayesian inference to find the most probable alignment sequence and complement points, respectively. The model-based calibration consists of three steps: deriving anchor transition probability, global alignment and inferring complementary points. The first step learns from a historical trajectory dataset, the *transition probability* of an object moving from one anchor point to another. In the second step, we feed the anchor transition probability as well as the spatial relationship between sample points and anchor points into the HMM to derive the global optimal alignment. The third step also utilizes the anchor transition probability to infer the likelihood of one or multiple anchor points being passed through by the routes in-between two aligned anchor points, and then complement the trajectory by inserting those anchor points whose likelihoods are more than a certain threshold.

### 4.2.1 Anchor Transition Probability

In this part we will derive the transition probability between anchor points. First of all, a reference map, represented as a directed graph $G(V, E)$, is built to indicate the direct transition probability between two anchor points. Given a reference system and a historical trajectory dataset, we can construct the reference map in the following steps:

1. We add each anchor point in the reference system to the vertex set $V$ of the reference map.
2. We add a directed edge from $a_i$ to $a_j$, denoted by $e(a_i, a_j)$, if there exists a trajectory $T$ in the historical trajectory dataset travelling from $a_i$ to $a_j$ directly, i.e., two consecutive points $p_i$, $p_{i+1}$ of $T$ are in close vicinity of $a_i$ and $a_j$ respectively. We denote such a trajectory by $T(a_i \rightarrow a_j)$.
3. Each edge $e(a_i, a_j)$ is annotated with the number of $T(a_i \rightarrow a_j)$.

After the reference map has been constructed, we can immediately get the 1-step transition probability from $a_i$ to $a_j$ as follows, if $e(a_i, a_j)$ exists in the map:

$$\Pr_1(a_i \rightarrow a_j) = \frac{|T(a_i \rightarrow a_j)|}{|T(a_i \rightarrow *)|}$$

where $T(a_i \rightarrow *)$ represents the trajectories travelling from $a_i$ to any other anchor point. Figure 5 gives an example of the reference map, where the direction of arrow represents the transition relationship between two anchor points and the number around each arrow indicates how many trajectories travel through the two anchor points consecutively. Based on this reference map, we can derive the 1-step transition probability, e.g., $Pr_1(a_1 \rightarrow a_5) = \frac{5}{9}$.

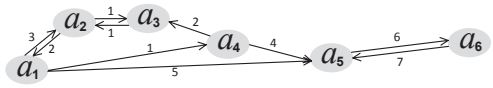



**Figure 5: An example of the reference map**

**$\lambda$-step transition probability.** The first order transition probability is not sufficient for inferring the relationship between anchor points without an edge. To address this issue, we leverage the first order probability to get higher order transition probabilities. First, a transition matrix $M$ with $m_{i,j} = \Pr_1(a_i \rightarrow a_j)$ is defined. It is easy to get that $M^2$ contains all the second order transition probability, and entries $m_{i,j}$ (after normalization) in $M + M^2$ correspond to the 2-step transition probability, which is the likelihood of transition from $a_i$ to $a_j$ within two steps. Analogously, we can get the $\lambda$-step transition probability by evaluating the matrix $M^{1:\lambda} = M + M^2 + ... + M^\lambda$. But it is not efficient to evaluate $M^{1:\lambda}$ during the calibration process since multiplication of large matrix is very expensive. In this paper, we pre-compute the transition matrix offline, by setting $\lambda$ to be sufficiently large to cover the most pairs of anchor points within the average distance of trajectory sample points.

**Background transition probability.** Sometimes due to the sparsity of historical data, it cannot reflect all the transition relationships between two anchor points even though they are close to each other. To get a complete reference map, we define a background transition matrix $B$ by considering the spatial proximity between two anchor points. Each entry $b_{i,j}$ of $B$, which represents the background transition probability from $a_i$ to $a_j$, is defined as $e^{-d(a_i, a_j)}$.

Finally we define the transition probability from $a_i$ to $a_j$, denoted by $\Pr(a_j|a_i)$, to be the normalized sum of the $\lambda$-step transition probability and the background transition probability, i.e.,

$$\Pr(a_j|a_i) = \frac{p_{i,j}}{p_{i,1} + p_{i,2} + ... + p_{i,|\mathcal{A}|}} \quad (1)$$

where $p_{i,j} = m_{i,j}^{1:\lambda} + b_{i,j}$.

### 4.2.2 Global Alignment

The geometry-based approach aligns each individual point in an isolated manner, which does not make use of the correlation between anchor points. Next we propose an HMM-based approach to find the most probable alignment by utilizing the transition probability in the derived reference map. In particular, the candidate anchor points are sequentially generated and evaluated on the basis of their likelihoods. When a new trajectory sample point is to be aligned, past hypotheses of the solution are extended to account for the new observation. Among all candidates in the last stage, the *surviving path* of anchor points with the highest joint probability is then selected as the final solution. In contrast to local alignment results in the geometry-based approach, the HMM-based approach

takes into account the anchor points in a collective manner when it generates the alignment.

Given a trajectory $T$, we treat each point $p_i \in T$ as an *observed state* and identify a set of candidate anchor points $A_i$, whose distance with $p_i$ is less than a certain threshold $\eta_{dist}$. Each of these candidate is regarded as a *hidden state* in the HMM. Each hidden state $a_j \in A_i$ has an *emission probability*, $\Pr(p_i|a_j)$, which is the likelihood of observing the point $p_i$ conditioned on $a_j$ being the ground truth. Intuitively, we would assign a higher probability to an anchor point if it is closer to $p_i$. In this paper, we assume $p_i$ follows a normal distribution with $a_j$ as the mean and a constant $\sigma$ as the variance, i.e., $\Pr(p_i|a_j) = N(a_j, \sigma^2)$. The *transition probability* between adjacent hidden states in the Markov chain, i.e., $\Pr(a_i|a_{i-1})$, can be obtained from the reference map that is derived earlier. Here we will adopt the first-order Markov chain based on the assumption of the 1-dependency, i.e., the probability of the current state is only dependent on the previous one, since the influence between two distant anchor points is usually very small.

