[Egyptian Informatics Journal 19 (2018) 165–177](https://doi.org/10.1016/j.eij.2018.03.001)

Contents lists available at [ScienceDirect](http://www.sciencedirect.com/science/journal/11108665)

Egyptian Informatics Journal

journal homepage: [www.sciencedirect.com](http://www.sciencedirect.com/)

[](http://crossmark.crossref.org/dialog/?doi=10.1016/j.eij.2018.03.001&domain=pdf)A comparative study of similarity evaluation methods among trajectories of moving objects

Nehal Magdy [⇑](#_bookmark0), Tamer Abdelkader, Khaled El-Bahnasy

*Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt*

a r t i c l e i n f o

*Article history:*

Received 3 March 2017

Revised 19 February 2018

Accepted 19 March 2018

Available online 26 March 2018

*Keywords:* Trajectory similarity Regression Interpolation Barcoding TWEDistance

a b s t r a c t

Evaluating similarity between moving objects’ trajectories has gained much attention in many applica- tion domains. There exist similarity measures in the literature that propose evaluating similarity between trajectories in the form of time stamped values. Their main drawback is that the similarity evaluation is affected by the different sampling rates as it is defined over sequences of time stamped values. One of these measures is the Time Warp Edit distance measure that was our base of measuring similarity in our TWEDistance operator. Therefore, in this paper, a comparative study is made between four different approaches: TWEDistance operator, regression, interpolation and curve barcoding. Similarity evaluation is made over trajectories of different sampling rates. Results show that interpolation achieves the highest accuracy compared to the other approaches with an average accuracy up to 90%. Experimental evaluation was made over synthetic and real datasets.

© 2018 Production and hosting by Elsevier B.V. on behalf of Faculty of Computers and Information, Cairo

University. This is an open access article under the CC BY-NC-ND license ([http://creativecommons.org/](http://creativecommons.org/licenses/by-nc-nd/4.0/)

[licenses/by-nc-nd/4.0/](http://creativecommons.org/licenses/by-nc-nd/4.0/)).

1. Introduction

There has been a tremendous growth in movement data due to availability of devices that could be used to track movement of objects. Moving objects are objects that change their location/value over time such as cars, stock shares, temperature and animals [[1]](#_bookmark35). Tracking their movement leads to a sequence of time stamped val- ues/locations known as trajectories. Trajectories hold the historical movement of an object and it can be expressed as a sequence of

positions [*l*1; *l*2; ··· ; *lm*] observed at discrete time instances [*t*1; *t*2; ··· ; *tm*] so a trajectory *T* can be expressed as follows *T* = [(*l*1; *t*1); (*l*2; *t*2); ··· ; (*lm*; *tm*)] where *m* is the length – number of observations – of *T* [[2,3]](#_bookmark36).

Evaluating similarity between trajectories is important for many application domains. For example, analyzing trajectories of customers in a supermarket to find similar movement patterns for a better management of products. Another example is finding

\* Corresponding author.

*E-mail addresses:* [nehalmagdy@cis.asu.edu.eg](mailto:nehalmagdy@cis.asu.edu.eg) (N. Magdy), [tammabde@cis.asu.](mailto:tammabde@cis.asu.edu.eg) [edu.eg](mailto:tammabde@cis.asu.edu.eg) (T. Abdelkader), [khaled.bahnasy@cis.asu.edu.eg](mailto:khaled.bahnasy@cis.asu.edu.eg) (K. El-Bahnasy).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

**Production and hosting by Elsevier**

frequent migration patterns of a migrating group of birds. In surveillance systems, it is possible to find suspicious object move- ments and rare trajectory patterns. Other domains such as stock and data analysis, recommender systems of travelling routes, rec- ommender systems for friends based on their interests, visited locations, likes, etc. and future prediction of phenomena such as storms, hurricanes and earthquakes [[3,4]](#_bookmark37).

Evaluating similarity between trajectories in their discretized form as a set of (*x*, *y*) pairs are affected by the sampling rate differ- ences. Applying discrete similarity measures on a trajectory and a resampled version – same trajectory with different sampling rate – of it may result a distance that is bigger than the distance between two truly different trajectories. While what is expected, is that a trajectory with its all resampled forms shall be the head nearest ones with the minimum distance (i.e. the same trajectory with dif- ferent sampling rate (each 2, 4 and 6 sec for example). Therefore, this paper handles the sampling rates issue based on the continu- ous form of a trajectory’s data. Three different approaches were used, which are regression, interpolation and curve barcoding.

