A Novel Step-Invariant Representation for Trajectory Learning

Qingzhe Li
George Mason University
Fairfax, Virginia, USA
qli10@gmu.edu

Jessica Lin
George Mason University
Fairfax, Virginia, USA
jessica@gmu.edu

Liang Zhao
George Mason University
Fairfax, Virginia, USA
lzhao9@gmu.edu

Huzefa Rangwala
George Mason University
Fairfax, Virginia, USA
rangwala@cs.gmu.edu

ABSTRACT

Trajectory data are usually recorded as a sequence of sampled data points. For most trajectory data collected from human carried GPS devices (e.g, Geolife dataset [32]), the data are collected in a constant sampling rate. Since people travel using different transportation modes, the variance of velocities is large, which causes the sample points of the trajectory to be very sparse in some parts but highly dense in some others. This phenomenon seriously challenges the existing trajectory distance measurements in the following two aspects. First, for sparse parts of the trajectories, although several trajectory distance measures have been developed, it is extremely challenging for the existing trajectory measures to work well when the data points are sparse (e.g, GPS data collected on highway) due to the absences of matched points pairs between two trajectories. Second, for highly dense parts of the trajectories, it is not scalable to large dataset even using the simplest Euclidean distance measure to compute trajectory distances.

In order to address the above challenges simultaneously, we propose a Step-Invariant Trajectory (SIT) representation with linear time translation from raw data to uniformly distributed trajectory points by dynamically changing the sampling rates. Based on SIT representation, we also propose two effective and scalable distance measures for SIT. We evaluate the effectiveness and efficiency of our representation along with its distance measures by performing multiple trajectory classification and clustering experiments. These results show that our distance measures on SIT representation is much more accurate and robust than other distance measures and representations on sparse trajectory datasets. Our approach can also achieve competitive accuracy with the state of the art model-based trajectory representations on dense datasets, but the time spends on translating to our representation are 115 times faster, on average, than translating to other model-based representations.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ACM SIGSPATIAL 2017, Nov.2017, Redondo Beach,CA,USA
© 2017 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference format:
Qingzhe Li, Jessica Lin, Liang Zhao, and Huzefa Rangwala. 2017. A Novel Step-Invariant Representation for Trajectory Learning. In *Proceedings of International Conference on Advances in Geographic Information Systems, Redondo Beach,CA,USA, Nov.2017 (ACM SIGSPATIAL 2017),* 11 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

A large amount of trajectory data have been generated and studied in multiple areas. For instance, in Geographic Information System (GIS), trajectory data are collected from positioning devices such as GPS [20, 32]. In computer vision, trajectory data are obtained from various sensors or surveillance systems [5, 16]. To study and mine trajectory data, one of the most fundamental problems is how to measure the dissimilarity/distance between two trajectories. Unfortunately, there is no universal way to measure the distance. Even though several distance measures were developed for trajectory data, after a set of experiments, Wang et al. [28] concluded that "there is no trajectory similarity measure that can beat all the others in every circumstance."

We motivate our approach by introducing a simple example. Consider two spatial trajectories $T1$ and $T2$ that move in the same straight route represented in 2D coordinate system.

- $T1 = \{(0, 1), (3, 1), (7, 1), (9, 1), (10, 1)\}$
- $T2 = \{(0, 1), (2, 1), (4, 1), (8, 1), (10, 1)\}$

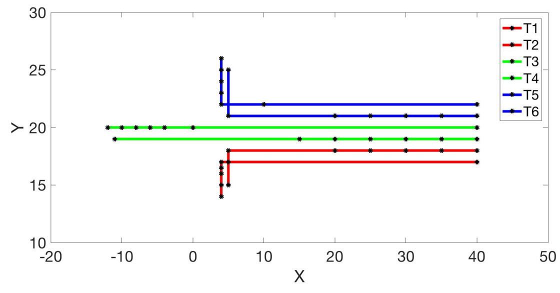
Intuitively, $Distance(T1, T2)$ should be zero since they are taking exact the same route. However, the distances computed by Euclidean Distance (EuDist), Dynamic Time Warping (DTW), and Discrete Frechét Distance (DFD)—all of which are the most commonly used trajectory distance measures—are shown in Table 1.

Distance Measure	Distance	Normalized Distance [1]
Euclidean Distance	3.3166	1.4832
DTW	4	0.6667
DFD	1	-

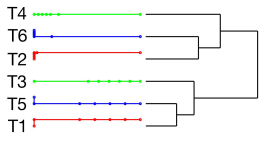
Table 1: The distances between two identical spatial trajectories with different sample points

[1]The normalized EuDist is computed by:

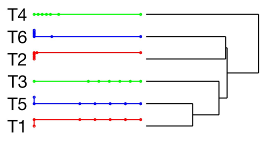
$$NormEuDist = \frac{EuDist(T1, T2)}{\sqrt{|T1|}} \quad (1)$$



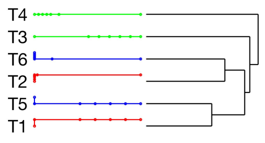
(a) The raw spatial trajectories to be clustered



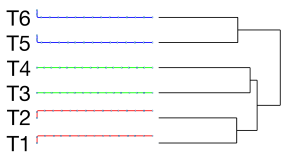
(b) Clustering by using Euclidean distance



(c) clustering by using Dynamic Time Warping distance



(d) Clustering by using Discrete Frechét distance



(e) Clustering by using Step-Invariant Euclidean distance

Figure 1: Single-link agglomerative clustering on various distance measures

We see that none of the three distance measures gives the expected distance for the identical spatial trajectories.

The distance measure is essential in trajectory learning. As another example, consider a set of spatial trajectories in Fig 1a. Intuitively, if we perform clustering on these trajectories, the result should group the trajectories with the same color together. However, Fig(1b, 1c, 1d) show the single-link agglomerative clustering result under Euclidean, Dynamic Time Warping and Discrete Frechét distance measures respectively. We can also see that none of these commonly used distance measures shows the expected result.

To solve the problem, researchers have developed different distance measures on trajectory data [4, 10, 17, 23, 27]. However, the distance is usually measured on the sampled data points of the actual trajectories. Therefore, the trajectory representation is a substantial problem that determines the quality of distance measure. Consider the trajectories in Figure 2. In order to compute the distance between two trajectories, all above trajectory distance measures need to compute the distances which connect the points between the two trajectories. We call such distance *basic distance*. The *basic distance* is composed of *true distance* and *alignment error*.