Finally, we can compute the posterior probability of all hidden state variables given a sequence of observations, i.e., $\Pr(a_k|p_1, ..., p_n)\ \forall a_k \in A_k$,. It can be rewritten as

$$\Pr(a_k|p_1, \cdots, p_k, p_{k+1}, \cdots, p_n) \tag{2}$$

$$\propto \Pr(a_k|p_1, \cdots, p_k) \Pr(a_k|p_{k+1}, \cdots, p_n) \tag{3}$$

where $\Pr(a_k|p_1, \cdots, p_k)$ is the forward probability and $\Pr(a_k|p_{k+1}, \cdots, p_n)$ the backward probability. We can apply the forward-backward algorithm [3] to calculate the probability of each candidate anchor point and select the most probable alignment sequence. Since the forward-backward algorithm has the time complexity of $O(T \cdot N^2)$, where $T$ is the length of sequence and $N$ is the number of symbols in the state alphabet, the time complexity of the global alignment algorithm is $O(N_T \cdot |A|^2)$, where $N_T$ is size of raw trajectory and $|A|$ is the total number of candidate anchor point sets (i.e., $|A_1 \cup A_2 \cup ... \cup A_n|$).

### 4.2.3 Inferring Complementary Points

Next we discuss how to infer the possible anchor points to be inserted in-between two consecutive points $a_i$ and $a_{i+1}$ in an aligned trajectory $\overline{T} = [a_1, \cdots, a_i, a_{i+1}, \cdots, a_n]$. The main idea is to generate all possible paths between adjacent anchor points in the aligned trajectory, evaluate the probability of these paths and obtain the probability of each anchor point on any of these path. Afterwards, the anchor points with high confidence will be selected as the complementary points. The Algorithm 2 illustrates the main structure of this approach.

We denote the probability of an anchor point $a^*$ being passed by the original route of $\overline{T}$ from $a_i$ to $a_{i+1}$ by $\Pr_i(a^*|\overline{T})$. The aim of this step is to find the anchor points $a^*$ such that $\Pr_i(a^*|\overline{T})$ is greater than a pre-defined *confidence threshold* $\eta_{confi}$ (line 13-14).

$\Pr_i(a^*|\overline{T})$ is defined as follows (line 11-12):

$$\Pr_i(a^*|\overline{T}) = \sum_{P \in PP_\lambda(a_i, a_{i+1})} \Pr_i(P|\overline{T}) \cdot exist(P, a^*) \tag{4}$$

where $PP_\lambda(a_i, a_{i+1})$ is the set of possible paths which are constructed by using anchor points and connect $a_i$ and $a_{i+1}$ within $\lambda$ intermediate steps. $exist(P, a^*)$ is an indicator, whose value is equal to one if $a^*$ lies in the path $P$, and zero otherwise. In the sequel, we will discuss how to obtain $PP_\lambda(a_i, a_{i+1})$ and compute $\Pr_i(P|\overline{T})$, respectively.

**1). Generate the possible paths.**

In order to obtain $PP_\lambda(a_i, a_{i+1})$, we need to enumerate all the possible paths from $a_i$ to $a_{i+1}$ within $\lambda$ hops. Let $N(a_i)$ denote the anchor points in the reference map that are directly reachable from $a_i$ in the reference map. We build a *path tree* from $a_i$ to $a_{i+1}$ to help us find possible paths from $a_i$ to $a_{i+1}$ (line 3-4). A *path tree* from $a_i$ to $a_{i+1}$ is built according to the following four rules: (1) the root of the tree is $a_i$; (2) the height of the tree is $\lambda + 1$; (3) the child nodes of $a_j$ are $N(a_j)$; (4) $a_{i+1}$ must be the leaf node. An example of a path tree from $a_1$ to $a_6$ in illustrated by Figure 6(a). With the help of the path tree, finding all the paths $P$ of $PP_3(a_1, a_6)$ is simplified to visiting the tree from root $a_1$ to all the leaf nodes.

However, the above process can be very time consuming when $\lambda$ is large and/or each anchor point connects many other anchor points. In order to reduce the search space in the path tree, we can utilize the $\lambda$-step transition matrix $M^{1:\lambda}$ that is pre-computed with the reference map (line 2). Based on the $\lambda$-step transition matrix, it is easy to derive the set of destinations $S(a_i \rightarrow)$ that can be reached from $a_i$ within $\lambda$ steps. Similarly, we can also get the set of sources $S(\rightarrow a_{i+1})$ that can reach $a_{i+1}$ within $\lambda$ steps. The joint set $S(a_i \rightarrow a_{i+1}) = S(a_i \rightarrow) \cap S(\rightarrow a_{i+1})$ then contains all the anchor points on the paths from $a_i$ to $a_{i+1}$ within $\lambda$ steps. After that we can delete the nodes that does not exist in $S(a_i \rightarrow a_{i+1})$ and their child nodes in the path tree, by which means many impossible paths can be filtered out and the search space gets reduced substantially. Continuing with the previous example, from $M^{1:3}$ we know that $a_3$ can never reach $a_6$ within 3 steps, so $a_3$ and its child nodes can be deleted from the original path tree. The optimized path tree is shown in Figure 6(b).

**Algorithm 2:** Model-based Complement

**Input**: $\lambda$-step transition matrix $M^{1:\lambda}$, transition probability $\Pr(a_j|a_i)$, aligned trajectory $\overline{T}$, $\eta_{confi}$
**Output**: Complemented trajectory $\overline{T}$

1 **for** *each* $a_i \in \overline{T}$ **do**
2  $S(a_i \rightarrow a_{i+1}) \leftarrow$ candidate complementary points in-between $a_i$ and $a_{i+1}$ based on $M^{1:\lambda}$;
3  Generate the path tree from $a_i$ to $a_{i+1}$ by using $S(a_i \rightarrow a_{i+1})$;
4  $PP_\lambda(a_i, a_{i+1}) \leftarrow$ all paths from $a_i$ to $a_{i+1}$ in the path tree;
5  **for** *each* $P \in PP_\lambda(a_i, a_{i+1})$ **do**
6  Calculate $\Pr_i(P|\overline{T})$ by using transition probability $\Pr(a_j|a_i)$;
7  Initialize a list of complementary points $\mathcal{L} \leftarrow \emptyset$;
8  **for** *each* $a^* \in S(a_i \rightarrow a_{i+1})$ **do**
9  $\Pr_i(a^*|\overline{T}) \leftarrow 0$;
10  **for** *each* $P \in PP_\lambda(a_i, a_{i+1})$ **do**
11  **if** $a^* \in P$ **then**
12  $\Pr_i(a^*|\overline{T}) += \Pr_i(P|\overline{T})$;
13  **if** $\Pr_i(a^*|\overline{T}) \geq \eta_{confi}$ **then**
14  Add $a^*$ to $\mathcal{L}$;
15  Insert all the anchor points of $\mathcal{L}$ into $\overline{T}$ in-between $a_i$ and $a_{i+1}$;
16 **return** $\overline{T}$

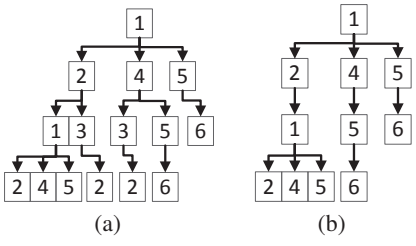



**Figure 6: An original path tree and its optimized path tree**

**2). Evaluate** $\Pr_i(P|\overline{T})$

Now we need to evaluate the probability of a path $P$ that connects $a_i$ and $a_{i+1}$ within $\lambda$ steps, conditioned on the observed alignment $\overline{T} = [a_1, \cdots, a_i, a_{i+1}, \cdots, a_n]$ (line 6), i.e.,

$$\Pr_i(P|\overline{T}) = \Pr_i(a_1^*, a_2^*, \cdots, a_k^*|a_1, \cdots, a_i, a_{i+1}, \cdots, a_n) \quad (5)$$

where $k \leq \lambda$ and $a_1^*, a_2^*, \cdots, a_k^*$ are the points on path $P$. The resulting trajectory will be $[a_1, \cdots, a_i, a_1^*, \cdots, a_k^*, a_{i+1}, \cdots, a_n]$.

However, the exact evaluation of Eq. (5) is too expensive to be feasible for calibration. To address this issue, we make the assumption that the probability of an anchor point is only affected by its precedent in a path. Then Equation 5 can be simplified as follows:

$$\begin{aligned} &\Pr_i(a_1^*, a_2^*, \cdots, a_k^*|a_1, \cdots, a_i, a_{i+1}, \cdots, a_n) \\ =&\Pr_i(a_1^*, \cdots, a_k^*|a_i, a_{i+1}) \\ =&\frac{\Pr(a_1^*|a_i)\Pr(a_2^*|a_1^*)\cdots\Pr(a_{i+1}|a_k^*)}{\Pr(a_{i+1}|a_i)} \end{aligned} \quad (6)$$

The time complexity of Algorithm 2 is $O(N_T \cdot |PP|^2)$, where $N_T$ is the size of the aligned trajectory and $|PP|$ is the average number of paths connecting two consecutive anchor points of $\overline{T}$ within $\lambda$ steps. Let $d_o$ denote the average out degree of an anchor point in the reference map, then $|PP|$ can be evaluated as $d_o^\lambda$. Usually the value of $\lambda$ is very small, which makes the practical time cost of Algorithm 2 reasonable. Besides, with the help of optimized path tree, only a small subset of the possible paths needs to be checked. The effect of this optimization will be verified in the experiment (Sec 5.3.7).

# 5. EXPERIMENT

In this section, we conduct extensive experiments to validate the effectiveness of our proposed calibration system, which entail different combinations of reference systems and calibration methods. All the algorithms in our system are implemented in Java and run on a computer with Intel Core i7-2600 CPU (3.40GHz) and 8 GB memory.

## 5.1 Experiment Setup

### 5.1.1 Data Preparation

**Trajectory Dataset:** We use a real trajectory dataset generated by 33,000+ taxis in a large city over three months. In total, this dataset has more than 100,000 trajectories. We define a trajectory as a high-sampling-rate trajectory if the average time interval between consecutive sample points is less than 10 seconds. According to this criterion, we select 11,028 high-sampling-rate trajectories from the dataset, and then divide them into two equal parts. One of them, called the training dataset, serves as an archived dataset which will be used for building a reference system, finding turning points and training the reference map. The other one, called the test dataset, is used for testing the effectiveness of calibration.

**Manipulated Trajectory Dataset:** We re-sample each trajectory $T$ in the test dataset to obtain three counterparts with varied sampling rates, i.e., a sample per 50, 100, 150 seconds, denoted as $T_{50}, T_{100}, T_{150}$. These trajectories refer to the same original route as the high sampled trajectory $T$ but have different sampling rates.

### 5.1.2 Anchor Point

**Grid Centroids:** We divide the area of the large city into 1570 by 1358 cells, each with a side of 100 meters, and get 2,132,060 grid centroids.