And the normalized DTW distance is computed by:

$$NormDTW = \frac{DTW(T1,T2)}{AlignmentCount} \tag{2}$$

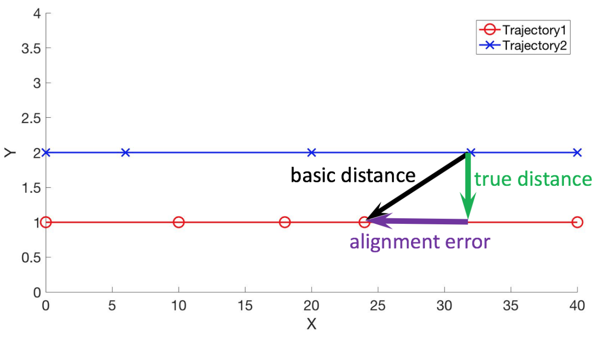


Figure 2: The relationship among *basic distance, true distance* and *alignment error*

The *true distance* is the radial distance from the point on one trajectory data samples to the "best match" point on the other trajectory. Notice that the "best match" point may not exist and not even close to any points on original trajectory samples. The *alignment error* is the distance from the "best match" points to the closest sample point on the same trajectory which is the noise of the *basic distance*. The relationship among the *basic distance*, the *true distance* and the *alignment error* is shown in Figure 2. The *true distance* is unknown since we do not know where is the "best match" point. The alignment error depends on the uniformity of the sampling points which is usually uncertain. When the sampling points are sparse, the *alignment error* would dominate the *basic distance*. Consequently the *basic distance* may not represent any relationship with the trajectory distance using any distance measure. So the trajectory distance measure problem can be reduced to the problem of finding the correct alignments between two trajectories. The alignment problem is also encountered in time series domain. Yankov et al. [30] use a constant rescaling factor combined with dynamic time warping to alleviate the alignment problem. However, it is not a good solution for trajectory data because moving objects usually have various velocities and missing values. There is no constant rescaling factor that can solve the alignment problem. In order to find a set of "good" alignments, we need a dynamic scaling factor to unify the sampling points so that the alignment error will no longer dominate the *basic distance*.

Based on the above observations, we propose a piecewise linear interpolation based representation that reduces the alignment error by dynamically resampling the trajectory data with equal distribution. That is, we dynamically add or remove points to translate the raw trajectories into trajectories with universal format such that the distance between two consecutive points in the same trajectories is equal to a constant number r. We call the translated trajectories the *Step-Invariant Trajectory (SIT)* representation of raw trajectory. Consider the two trajectories in Figure 2 again. If we set the constant step distance r such that $r <<$ *True Distance*, the *true distance* will dominate the *basic distance*. In the previous example with two identical spatial trajectories $T1$ and $T2$, when $r = 2$ their *SIT* representations would be:

- $T1' = \{(0,1),(2,1),(4,1),(6,1),(8,1),(10,1)\}$
- $T2' = \{(0,1),(2,1),(4,1),(6,1),(8,1),(10,1)\}$

The distance between $T1'$ and $T2'$ would be zero for all distance measures. Similarly, for the clustering task in Figure 1a, the result of agglomerative clustering using *Step-Invariant Trajectory* representation is shown in Figure 1e.

The rest of this paper is organized as follows: in Section 2 we give the literature review briefly in three related topics: trajectory distance measure, trajectory representation, trajectory reconstruction. Then, we introduce our *Step-Invariant Trajectory* representation along with its distance measures in Section 3. In Section 4, we evaluate our representation with its distance measures by applying 1-NN classification algorithm and agglomerative clustering algorithm. We compare the accuracy of our method with other distance measures on raw trajectory data and the state of the art representations to show the effectiveness and efficiency of our method. We conclude in Section 5.

2 RELATED WORK

We provide review of relevant literature on three related topics including trajectory distance measures, trajectory representation and trajectory reconstruction.

2.1 Trajectory Distance Measure

Many distance measures on trajectory data have been developed in recent years. Researchers have conducted effectiveness studies [15, 26, 28, 31] on popular trajectory distance measures (e.g., Euclidean Distance (EuDist)[23], Dynamic Time Warping (DTW)[17] , Discrete Freché Distance (DFD)[4], Longest Common Sub-Sequence (LCSS) [27] etc.) for raw trajectory data. These studies have a similar conclusion, that no single trajectory distance measure can beat other distance measures on every circumstance. Wang et. al[28] performed a series of transformations including increasing and decreasing the resampling rates, shifting the sample points and adding noise on the same set of real GPS trajectories to evaluate the effectiveness of various distance measures by comparing the computed distances (the closer to zero, the better the distance measure). In their conclusion, all the distance measures studied are sensitive to decreasing sampling rate. This observation is consistent with the case when the *alignment error* dominates the *basic distance.*

2.2 Trajectory Representation

Many researchers propose methods to represent the trajectory data using a constant number of coefficients obtained from different ways [6, 14, 18, 29]. Naftel et al. [18] learn trajectory data with Discrete Fourier Transform (DFT) coefficients. Jung et al. [6] used polynomial curve fitting algorithm to find a vector of suitable parameter sets to represent the trajectories. However, the constant dimensional parameter based representations may not properly represent trajectories with different route lengths since longer trajectories contain more information than shorter ones.

Some other researchers proposed segmentation-based trajectory representations along with some special distance measures. Porikli [22] proposed a set of trajectory distance metrics based on Hidden Markov Model based representation. However, these distance metrics rely on high quality trajectory data. Lee et al. [10] proposed a partition-and-group framework for trajectory mining tasks. Their proposed work break trajectory into a set of line segment using Minimum Description Length, then the authors define a three-component distance which integrates the perpendicular distance, the parallel distance, and the angle distance together. Finally they group the line segment together by using the three-component distance. In [8, 9, 13], the authors also use the three-component distance to perform different trajectory mining tasks. Their approach is approximate so the performance will degrade when the trajectory is complex (e.g., trajectory contains short detours). In [2, 19, 25], the authors discretize the trajectory using various methods, but all the proposed methods fail to handle the “boundary problem" (i.e. when points are near the boundary and discretized into different symbols). In [19], the authors use a fuzzy method which increases the computational complexity.

Morris et. al. [16] utilize the repetitive nature of the trajectories to build activity models in a 3-stage hierarchical clustering learning process, which characterizes the activities at multiple levels of resolution. Hu et. al. [5] build an incremental version of Dirichlet Process Mixture Model (DPMM) for trajectory clustering problem. Xu.H [29] proposed a shrinkage-based framework for unsupervised trajectory learning problem to improve the accuracy of clustering by using multi-kernal-based estimation process to iteratively leverage multiple structural information within a trajectory and the local motion patterns across multiple trajectories. In [14], Lin et al. use a 3-phase approach to represent the trajectory data. They first build a scene-specific thermal transfer field on trajectory training dataset. Then they build a 3D tube for each trajectory in the test dataset based on the thermal transfer field. Finally, for each trajectory, they generate a feature vector via a droplet process. The above model-based representations are able to achieve a higher accuracy then distance-based learning using raw trajectory data. However, these model-based representations require sufficient amount of trajectories in the dataset and high sampling rates. As a result, a significant amount of time is spent on building the model. These representations also require careful parameter tuning to achieve desired results. Therefore, these approaches are inconvenient to be applied on new datasets.

2.3 Trajectory Reconstruction

Some researchers propose methods to recover actual trajectories from a sparse trajectory dataset. In [11][24][12], the authors reconstruct the trajectories by using the information from other trajectories in the same dataset. A comparative survey [1] introduces several trajectory reconstruction methods based on various map inference algorithms. These methods are used for sparse trajectories, and do not handle both under-sampled case and over-sampled case simultaneously. While both the trajectory reconstruction approach and our Step-Invariant Trajectory (SIT) representation work by adding, removing and modifying the coordinates of the raw data, there are significant differences between our proposed method and existing approaches. The reconstruction methods focus on recovering the original trajectory from the trajectory data samples, while our method focuses on providing a good distance measure for trajectory data. On the other hand, for extremely sparse trajectory data, performing trajectory reconstruction before our Step-Invariant transformation is also helpful to achieve a better result.