**Archived Samples:** We use the 1,485,284 sample points in the training dataset as the anchor points to build the data-based reference system.

**POI Clusters:** We purchase about 510,000 POI points of the same city from a reliable third-part company. Approximately 17,000 POI clusters are obtained using DBScan and the geometric centre of each cluster is used as the anchor point.

**Turning Points:** We extract about 32,000 potential locations of turning points from the training dataset, and finally generate 2,400 turning points with the method described in Section 3.

## 5.2 Evaluation Approach

### 5.2.1 Calibration Methods

We propose four types of anchor points and two calibration methods, which lead to eight combinations of calibration process. But we do not use the model-based calibration method with the grid centroids and archived samples, since their cardinalities are very large that renders the inference process not efficient. Therefore in the following experiments, we will apply the geometry-based approach to all types of anchor points, and the model-based approach on POI clusters and turning points only. All these calibration strategies and their abbreviations are listed in Table 3, in which SP stands for the method of using the raw trajectories without any calibration.

**Table 3: Calibration Methods**

| Anchor points | Calibration | | Method |
|---|---|---|---|
| | Geometry-based | Model-based | |
| sample points | N/A | | SP |
| grid centroids | √ | | GC |
| archived samples | √ | | AS |
| POI clusters | √ | | POI+G |
| POI clusters | | √ | POI+M |
| turning points | √ | | TP+G |
| turning points | | √ | TP+M |

### 5.2.2 Parameters

Table 4 lists all the parameters we used throughout the experiments, that all the parameters are assigned the default values unless specified explicitly.

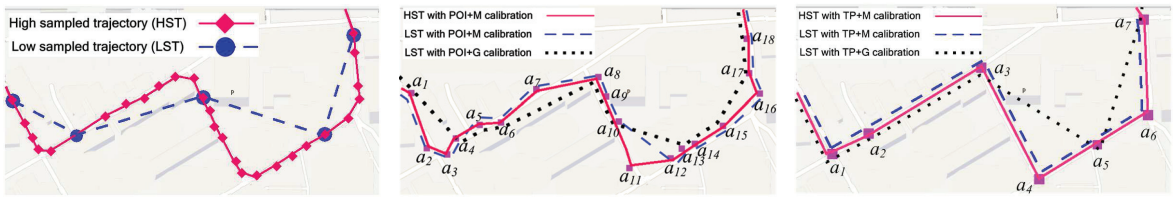
**Table 4: Parameter settings**

| Notation | Explanation | Default value |
|---|---|---|
| $\eta_{dist}$ | range of tolerance | 50m |
| $\eta_{confi}$ | confidence threshold of model-based complementing | 0.8 |
| $\sigma$ | standard deviation of the distribution of anchor points | 10m |
| $\lambda$ | maximum number of steps in transition matrix | 10 |

## 5.3 Performance Evaluation

### 5.3.1 Visualization of calibration effect

Before conducting the quantitative performance evaluation, we give an intuitive illustration for the calibration effect by visualizing the results. Figure 7(a) shows two trajectories with different sampling rates but referring to the same route. It can be imagined that conducting similarity analysis on them directly will result in a poor quality answer. Figure 7(b) illustrates their calibration result by using POI-based anchor points (represented by solid squares). For the high sampled trajectory, geometry-based and model-based approaches produce the same result (only POI+M calibration result is shown for the sake of conciseness). But they make difference on

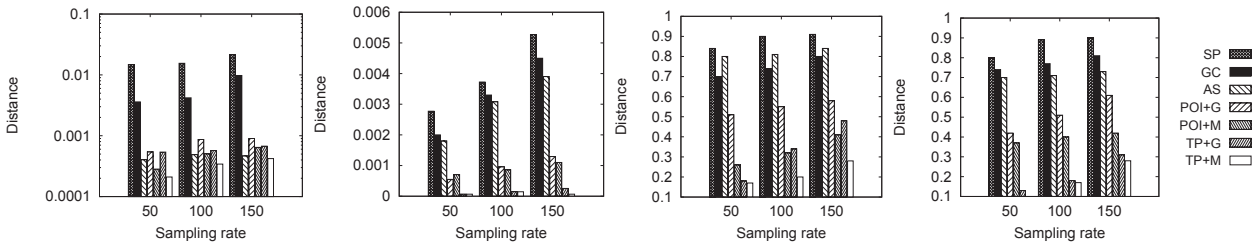



(a) High and low sampled trajectories from the same route

(b) Calibration with POI-based anchors

(c) Calibration with turning point based anchors

**Figure 7: Visualization of Calibration Effect**




(a) Euclidean Distance

(b) DTW

(c) LCSS

(d) EDR

**Figure 8: Distance between the trajectories referring to the same original route**

how to choose the complementary anchor points for the low sampled trajectory. Specifically, geometry-based approach can only complement the anchor points that are spatially "around" the trajectory segments (e.g., $a_6, a_8, a_{10}, a_{14}, a_{17}$), whereas the model-based method can choose more anchor points that are actually on the original route, and thus gain better calibration result (i.e., the blue dashed line is more similar with the red solid line). Figure 7(c) demonstrates the calibration effects by using turning points as the reference system. We can see that turning points can give more precise and concise representation for both high sampled and low sampled trajectories. Besides, the advantage of the model-based method is more obvious as we can see it fully recovers the original route for the low sampled trajectory.

### 5.3.2 *Effect on Similarity Measures: Self Comparison*

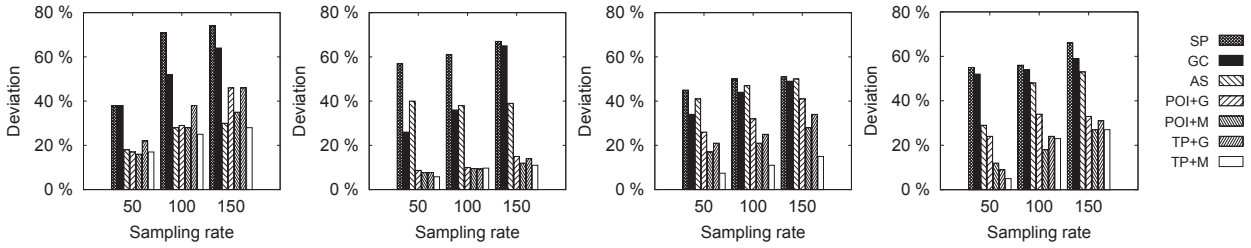
In the first set of experiments, we evaluate how the calibration methods can improve the effectiveness of trajectory similarity measures. For each trajectory $T$ of the test dataset, we use Euclidean distance (ED), DTW, LCSS and EDR to calculate the distances between $T$ and its three low-sampling-rate counterparts, i.e., $d(T, T_{50})$, $d(T, T_{100})$ and $d(T, T_{150})$. Analogously, we use these four measures to calculate the distances between $\overline{T}$ and their calibrated low-sampling-rate counterparts, i.e., $d(\overline{T}, \overline{T}_{50})$, $d(\overline{T}, \overline{T}_{100})$ and $d(\overline{T}, \overline{T}_{100})$. Since each pair of trajectories in comparison refers to the same original route, a smaller distance value means better effectiveness of the similarity measure. Figure 8 shows the results of the normalized distances (i.e., distance over the size of trajectory) based on the raw trajectories (denoted by *SP*) and trajectories with different calibration schemes. Not surprisingly, all distances gradually increase with the drop of sampling rate since more sample points characterizing the major shapes of trajectories are lost. However, raw trajectories have considerably greater distances than the calibrated trajectories do at all sampling rates, which demonstrates the ability of the proposed calibration methods to improve the accuracy of the common similarity measures. A general phenomenon from this figure is that, the POI and TP based methods achieve better effectiveness since the corresponding distance values are very close to the ground truth (zero), especially for ED and DTW distance. Besides, by learning the knowledge hidden in the historical data, model-based approaches (i.e., POI+M, TP+M) lead to even better performance compared to the geometry-based approaches (i.e., POI+G, TP+G). Consequently, the combination of turning points as the reference system and model-based calibration procedure (i.e., TP+M) turns to be the most robust approach in terms of the capability of recognizing the trajectories of the same route, as we can see that the distance based on it is always the smallest among all the methods.

### 5.3.3 *Effect on Similarity Measures: Cross Comparison*

A good calibration method should not only improve the ability to recognize the trajectory variants of the same route, but can also preserve the distance between any trajectories regardless of their sampling strategies. In this experiment we randomly select 5,000 trajectory pairs from the test dataset, and for each pair $(T_A, T_B)$ we use the four distance measures to calculate the distances between them, denoted as $d(T_A, T_B)$. $d(T_A, T_B)$ is regarded as the ground truth of the distance between the routes of $T_A$ and $T_B$. Then we calculate the distances between $T_A$ and different variants of $T_B$, i.e., $d(T_A, T_{B50})$, $d(T_A, T_{B100})$ and $d(T_A, T_{B150})$. Finally, we put $T_A$, $T_B$ and its variants through the calibration system, and re-calculate the distances between them, i.e., $d(\overline{T}_A, \overline{T}_{B50})$, $d(\overline{T}_A, \overline{T}_{B100})$ and $d(\overline{T}_A, \overline{T}_{B150})$. In order to illustrate how well these distances resemble their ground truth in a more intuitive way, we show in the results the *distance deviation (dev)* calculated by the following equation instead of the original distances:

$$dev(V(T_A), V(T_B)) = \frac{|d(V(T_A), V(T_B) - d(T_A, T_B)|}{d(T_A, T_B)}$$

where $V(T)$ denotes the variance of $T$ (e.g., with different sampling rates and/or calibration). The results are shown in Figure 9, where smaller deviation means that the evaluated distance is closer to the ground truth. As we can see that most average deviations of the raw trajectories are over 50%, and increase quickly with the drop of sampling rate. To the contrary, all the distance devi-




(a) Euclidean Distance (b) DTW (c) LCSS (d) EDR

**Figure 9: Distance deviation of calibrated trajectories from the ground truth**

ations between calibrated trajectories are smaller compared to the raw trajectories, which demonstrates the usefulness of our calibration methods in preserving the distances when the sampling strategies vary. Consistent with the previous experiment, POI and TP based approaches obtain much better performance as their $dev$ are all below 0.3, and even less than 0.1 for DTW distance. Again, the model-based approaches outperform the geometry-based approaches for all distance measures, and TP+M approach achieve the best calibration results for most distance measures.