3 METHODOLOGY

3.1 Terms and Notations

Def. 1 *Spatial-Temporal Trajectory (STT)*: A *Spatial-Temporal Trajectory* is a sequence of coordinates with timestamps (i.e, {latitude, longitude, timestamp}) obtained from an object moving in a 2-d or 3-d open space.

Def. 2 *Spatial Trajectory (ST)*: A single *Spatial Trajectory* $T = \{P_1, P_2, ..., P_m\}$ is a polyline which connects a sequence of points $P_i = (x_i, y_i)$ where (x_i, y_i) is the coordinate of P_i. We may use letter "T" followed by an integer trajectory id to represent single trajectory. We use letter "P" or "Q" with subscript to represent points (locations) in a trajectory. Unless stated, all the trajectories in the rest of this paper are *Spatial Trajectories.*

Def. 3 *Pointwise Euclidean distance*: We use $|P_i, P_j|$ to denote the *Euclidean distance* between two points P_i and P_j.

Def. 4 *Step-Invariant Trajectory*: A *Step-Invariant Trajectory (SIT)* is a *Spatial Trajectory* in which the Euclidean distance between every pair of two consecutive points is equal to a constant step distance r, where r is the only parameter chosen by user which controls the granularity of the *SIT.*

Def. 5 *Subtrajectory*: A subtrajectory $C^{[s,e]}$ is a subsequence of a trajectory $T = \{P_1, P_2, ..., P_m\}$,where s is the start index, e is the end index in T and $1 \leq s < e \leq m$. We use a sliding window of length n, where $n < m$ to extract all possible subtrajectories $C^{[s,s+n-1]}$ from T, where $1 \leq s \leq |T| - n + 1$.

Def. 6 *Trajectory length*: The length of a trajectory is the count of points in that trajectory. We use "$|T|$" to denote the length of trajectory.

Def. 7 *Trajectory route distance*: The *trajectory route distance* is the sum of the Euclidean distances between all consecutive pairs. We use "$||T|| = \sum_{i=1}^{|T|-1} |P_i, P_{i+1}|$" to denote the *trajectory route distance.*

Def. 8 *Spatial Trajectory dissimilarity/distance*: *Spatial Trajectory dissimilarity/distance* is the dissimilarity/distance between two trajectories under some trajectory distance measures.

3.2 Step-Invariant Trajectory Representation

In Section 1, we briefly introduced the advantages of *Step-Invariant Trajectory* representation when measuring the distance between two trajectories. In this section, we describe how to translate the raw trajectory into *Step-Invariant Trajectory* in linear time. By **Def. 4**, the "Step-Invariant" trajectory should have the same constant step distance between every pair of two consecutive points. In order to achieve the property of constant step distance, we use piecewise linear interpolation technique to force the step distance r equal to a user-chosen constant value which is depend on individual dataset. The user should chose a upper bound value of r such that r <<*true distance* and r is small enough so that the *Step-Invariant Trajectory* does not lose too many details of the original trajectory. Meanwhile, the values of r should not be too small in which will lead to unnecessarily high sampling rates and high computational complexity. However, we will demonstrate the value of r can be still chosen from a wide range of values in Section 4.4. We describe the related technique and implementation details below, and propose two trajectory distance measures for *Step-Invariant Trajectory* representation in Section 3.3.

3.2.1 *Dynamic Step-Invariant Trajectory Transformation by Piecewise Linear Interpolation.*

To formalize the piecewise linear interpolation technique, we give the equation below to show how to compute the coordinates. Consider a simple trajectory with only 3 sample points: $T1 = \{P_0, P_1, P_2\}$ in Figure 3a. We use piecewise linear interpolation to add some points $P_0^{(i)} (i = 1, 2, 3...)$ on line segment $\{P_0, P_1\}$ where $|P_0, P_0^{(i)}| = i * r$. So the coordinate $(x_0^{(i)}, y_0^{(i)})$ of $P_0^{(i)}$ can be computed by solving Equation 3 and Equation 4:

$$\frac{x_0^{(i)} - x_0}{x_1 - x_0} = \frac{i * r}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}} \tag{3}$$

$$\frac{y_0^{(i)} - y_0}{y_1 - y_0} = \frac{i * r}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}} \tag{4}$$

Given the constant step distance r, for any $i = 1, 2, 3...$, with the constrain of $i * r < |P_0, P_1|$, solving the above equation, $P_0^{(i)} = (x_0^{(i)}, y_0^{(i)})$ is determined as below:

$$x_0^{(i)} = \frac{i * r}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}} * (x_1 - x_0) + x_0 \tag{5}$$

$$y_0^{(i)} = \frac{i * r}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}} * (y_1 - y_0) + y_0 \tag{6}$$

Now we can translate the segment $\{P_0, P_1\}$ to $\{P_0, P_0^{(1)}, P_0^{(2)}, ..., P_0^{(k)}\}$, where k is the maximized value of i that satisfies the above constraint. In Figure 3b, we use solid red line to represent exactly one step, dashed red line to represent multiple steps.

3.2.2 *Handling the "tail problem".*

Since r is a constant distance of each step on Step-Invariant Trajectory, it is very common that there does not exist an integer i such that $i * r = |P_0, P_1|$. We call such problem "tail problem." Consider again the trajectory $T1 = \{P_0, P_1, P_2\}$ in Figure 3a. We currently have $k * r < |P_0, P_1| < (k + 1) * r$. To deal with the "tail problem", we use a substitution $P_1' = (x_1', y_1')$ of P_1 where P_1' is located on segment $\{P_1, P_2\}$ and $|P_0^{(k)}, P_1'| = r$. So we can get the coordinate of $P_1' = (x, y)$ by solving the following equations:

Let the equation of line segment $\{P_1, P_2\}$ be:

$$y = ax + b \tag{7}$$

We have known the coordinates of $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, so we have:

$$y_1 = ax_1 + b \tag{8}$$

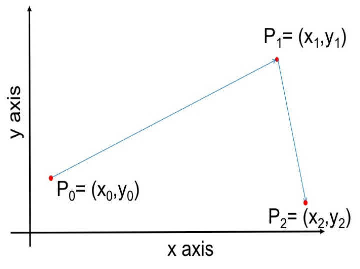
$$y_2 = ax_2 + b \tag{9}$$

Solve (8) and (9) together, we will have following result:

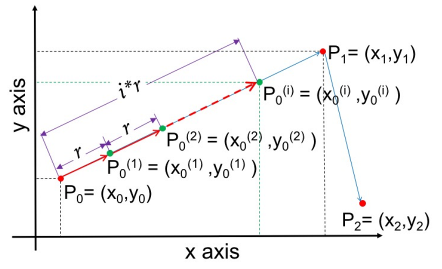
$$a = \frac{y_2 - y_1}{x_2 - x_1} \tag{10}$$

$$b = y_1 - \frac{y_2 - y_1}{x_2 - x_1} x_1 \tag{11}$$

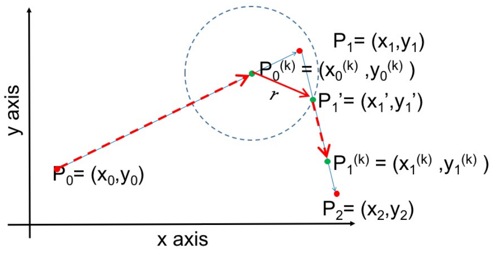
Since P_1' is on line segment $\{P_1, P_2\}$ and $|P_0^{(k)}, P_1'| = r$, we have:



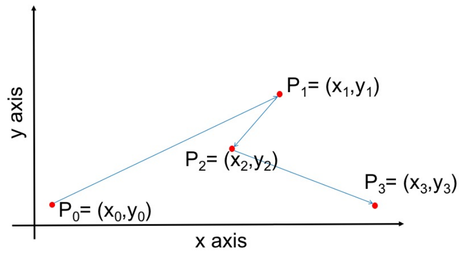
(a) Raw trajectory: T1 = $\{P_0, P_1, P_2\}$



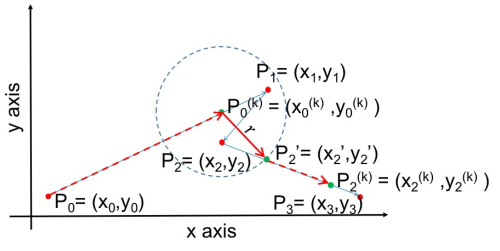
(b) Adding Points to $T1$ by using linear interpolation



(c) Handling "tail problem" points in $T1$ Case 1: $|P_0^{(k)}, P_2| \geq r$



(d) Raw trajectory: T2 = $\{P_0, P_1, P_2, P_3\}$



(e) Handling the tail points in $T2$ Case 2: $|P_0^{(k)}, P_2| < r$

Figure 3: Translating to Speed-Invariant Trajectory

$$y_1' = ax_1' + b \tag{12}$$

$$\sqrt{(x_0^{(k)} - x_1')^2 + (y_0^{(k)} - y_1')^2} = r \tag{13}$$

$$\text{subject to} \begin{cases} x_1' \in [min(x_1, x_2), max(x_1, x_2)] \\ y_1' \in [min(y_1, y_2), max(y_1, y_2)] \end{cases} \tag{14}$$

So we can compute the coordinate of P_1' by solving Equation (12) and Equation (13) with Constraint (14). When handling "tail problem", we may encounter two conditions. Consider trajectory $T1$ in Figure 3a,3b and 3c, in this case, $|P_0^{(k)}, P_2| \geq r$, Equation (12) and (13) will have exactly one set of solution satisfying Constraint (14) as the coordinate of P_1'. However, consider trajectory T2 in Figure 3d and 3d, in this case, $|P_0^{(k)}, P_2| < r$, the above equations do not have any solution satisfying Constraint (14). The second case is corresponding to the unnecessarily high sample rate points in raw dataset which partially accounts for high computational complexity and mismatched alignments when computing the distance between two trajectories. After interpolating a set of points, we reach the last point $P_0^{(k)}$ on line segement $\{P_0, P_1\}$. Then we could not find a point P_1' such that $|P_0^{(k)}, P_1'| = r$ and P_1' is on line segment $\{P_1, P_2\}$. We need to skip the line segment $\{P_1, P_2\}$ and compute the first point P_2' on next segment $\{P_2, P_3\}$ with $|P_0^{(k)}, P_2'| = r$. We can translate any raw trajectory to *Speed-Invariant Trajectory* by accordingly use the above techniques.

Finally, the Step-Invariant trajectory of $T1$ in Figure 3a will be:

$T1' = \{P_0, P_0^{(1)}, P_0^{(2)}, ..., P_0^{(k_0)}, P_1', P_1^{(1)}, P_1^{(2)}, ..., P_1^{(k_1)}\}$ as in Figure 3c and the Step-Invariant trajectory of $T2$ in Figure 3d will be:

$T2' = \{P_0, P_0^{(1)}, P_0^{(2)}, ..., P_0^{(k_0)}, P_2', P_2^{(1)}, P_2^{(2)}, ..., P_2^{(k_2)}\}$ as in Figure 3e.

3.3 Step-Invariant Trajectory Distance Measures

The *Step-Invariant Trajectory* provides a representation with uniform distributed trajectory points which can significantly reduce the alignment error, so it dramatically improves the quality of Euclidean distance between two equal length *Step-Invariant Trajectories*. To measure the distance between the Step-Invariant Trajectories with different lengths, we borrow a concept from time series subsequence matching [3] to define two best-match Euclidean distances on *Step-Invariant Trajectory*.

3.3.1 Best-Match Euclidean Distance (BMED). Given two Step-Invariant trajectories $T1 = \{P_1, P_2, ..., P_m\}$ and $T2 = \{Q_1, Q_2, ..., Q_n\}$, where $m < n$. As stated in **Def.5**, we use a sliding window with the length of m to extract a set of sub-trajectories $C_2^{[s,e]}$ from $T2$, then compute the Euclidean distances between $T1$ and $C_2^{[s,e]}$. The Best-Match Euclidean Distance is the minimal euclidean distances divided by the square root of m. The square root of m is used to normalize the distances computed from various lengths.

$$BMED(T1, T2) = \frac{\min(EuDist(T1, C_2^{[s,e]}))}{\sqrt{m}} \tag{15}$$

where C_2 is a subtrajectory of $T2$ with $1 \leq s \leq n - m + 1$and $e = s + m - 1$.

3.3.2 Penalized Best-Match Euclidean Distance (PBMED). For some applications, the expected distance measure should be able to reflect the differences between the two route distances. For example, for some classification/clustering problems, the route distance is also an important feature (e.g. Two similar trajectories with different route distances belong to two different classes/clusters). In this case, the BMED will be small and will not separate the two trajectories correctly. In this case, a Penalized Best-Match Euclidean Distance is proposed to solve the problem. For different applications and datasets, the user may define the penalty function accordingly. We use two intuitive penalty factors which are: (1) Alignment Penalty (AP) and (2) Route Distance Penalty Coefficient DPC.

$$PBMED(T1, T2) = \sqrt{\frac{\min(EuDist(T1, C_2^{[s,e]})^2) + AP}{n}} * DPC \quad (16)$$

where,

$$AP = UnmatchedHead + UnmatchedTail \quad (17)$$

$$UnmatchedHead = \sum_{i=1}^{s-1} |P_1, Q_i|^2 \quad (18)$$

$$UnmatchedTail = \sum_{i=e+1}^{n} |P_m, Q_i|^2 \quad (19)$$

$$DPC = \frac{n}{m} \quad (20)$$

The complexity of BMED and PBMED are both $O(m(n-m))$. For most trajectory mining tasks, under SIT representation, the differences between m and n are very small, so the computational complexity would be close to $O(m)$. The user can also modify the AP and DPC functions for different applications and/or datasets.

4 EVALUATION

4.1 Datasets

4.1.1 Synthetic Highway Datasets. This set of data simulate vehicles trajectories recorded by GPS devices moving along a 2,400 meters straight highway with four lanes on the same direction. The speed limit is set to 20 meters/second (about 45 miles/hour), so the vehicles should take about 2 minutes to pass through the highway segment. The Synthetic Highway Datasets include 7 datasets with various sampling rates of k records per minute, where $k \in \{4, 6, 12, 20, 30, 60, 120\}$ respectively. To simulate various instant velocities, for each trajectory, we uniformly generate $2k$ random numbers from 0 to 2400 as the values on X direction. The width of each lane is 3 meters. We also add Gaussian noise with $\sigma = 3$ to simulate the minor turns on the wheel or the errors of the GPS device. For each lane we generate 100 trajectories. The label information was given according to the corresponding lane as the ground truth. Finally, for each dataset there are 4 classes with 100 trajectories in each class. Figure 4 shows one of the dataset with $k = 12$. The colors on the heat map indicate the class labels.

4.1.2 Semi-Synthetic San Francisco Bay Area Taxi Dataset . To generate the semi-synthetic dataset from unlabelled dataset, we first randomly select 100 trajectories from the SFO Taxi datasets [21]. For each selected trajectory, we create 10 mutative trajectories by repeatedly performing the following transformations:

(1) adding the same number of points by random linear interpolation;
(2) removing the original points;
(3) removing some points by a random decreasing sampling rate G , where $0 < G < 50\%$;
(4) randomly shifting the remaining points so the trajectories belonging to the same class will not overlap with each other.

Step (1) and Step (2) are used for generating similar trajectories following similar routes but with non-identical coordinates. Step (3) and Step (4) are used in [28] to test the effectiveness of different distance measures. In [28], the result shows that none of the six trajectory similarity measures can survive from decreasing the sampling rate. So we create this semi-synthetic dataset showing in Figure 5 to challenge the limitations of most existing trajectory similarity measures.

4.1.3 CROSS Dataset [16]. The CROSS dataset, also used as the benchmark in [14, 16], simulates four-way traffic with 19 through and turn patterns. The dataset is divided into a training set containing 1900 normal trajectories with about 100 trajectories in each cluster, and a testing set containing 9500 normal trajectories with about 500 trajectories in each cluster. We use the whole dataset for classification as in [14, 16] and 1900 trajectories in training set for clustering as in [14, 29].

4.1.4 Vehicle Motion Trajectory (VMT) dataset [5]. This dataset, used in [5, 14, 29], contains 1500 trajectories belonging to 15 clusters unevenly. The trajectories in this dataset are noisy and oversampled.

4.2 Trajectory Classification

For classification problems, we use 1-NN classifier on our representation with either Best-Match Euclidean Distance (BMED) or Penalized Best-Match Euclidean Distance (PBMED) depending on the different classification objectives. For example, if we want to consider two trajectories following similar routes but having different *route distances* as belonging to the same class, the BMED is desired. Otherwise PBMED is desired.

We evaluate the effectiveness of our approach by computing a commonly used Classification Accuracy (CA). CA is computed as the ratio between the sum of the main diagonal element on confusion matrix and the size of the test dataset.

We use our Highway datasets and SFO dataset to compare the CA using our BMED or PBMED distance measure on the SIT representation, with the CA using DTW, LCSS, and DFD distance measures

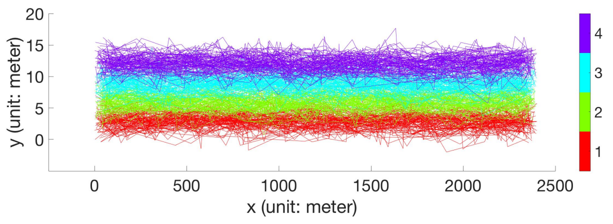


Figure 4: Synthetic Highway Dataset

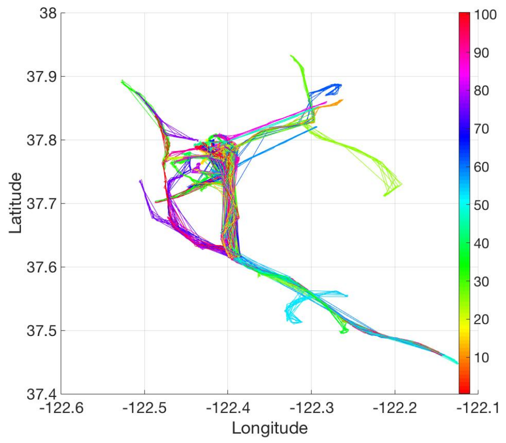


Figure 5: Semi-Synthetic San Francisco Bay Area Taxi Dataset

on raw data. In order to show the effectiveness of the SIT representation, we also add the results of BMED or PBMED on the raw data to the experiments.

4.2.1 Classification Results on Highway Datasets. For this datasets, we want to classify the trajectories on different lanes, so we use BMED as our distance measure. For each dataset in Highway data, we randomly select 10 trajectories from each class as the training set. We use the rest of trajectories as the testing set. We use Best-Match Euclidean Distance as our distance measure on Speed-Invariant representation with $r = 6$ to compare with four popular distance measures on raw representation. For LCSS and DTW, we choose the window size rate $w = 30\%$. For LCSS, we evaluate 20 different ε values and report the best classification accuracy. The results are shown in Figure 6. Our SIT representation achieves over 95% accuracy in all cases. All the other distance measures on raw trajectories do not work when the sampling rate is less than 20 per minute. As explained in Section 1, other distance measures will not perform well on raw trajectories when the alignment error dominates the *basic distance* LCSS works relatively well when the sampling rate is greater than 20 samples per minutes, but notice that LCSS distance is sensitive to the choice of parameter ε. For this dataset the best ε exists because the interval between the neighbor lanes is a constant number. However, for other datasets, such best constant ε may not exist, as we will show in the SFO dataset. DTW seems to only work when the sampling rate is greater than 120 samples per minute. However, the high sampling rates result in shorter battery life, which is fatal shortage on GPS devices.

4.2.2 Classification Results on SFO dataset. The settings of this experiment are the same as the Highway datasets except we use Penalized Best-Match Euclidean Distance (PBMED) for this dataset since a trajectory similar to some sub-trajectory of a longer trajectory is considered to belong to different classes. The results are shown in Figure 7. We use 100 to 500 trajectories for training and the rest of the 1000 trajectories in the dataset for testing. We can clearly see that our PBMED on the SIT representation is superior

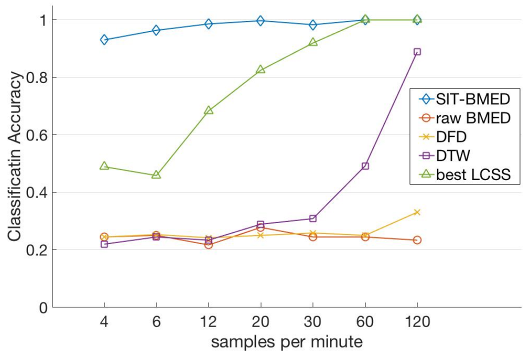


Figure 6: Classification Accuracy on Highway Dataset with Increasing Sampling Rate

than other measures on raw data. Moreover, even using a very small size training set, e.g., 1 trajectory for each class, our approach can still achieve about 90% accuracy. This is very important because in real world datasets, labelled data are very precious, and using less training data in 1-NN classifier needs less time for classification.