### 5.3.4 Resynchronization Capability

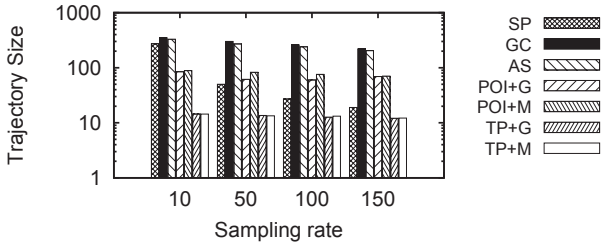
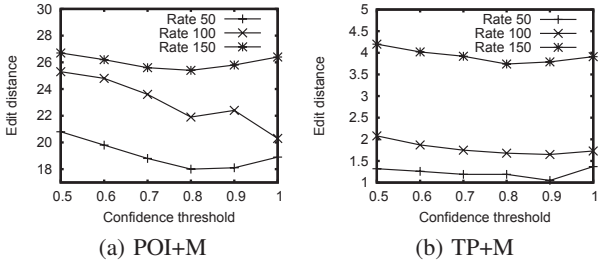



**Figure 10: Evaluation of resynchronization capability**

In this set of experiments, we evaluate the resynchronization capability of our calibration system. Intuitively, an effective calibration system should transform a specific trajectory into the one with similar sampling rate regardless of its original sampling rate. Thus for each trajectory in the test dataset, we calibrate its low-sampling-rate counterparts and obtain the calibrated trajectories, i.e., $\overline{T}_{50}$, $\overline{T}_{100}$,$\overline{T}_{150}$, and then compare the size between $T$ ($\overline{T}$) and $T_{50}$ ($\overline{T}_{50}$), $T_{100}$ ($\overline{T}_{100}$), $T_{150}$ ($\overline{T}_{150}$). Figure 10 shows how the average sizes of the raw trajectories and the calibrated trajectories change with the sampling rate (10s, 50s, 100s and 150s). As we can see from this figure, the sizes of the raw trajectories decrease significantly with the drop of sampling rate. To the contrary, the average sizes of calibrated trajectories much more stable with the variation of sampling rates, which verifies our expectation that the reference systems are effective in resynchronizing all the trajectories with more unified sampling rates.

### 5.3.5 Effect of Confidence Threshold

Next we test how the confidence threshold $\eta_{confi}$ used in the model-based approach (i.e., POI+M and TP+M) affects the calibration performance. Recall that a higher $\eta_{confi}$ results in fewer but more accurate anchor points inserted into the calibrated trajectory. In order to work out a good trade-off between the completeness and correctness of the complementary points, we tune the confidence threshold $\eta_{confi}$ from 0.5 to 1 with the step of 0.1. Meanwhile we calculate the edit distance between the calibrated trajectories and their low-sampling-rate counterparts using the POI+M and TP+M methods with different $\eta_{confi}$. As shown in Figure 11, generally all the distance values decrease when the confidence threshold rises, since a lot of incorrect insertions are avoided. However, when the threshold goes too high ($\geq 0.8$), the distances start to increase, which means the calibration effectiveness gets worse. The reason is that (almost) no anchor points can have high enough confidence to be complemented into the trajectories, thus leaving the low-sampling-rate trajectories largely incomplete. Based on the observations of this experiment, we recommend the threshold with the value between 0.8 and 0.9 to be appropriate.




(a) POI+M (b) TP+M

**Figure 11: Edit distance of high rate trajectories and low rate trajectories with different confidence**

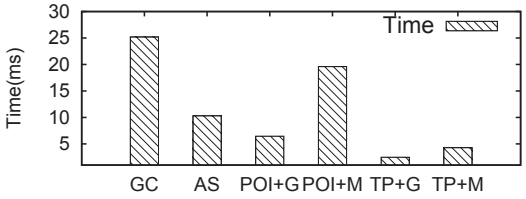



**Figure 12: Average time cost for calibrating one trajectory in millisecond**

### 5.3.6 Calibration Time Cost

We also evaluate the calibration time cost, which is especially important for online calibration systems. The average time cost for calibrating a single trajectory is shown in Figure 12, from which we observe that all the methods can calibrate a trajectory within tens of milliseconds. GC turns out to be the most inefficient approach, because the cardinality of grid centroids is too large, which increases the search space in geometry-based alignment and complement. Besides, the order of time costs for GC, AS, POI+G and TP+G is consistent with the number of anchor points in respective reference systems. This implies that the efficiency of calibration is heavily dependent on the cardinality of anchor points. The

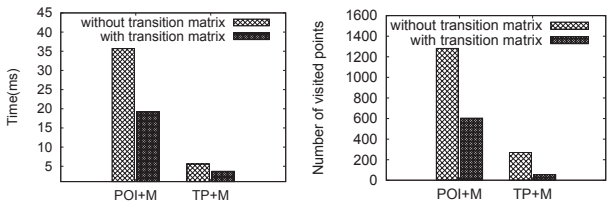



**Figure 13: Effect of transition matrix optimization**

geometry-based approach constantly runs faster than the model-based approach, since the model-based approach involves expensive inference on the reference map.

### 5.3.7 *Effect of Transition Matrix Optimization*

Recall that we have used the pre-computed transition matrix $M^{1:\lambda}$ to accelerate the model-based calibration in both alignment and complement phases. In this experiment, we evaluate the effect of this optimization by comparing the running time of the POI+M and TP+M approaches with and without using the transition matrix. Figure 13(a) and Figure 13(b) demonstrate the average time cost and the number of probed anchor points for calibrating a single trajectory. As expected the pre-computation in the transition matrix brings significant speed-up to both approaches.

### 5.3.8 *Effect on Similarity Queries*

The ultimate purpose of calibration is to improve the robustness and effectiveness of similarity-based analysis for trajectories. The last set of experiments is conducted to verify if the K-nearest neighbor search – the most important type of similarity query – can benefit from our proposals. We first randomly choose 500 query trajectories from the test dataset and find their 20 NNs, which are considered as the ground truth of this kNN query. Then we keep the query trajectories unchanged and transform all the trajectories in the test dataset by dropping sampling rates and applying calibration methods. Finally the same kNN queries are issued against the transformed datasets and the query precision is defined as the proportion of the correct results (the one existing in ground truth) in the new kNN result set. As shown in Figure 14, though the precisions based on all distance measures reduce with the decrease of sampling rate, calibration methods can improve the accuracy of kNN search results. This benefit is especially obvious for turning point based approaches as their precisions are constantly improved by around 20 percent compared to the ones without any calibration.