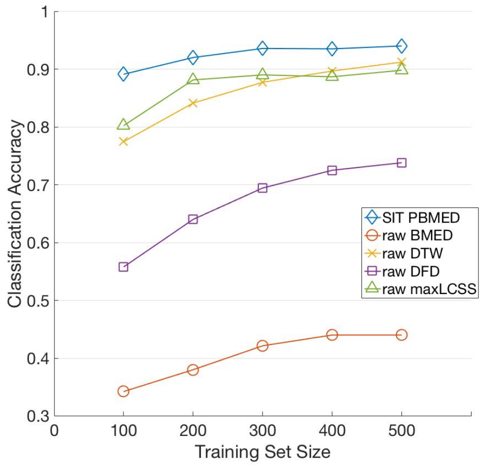


Figure 7: Classification Accuracy on SFO dataset with Increasing Training Size

4.2.3 Classification Results on CROSS Dataset [16]. This dataset has fixed training set and testing set, so we can directly compare our method with the original method: 3-Stage Hierarchical Learning [16], and the state of the art approach: Tube-and-Droplet based representation [14]. We also add the BMED and DTW distance measures using 1-NN classifier on raw data as the baseline method. The results are shown in Table 2. We can see that our method performs as good as the state of the art methods [14]. The Tube-Droplet method outperforms other methods except ours, however, it requires careful tuning of the parameters, and it takes a long time to build the model.

Method	Accuracy
Raw+BMED+1-NN	93.1
Raw+DTW+1-NN	95.6
3-Stage Hierarchical Learning Process[16]	96.8
tDPMM [5]	98.0
Tube-Droplet[14]	**98.6**
SIT+BMED+1-NN (Our Method)	**98.6**

Table 2: Classification Accuracy on CROSS[16] dataset

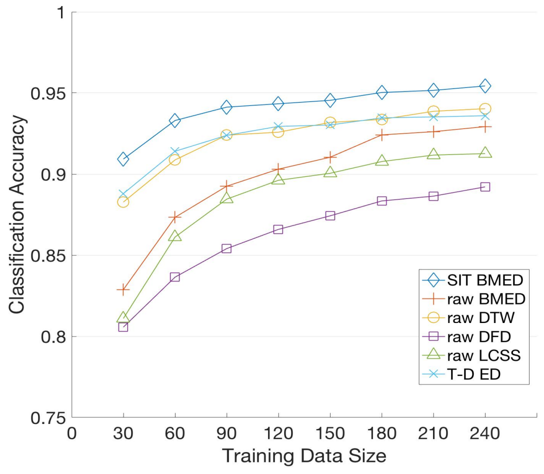


Figure 8: Classification Accuracy on VMT Dataset with Increasing Training Size

4.2.4 Classification Results on VMT dataset. We randomly select t trajectories from each class in original VMT dataset as the training set and use the remaining part as the testing set, where $t \in \{2, 4, 6, 8, 10, 12, 14, 16\}$. There are 15 classes, so the sizes of the training set are between 30 and 240. Then we classify the dataset using *BMED,DTW,maxLCSS,DFD* with raw data, Euclidean Distance with Tube-and-Droplet representation and BMED with our SIT representation. We repeat above experiments 30 times and record the average classification accuracy. The results are shown in Figure 8. We can see that even though there is only limited amount of labelled data available, our approach can still achieve above 90% accuracy.

4.3 Trajectory Clustering

We compare the effectiveness of our SIT representation with two mean-shift based methods: Mean Shift (MS) and Manifold Blurring Mean Shift (MBMS), and 2 state of the art representations, Adaptive Multi-Kernel-based Shrinkage(AMKS) [29] and Tube-and-Droplet [14].

In recent studies [5, 14, 29], the authors use the learning accuracy measure defined in [5] to evaluate the effectiveness of clustering results. The learning accuracy measure makes sense when the number of learned clusters is different from the number of clusters in

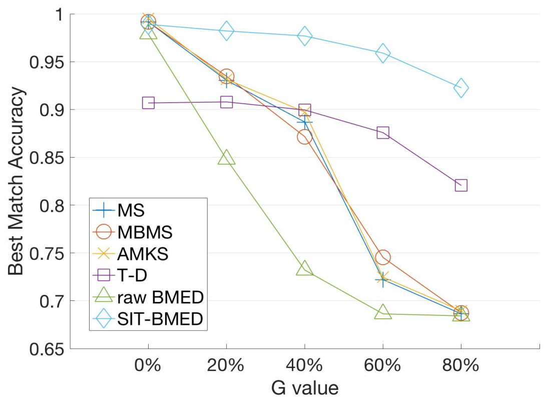


Figure 9: Best Match Accuracy on CROSS Dataset with Increasing G Value

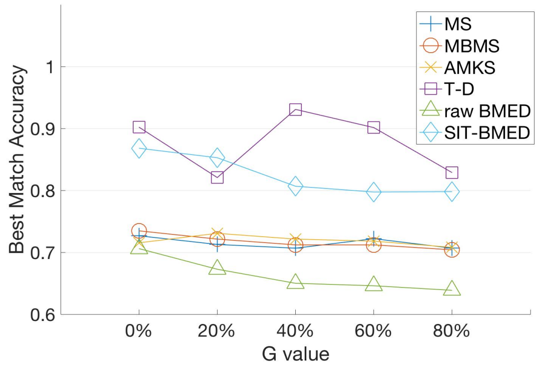


Figure 10: Best Match Accuracy on VMT Dataset with Increasing G values

the ground truth. For experiments where the number of clusters is given. The learning accuracy will not reflect the actual effectiveness of the clustering results because the definition of learning accuracy does not require one-to-one mapping. To compute the actual accuracy with one-to-one mapping, we first build a confusion matrix by randomly assigning the cluster Id to the learned clusters. Then we apply the Hungarian algorithm [7] and adjust the permutation of the learned cluster Ids to maximize the sum of the main diagonal on the confusion matrix. We call the optimal confusion matrix the *Best Match Confusion Matrix (BMCM)*. After that, we can define the *Best Match Accuracy* as:

$$BestMatchAccuracy = \frac{sum(diagonal(BMCM))}{totalnumberof trajectories} \tag{21}$$

Some previous work [14, 29] employing K-means clustering algorithm to compare different representations/distance measures. K-means algorithm usually has large variance of clustering accuracy with various initial centroids which may cause the inaccurate results when evaluate the effectiveness of various representations and/or distance measures. So we cluster the trajectories by using agglomerative clustering algorithm which is a stable algorithm. Then we cut the dendrogram into the desired number of clusters (based on the number of clusters in the ground truth) to evaluate the effectiveness of various representations and/or distance measures. We use six linkages [2] including single,complete,average, weighted average, centroid, Ward's method to cluster the datasets, and report the best accuracy over the six linkages.

In addition, according to Wang's conclusion [28], decreasing sampling has the most significant influence on the accuracy. We also test the robustness by randomly removing some points. We randomly remove G percent points from each trajectory in the dataset.

The result for the CROSS dataset show in Figure 9. We can see that the Tube-and-Droplet[14] and our method are less sensitive to the value of G. However, no matter how we adjust the parameters, the accuracy for the Tube-and-Droplet[14] method is still worse than our method.