## 6. RELATED WORK

To our knowledge, there is no existing work on trajectory calibration or trajectory rewriting. In our model-based calibration approach, we leverage the historical trajectory data to find the best alignment and infer the possible intermediate anchor points to complement the trajectory, which share similar inspiration and techniques with the work on uncertainty management of trajectories and hot route discovery. Therefore in this section, we will firstly review these two lines of related work. As the goal of this work is to improve the effectiveness of similarity-based analysis, the distance measures studied in this paper are also reviewed at last.

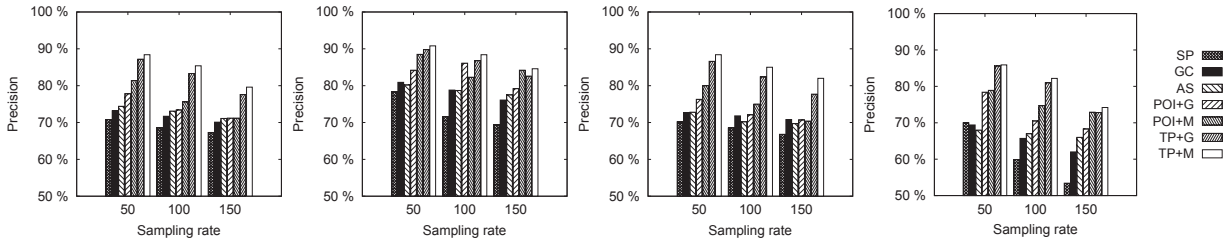
**Managing Uncertainty of Trajectories.** Several works have addressed the uncertainty issues of moving objects. [34, 33] proposed an information cost model which captures uncertainty and deviation in the moving objects updating problem. Proser et al. [27] models moving objects with a concept of spatial zones that define an object's whereabouts during two consecutive sampling positions as an ellipse under constraint maximum velocity. Trajcevski et al. proposes a three-dimensional cylinder to measure a new concept of *uncertain trajectory* in order to limit errors that could occur while capturing the movement of an object. Based on the model, a set of spatiotemporal operators and algorithms are proposed for continuous range queries [31] and nearest neighbour queries [30]. Cheng et al [7] proposes a new model, which shows that the location uncertainties are updated at every time instant and range queries are issued at a current time point. Zhang et al. [35] designs an integrated indexing structure for inferring the future location of uncertainty moving objects. An intuitive model for an uncertain trajectory is proposed in [36] to represent object movement along a road network, providing a unified probability distribution function (pdf) for the possible locations of a moving object at a given time-instant. [37] is the most relevant work in completing trajectories, but it applies its methods on the road network and does not give a concrete existing probability of a certain point.

**Hot Route Discovery.** [21] extracts hot routes by using a density-based algorithm *FlowScan* based on a concept called "traffic density-reachable". [29] investigates efficient ways to find and monitor hot motion paths that are defined as those visited by a certain number of moving objects. Nevertheless these two works are limited to mine frequently visited paths only. The focus of [23, 11, 12, 38] is on mining trajectory patterns to help find the popular routes from a start location to a destination. [11] proposes to mine a sequence of temporally annotated points called T-pattern, in order to find all T-patterns with sufficiently high support. Similarly, in [23, 12, 38], frequent paths or sequences are explored by existing sequential pattern mining algorithms. However, not every pair of start and end locations is able to match patterns given by these works. Chen et al. [6] evaluates the probability from a start point to the destination, but they only consider the forward probability from one place to another without considering the backward probability.

**Trajectory Distance Measures.** There are a large number of trajectory distance measures, among which Euclidean distance, DTW, LCSS and EDR are the most representative. DTW [19] is originally introduced for signal processing, which allows time-shifting in comparison. LCSS [18] is proposed based on the concept of the longest common subsequence, which is robust to noise by allowing skip of some sample points. EDR [5], which is based on edit distance, is also robust to noise and addresses some deficiencies in LCSS.

## 7. CONCLUSIONS

In this paper we have taken an important step towards effective calibration of trajectories with different sampling strategies to make them compatible when using many existing trajectory similarity measures. After studying the impact of trajectory heterogeneity on similarity measures, we have proposed a framework of trajectory calibration. We have examined four different types of anchor points which can be used to build stable reference systems. On top of that, two calibration approaches, the geometry-based approach and the model-based approach, are designed to align and complement trajectory data using the anchor points in the reference system. Extensive experiments have been conducted using a real trajectory dataset and a range of commonly used trajectory similarity measures. We have demonstrated that the calibration process can significantly improve the effectiveness of most popular trajectory similarity measures. The model-based calibration approach, which is based on using turning points to build the reference system, is shown to be particularly effective. This calibration process and its algorithms can be easily integrated with most existing works on




(a) Euclidean Distance (b) DTW (c) LCSS (d) EDR

**Figure 14: Effect of calibration methods on the accuracy of K-nearest neighbor search**

trajectory processing and mining, to reduce their reliance on high quality (densely sampled) trajectory data and to improve their similarity measure effectiveness. The ideas from this work open a new direction for future research, such as incorporating the temporal information in calibration, and novel indexing methods and query processing algorithms with calibrated trajectories based on the underlying reference system.

# 8. ACKNOWLEDGMENTS

This research is partially supported by the Australian Research Council (Grants DP110103423 and DP120102829), Natural Science Foundation of China (Grant No.61232006) and National 863 High-tech project (No: 2012AA011001).

# 9. REFERENCES

[1] Y. Cai and R. Ng. Indexing spatio-temporal trajectories with chebyshev polynomials. In *SIGMOD*, pages 599–610, 2004.

[2] V. Chakka, A. Everspaugh, and J. Patel. Indexing large trajectory data sets with seti. In *CIDR*, 2003.