Figure 10 shows the results for the VMT dataset. Our method and the Tube-and-Droplet method still perform better than other methods, but Tube-and-Droplet method needs a lot of effort on tuning various parameters. All methods are insensitive to the G values in this dataset, because as mentioned before, the VMT dataset is oversampling and noisy. Even if we remove 80% points (i.e. G = 80%) from each trajectory, there are still enough points (about 10 to 40 points) remaining to represent the trajectory. Furthermore, some noisy points may be removed.

4.4 Sensitivity Test on Parameter Settings and Running Time

Unlike other approaches which usually require careful tuning of three or more parameters to achieve good results, our approach only requires one parameter–the step distance r which decides the granularity of the trajectory. We tried ten different *Step Distances* with *Step Distance*$= i * minR$, where $i \in [1, 10]$ and $minR$ is the smallest step distance we choose. For Highway dataset, we choose $minR = 1$; for SFO dataset we choose $minR = 0.001$; for CROSS [16] we choose $minR = 10$ and VMT [5] we choose $minR = 3$. The results for classification accuracy and clustering accuracy with different *Step Distances* setting are shown in Figure 11 and 12 respectively. We can see that our SIT representation is insensitive to *Step Distance* settings. *Step Distance* can be chosen from a wide range values. The accuracy for clustering varies within about ±5% range when using different *Step Distance* since smaller *Step Distances* are able to keep more details of the raw trajectory, but it is also more sensitive to the noises in the raw data.

We use a Macbook Pro with i7 CPU and 16G memory to evaluate the efficiency of translating to our representation and computing the BMED distance. Table 3 shows the running time spent on translating to AMKS representation, T-D representation and our SIT representation respectively. It only takes less than 4 seconds to translate to our *Speed-Invariant Trajectory* representation. Our representation achieves 115 time faster on average than the model-based approaches [14, 29] on translating to the representations. We also evaluate the running time spent on computing the BMED

[2]we do not use the linkage of centroid and Ward's method for raw and SIT representations because they are not matrix-based representations

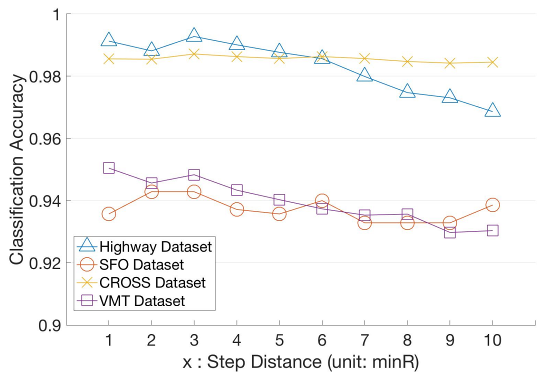


Figure 11: Classification Accuracy with Different Step Distance $x * r$

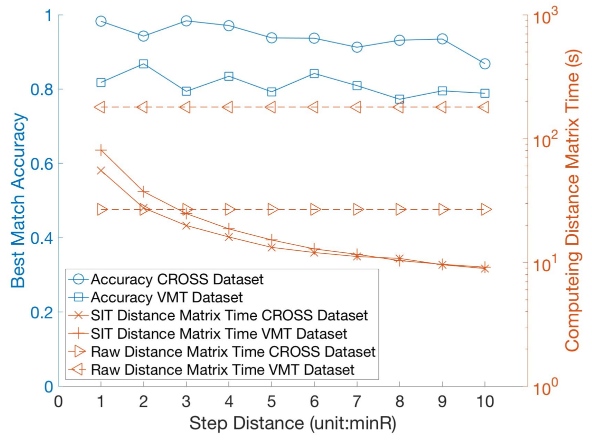


Figure 12: Best Match Clustering Accuracy and Running Time Using SIT representation on Computing the Distance Matrix with Different Step Distance on CROSS Dataset

matrices with different *Step Distances* for clustering. For CROSS dataset, the differences between SIT and raw representation are not significant since the average length of the trajectories in CROSS dataset is about 8. It is thus difficult to reduce the length when using SIT representation and preserve the accuracy at the same time. For VMT dataset, it takes 180.79 seconds to compute the BMED on raw trajecotries, while our method only needs about 10 seconds. Because the VMT dataset is over-sampled, the average length of the trajecotries is about 135. Using SIT representation can significantly reduce the length while keeping high accuracy. We can see that smaller *Step Distances* do not guarantee higher accuracy, but require higher computational cost.

5 CONCLUSION

In this work, we propose a linear time *Step-Invariant Trajectory* representation to automatically unify the uncertain sampled trajectory data. The experimental results show the superiority of our approach comparing to other distance measures on the raw data.

Representation	CROSS	Speed Up	VMT	Speed Up
AMKS (20 iterations)	286s	91.96	262s	192.64
Tube-Droplet	268s	86.17	184s	135.29
Our SIT ($r = 3$)	**3.11s**	1	**1.36s**	1

Table 3: Running Time for translating to new Representation

Our approach also achieves competitive results compared with the state of the art trajectory representations on effectiveness. The experiments show that when the raw dataset contains a lot of missing data, as long as the missing data points do not significantly change the shape of the original trajectory, our approach can still achieve high accuracy while the accuracy using other distance measures/representations degrades significantly. Furthermore, on average, translating to our *Step-Invariant Trajectory* representation is 115 times faster than using the model-based approaches[14, 29]. Finally, it is important that a good representation can be easily used on other datasets and algorithms. Our approach only needs to set one intuitive parameter while other representations need to carefully tune more than three parameters. In addition to using *Step-Invariant Trajectory* representation with *BMED/PBMED* as a standalone method, the SIT representation can also be integrated with other methods. Specifically, our method can unify the data distribution and improve the distance measure after the trajectory reconstruction step. Our method can also be used as a pre-processing step to provide high quality input data for model-based representations.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Weiyao Lin and Hongteng Xu for providing the Matlab code of the *Tube-Droplet* and AMKS methods on their websites.

The authors would also like to thank the Yang Zhou for providing the suggestions on tuning the parameters of the *Tube-Droplet* method.

REFERENCES

[1] Mahmuda Ahmed, Sophia Karagiorgou, Dieter Pfoser, and Carola Wenk. 2015. A comparison and evaluation of map construction algorithms using vehicle tracking data. *GeoInformatica* 19, 3 (2015), 601–632.

[2] Lei Chen, M. Tamer Özsu, and Vincent Oria. 2004. Symbolic Representation and Retrieval of Moving Object Trajectories. In *Proceedings of the 6th ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR '04)*. ACM, New York, NY, USA, 227–234. https://doi.org/10.1145/1026711.1026749

[3] Christos Faloutsos, M. Ranganathan, and Yannis Manolopoulos. 1994. Fast Subsequence Matching in Time-series Databases. In *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data (SIGMOD '94)*. ACM, New York, NY, USA, 419–429. https://doi.org/10.1145/191839.191925

[4] Joachim Gudmundsson, Patrick Laube, and Thomas Wolle. 2012. *Computational Movement Analysis.* Springer Berlin Heidelberg, Berlin, Heidelberg, 423–438. https://doi.org/10.1007/978-3-540-72680-7_22

[5] W. Hu, X. Li, G. Tian, S. Maybank, and Z. Zhang. 2013. An Incremental DPMM-Based Method for Trajectory Clustering, Modeling, and Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 5 (May 2013), 1051–1065. https://doi.org/10.1109/TPAMI.2012.188

[6] Young-Kee Jung, Kyu-Won Lee, and Yo-Sung Ho. 2001. Content-based event retrieval using semantic scene interpretation for automated traffic surveillance. *IEEE Transactions on Intelligent Transportation Systems* 2, 3 (Sep 2001), 151–163. https://doi.org/10.1109/6979.954548

[7] H. W. Kuhn and Bryn Yaw. 1955. The Hungarian method for the assignment problem. *Naval Res. Logist. Quart* (1955), 83–97.

[8] Jae-Gil Lee, Jiawei Han, and Xiaolei Li. 2008. Trajectory Outlier Detection: A Partition-and-Detect Framework. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering (ICDE '08)*. IEEE Computer Society, Washington, DC, USA, 140–149. https://doi.org/10.1109/ICDE.2008.4497422

[9] Jae-Gil Lee, Jiawei Han, Xiaolei Li, and Hector Gonzalez. 2008. TraClass: trajectory classification using hierarchical region-based and trajectory-based clustering. *Proceedings of the VLDB Endowment* 1, 1 (2008), 1081–1094.

[10] Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. 2007. Trajectory clustering: a partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. ACM, 593–604.

[11] C. Li, Z. Han, Q. Ye, S. Gao, L. Pang, and J. Jiao. 2014. Locality-Constrained Sparse Reconstruction for Trajectory Classification. In *2014 22nd International Conference on Pattern Recognition*. 2602–2606. https://doi.org/10.1109/ICPR.2014.449

[12] Yang Li, Yangyan Li, Dimitrios Gunopulos, and Leonidas Guibas. 2016. Knowledge-based Trajectory Completion from Sparse GPS Samples. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '16)*. ACM, New York, NY, USA, Article 33, 10 pages. https://doi.org/10.1145/2996913.2996924

[13] Zhenhui Li, Jae-Gil Lee, Xiaolei Li, and Jiawei Han. 2010. Incremental clustering for trajectories. In *International Conference on Database Systems for Advanced Applications*. Springer Berlin Heidelberg, 32–46.

[14] W. Lin, Y. Zhou, H. Xu, J. Yan, M. Xu, J. Wu, and Z. Liu. 2017. A Tube-and-Droplet-based Approach for Representing and Analyzing Motion Trajectories. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PP, 99 (2017), 1–1. https://doi.org/10.1109/TPAMI.2016.2608884

[15] N. Magdy, M. A. Sakr, T. Mostafa, and K. El-Bahnasy. 2015. Review on trajectory similarity measures. In *2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*. 613–619. https://doi.org/10.1109/IntelCIS.2015.7397286

[16] B. T. Morris and M. M. Trivedi. 2011. Trajectory Learning for Activity Understanding: Unsupervised, Multilevel, and Long-Term Adaptive Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33, 11 (Nov 2011), 2287–2301. https://doi.org/10.1109/TPAMI.2011.64

[17] C. Myers, L. Rabiner, and A. Rosenberg. 1980. Performance tradeoffs in dynamic time warping algorithms for isolated word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 28, 6 (Dec 1980), 623–635. https://doi.org/10.1109/TASSP.1980.1163491

[18] A. Naftel and S. Khalid. 2006. Motion Trajectory Learning in the DFT-Coefficient Feature Space. In *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*. 47–47. https://doi.org/10.1109/ICVS.2006.41

[19] Nikos Pelekis, Ioannis Kopanakis, Evangelos E. Kotsifakos, Elias Frentzos, and Yannis Theodoridis. 2011. Clustering uncertain trajectories. *Knowledge and Information Systems* 28, 1 (2011), 117–147. https://doi.org/10.1007/s10115-010-0316-x

[20] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. 2009. CRAWDAD dataset epfl/mobility (v. 2009-02-24). Downloaded from http://crawdad.org/epfl/mobility/20090224. (Feb. 2009). https://doi.org/10.15783/C7J010

[21] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. 2009. CRAWDAD dataset epfl/mobility (v. 2009-02-24). Downloaded from http://crawdad.org/epfl/mobility/20090224. (Feb. 2009). https://doi.org/10.15783/C7J010

[22] Fatih Porikli and Fatih Porikli. 2004. Trajectory distance metric using Hidden Markov Model based representation. In *In Proc. ECCV PETS Workshop.*

[23] A. C. Sanderson and A. K. C. Wong. 1980. Pattern Trajectory Analysis of Nonstationary Multivariate Data. *IEEE Transactions on Systems, Man, and Cybernetics* 10, 7 (July 1980), 384–392. https://doi.org/10.1109/TSMC.1980.4308519

[24] Han Su, Kai Zheng, Jiamin Huang, Haozhou Wang, and Xiaofang Zhou. 2015. Calibrating Trajectory Data for Spatio-temporal Similarity Analysis. *The VLDB Journal* 24, 1 (Feb. 2015), 93–116. https://doi.org/10.1007/s00778-014-0365-y

[25] Nguyen Huy Thach and Einoshin Suzuki. 2010. A symbolic representation for trajectory data. In *The 24th Annual Conference of the Japanese Society for Artificial Intelligence, JSAI, Nagasaki*. 1–4.

[26] Kevin Toohey and Matt Duckham. 2015. Trajectory Similarity Measures. *SIGSPATIAL Special* 7, 1 (May 2015), 43–50. https://doi.org/10.1145/2782759.2782767

[27] Michail Vlachos, Dimitrios Gunopoulos, and George Kollios. 2002. Discovering Similar Multidimensional Trajectories. In *Proceedings of the 18th International Conference on Data Engineering (ICDE '02)*. IEEE Computer Society, Washington, DC, USA, 673–. http://dl.acm.org/citation.cfm?id=876875.878994

[28] Haozhou Wang, Han Su, Kai Zheng, Shazia Sadiq, and Xiaofang Zhou. 2013. An Effectiveness Study on Trajectory Similarity Measures. In *Proceedings of the Twenty-Fourth Australasian Database Conference - Volume 137 (ADC '13)*. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 13–22. http://dl.acm.org/citation.cfm?id=2525416.2525418

[29] Hongteng Xu, Yang Zhou, Weiyao Lin, and Hongyuan Zha. 2015. Unsupervised Trajectory Clustering via Adaptive Multi-Kernel-based Shrinkage. In *International Conference on Computer Vision (ICCV)*. IEEE.

[30] Dragomir Yankov, Eamonn Keogh, Jose Medina, Bill Chiu, and Victor Zordan. 2007. Detecting Time Series Motifs Under Uniform Scaling. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07)*. ACM, New York, NY, USA, 844–853. https://doi.org/10.1145/1281192.1281282

[31] Zhang Zhang, Kaiqi Huang, and Tieniu Tan. 2006. Comparison of Similarity Measures for Trajectory Clustering in Outdoor Surveillance Scenes. In *18th International Conference on Pattern Recognition (ICPR'06)*, Vol. 3. 1135–1138. https://doi.org/10.1109/ICPR.2006.392

[32] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. 2009. Mining Interesting Locations and Travel Sequences from GPS Trajectories. In *Proceedings of the 18th International Conference on World Wide Web (WWW '09)*. ACM, New York, NY, USA, 791–800. https://doi.org/10.1145/1526709.1526816