[3] E. Charniak. *Statistical language learning*. MIT press, 1996.

[4] L. Chen and R. Ng. On the marriage of lp-norms and edit distance. In *PVLDB*, pages 792–803, 2004.

[5] L. Chen, M. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *SIGMOD*, pages 491–502, 2005.

[6] Z. Chen, H. Shen, and X. Zhou. Discovering popular routes from trajectories. In *ICDE*, pages 900–911, 2011.

[7] R. Cheng, D. Kalashnikov, and S. Prabhakar. Querying imprecise data in moving object environments. *TKDE*, 16(9):1112–1127, 2004.

[8] P. Cudre-Mauroux, E. Wu, and S. Madden. Trajstore: An adaptive storage system for very large trajectory data sets. In *ICDE*, pages 109–120, 2010.

[9] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231, 1996.

[10] E. Frentzos, K. Gratsias, N. Pelekis, and Y. Theodoridis. Nearest neighbor search on moving object trajectories. In *SSTD*, pages 328–345, 2005.

[11] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory pattern mining. In *KDD*, pages 330–339, 2007.

[12] H. Gonzalez, J. Han, X. Li, M. Myslinska, and J. Sondag. Adaptive fastest path computation on a road network: A traffic mining approach. In *PVLDB*, pages 794–805, 2007.

[13] R. Güting and M. Schneider. Realm-based spatial data types: the rose algebra. *The VLDB Journal*, 4(2):243–286, 1995.

[14] M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *Pervasive Computing, IEEE*, 7(4):12–18, 2008.

[15] J. Henrikson. Completeness and total boundedness of the hausdorff metric. *MIT Undergraduate Journal of Mathematics*, 1:69–79, 1999.

[16] H. Jeung, H. Shen, and X. Zhou. Convoy queries in spatio-temporal databases. In *ICDE*, pages 1457–1459, 2008.

[17] H. Jeung, M. Yiu, X. Zhou, C. Jensen, and H. Shen. Discovery of convoys in trajectory databases. In *PVLDB*, volume 1, pages 1068–1080. VLDB Endowment, 2008.

[18] J. Kearney and S. Hansen. Stream editing for animation. Technical report, DTIC Document, 1990.

[19] J. Kruskal. An overview of sequence comparison: Time warps, string edits, and macromolecules. *SIAM review*, pages 201–237, 1983.

[20] J. Lee, J. Han, and K. Whang. Trajectory clustering: a partition-and-group framework. In *SIGMOD*, page 604, 2007.

[21] X. Li, J. Han, J. Lee, and H. Gonzalez. Traffic density-based discovery of hot routes in road networks. *Advances in Spatial and Temporal Databases*, pages 441–459, 2007.

[22] Z. Li, B. Ding, J. Han, and R. Kays. Swarm: Mining relaxed temporal moving object clusters. In *PVLDB*, volume 3, pages 723–734, 2010.

[23] N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D. Cheung. Mining, indexing, and querying historical spatiotemporal data. In *KDD*, pages 236–245, 2004.

[24] A. Moore. An introductory tutorial on kd trees. Efficient memory-based learning for robot control. PhD Thesis, Carnegie Mellon University, 1991.

[25] J. Ni and C. Ravishankar. Indexing spatio-temporal trajectories with efficient polynomial approximations. *TKDE*, 19(5):663–678, 2007.

[26] OpenStreetMap. http://www.openstreetmap.org/.

[27] D. Pfoser and C. Jensen. Capturing the uncertainty of moving-object representations. In *Advances in Spatial Databases*, pages 111–131. Springer, 1999.

[28] D. Pfoser, C. Jensen, and Y. Theodoridis. Novel approaches to the indexing of moving object trajectories. In *VLDB*, pages 395–406, 2000.

[29] D. Sacharidis, K. Patroumpas, M. Terrovitis, V. Kantere, M. Potamias, K. Mouratidis, and T. Sellis. On-line discovery of hot motion paths. In *EDBT*, pages 392–403, 2008.

[30] G. Trajcevski, R. Tamassia, H. Ding, P. Scheuermann, and I. Cruz. Continuous probabilistic nearest-neighbor queries for uncertain trajectories. In *EDBT*, pages 874–885, 2009.

[31] G. Trajcevski, O. Wolfson, K. Hinrichs, and S. Chamberlain. Managing uncertainty in moving objects databases. *ACM Transactions on Database Systems (TODS)*, 29(3):463–507, 2004.

[32] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *ICDE*, pages 673–684. IEEE, 2002.

[33] O. Wolfson, S. Chamberlain, S. Dao, L. Jiang, and G. Mendez. Cost and imprecision in modeling the position of moving objects. In *ICDE*, pages 588–596. IEEE, 1998.

[34] O. Wolfson, A. Sistla, S. Chamberlain, and Y. Yesha. Updating and querying databases that track mobile units. *Distributed and parallel databases*, 7(3):257–387, 1999.

[35] M. Zhang, S. Chen, C. S. Jensen, B. C. Ooi, and Z. Zhang. Effectively indexing uncertain moving objects for predictive queries. In *PVLDB*, pages 261–272, 2009.

[36] K. Zheng, G. Trajcevski, X. Zhou, and P. Scheuermann. Probabilistic range queries for uncertain trajectories on road networks. In *EDBT*, pages 283–294, 2011.

[37] K. Zheng, Y. Zheng, X. Xie, and X. Zhou. Reducing uncertainty of low-sampling-rate trajectories. In *ICDE*, pages 1144–1155. IEEE, 2012.

[38] Y. Zheng, L. Zhang, X. Xie, and W. Ma. Mining interesting locations and travel sequences from gps trajectories. In *WWW*, pages 791–800, 2009.