Regression is used to predict an equation that best fits the tra- jectories’ points with the least error. For evaluating similarity between two trajectories R and S, R’s movement is approximated

by a function *f* (*n*) with minimum residual error *RSSR*. Then, how much the *f* (*n*) approximates S’s trajectory is evaluated with a min- imum error *RSSS*. Regression similarity is then equal to the absolute

difference between *RSSS* and *RSSR*. Three types of regression are

<https://doi.org/10.1016/j.eij.2018.03.001>

1110-8665/© 2018 Production and hosting by Elsevier B.V. on behalf of Faculty of Computers and Information, Cairo University. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

considered: Linear, Logarithmic and polynomial regression of degree up to 10.

Using interpolation, the objective is to estimate the value of the function that the data points represent for an intermediate value. For evaluating similarity between two trajectories R and S, R’s movement is divided into intervals and for each interval a function is generated. Then, the points of trajectory S are interpolated in R’s functions. Interpolation similarity is then equal to the Euclidean distance between the interpolated values and the original values of S. Two types of interpolation are used, Linear and Cubic Interpolation.

In curve barcoding, a trajectory is represented as an image and this image is converted into a binary barcode sequence and the Hamming distance – the number of positions at which the corre- sponding elements are different – is used to evaluate similarity between two trajectories’ barcodes.

The Experiments are based on the previously mentioned approaches in addition to a discrete similarity operator [[5]](#_bookmark38) – TWEDistance – that is based on Time Warp Edit Distance (TWED) [[6]](#_bookmark39). Similarity evaluation was based on different similarity mean- ings which are spatial proximity, speed and direction. The experi- ments were made on synthetic and real datasets. The results show that interpolation gives the highest accuracy results over the other approaches. So our contribution is twofold:

* Proposing generic continuous trajectory similarity measures based on regression, barcoding and Interpolation.
* A Comparative study between these continuous based approaches and our TWEDistance operator that is proposed in

[[5]](#_bookmark38), by evaluating the measures through a set of experiments over different datasets to show the accuracy of the approaches with the existence of different sampling rates.

The rest of the paper is organized as follows: In Section [2](#_bookmark2), the background and the related work are presented. In Section [3](#_bookmark5), a detailed discussion of the TWEDistance operator is given. In Sec- tion [4](#_bookmark7), three approaches which are regression, interpolation and curve barcoding and how they are utilized to evaluate the similar- ity are discussed. Then, the experiments, and the output results are shown in Section [6](#_bookmark26). Finally, Section [7](#_bookmark44) concludes the paper, and sets ideas for future research work.

1. Related work

One of the important topics related to moving objects is how to evaluate the similarity between their trajectories [[3,7]](#_bookmark37). The work in

[[8]](#_bookmark40) is based on spatial raw representation where trajectories’ fixes are aligned at the same positions, trajectories must have same number of fixes, local time shifting[1](#_bookmark3) is not taken into consideration and its efficiency decreases with the existence of noise. The works in [[9–12]](#_bookmark45), are based on the geometric shape of trajectories. They con- sider local time shifting, and compare trajectories of possibly differ- ent number of fixes. In [[9,10]](#_bookmark45), they consider local time shifting. Other works, such as in [[13,14]](#_bookmark46), conclude that similarity measures that are based on raw representation of trajectories are sensitive to rotation, shifting and scaling. Therefore, they proposed a measure based on movement direction. The work in [[13]](#_bookmark46) builds on the work in [[14]](#_bookmark47), and takes into account local time shifting, robustness to noise and the possibility of comparing trajectories of different lengths. Other measures that are based on time series representation of a trajectory are usually associated with a distance function that can either be metric or non-metric [[7]](#_bookmark41). A distance function is said to be metric if

it satisfies non-negativity, uniqueness, symmetry and triangle inequality [[6,7]](#_bookmark39). Based on the used distance function, these measures can be divided into two classes. The first class uses L1-norm and L2- norm as a distance measure such as dynamic time warping [[15]](#_bookmark48), edit distance with real penalty [[16]](#_bookmark48), and time warp edit distance [[6]](#_bookmark39). The second class scores similarity based on a matching threshold such as longest common subsequence [[17]](#_bookmark48), and edit distance on real sequences [[18]](#_bookmark48). They handle local time shifting, and trajectories can have different lengths. Another work is proposed in [[19]](#_bookmark48), where similarity is based on both the speed and the path of moving objects. It follows a warping approach based on dynamic time warping. This approach works on trajectories of different lengths and handles local time shifting, but it is not robust to noise.

Most of the proposed measures are not reliable with the exis- tence of different sampling rates between trajectories. For exam- ple, in [Fig. 1](#_bookmark1), Trajectory R and S are the same but with different sampling rates. Using TWED measure to evaluate similarity, the result is that the two trajectories are not the closest. Therefore, in this paper, a comparative study is made between a discrete based similarity operator – TWEDistance – and three continuous based approaches: regression, interpolation and curve barcoding.

R S

Fig. 1. Same trajectories with different sampling rate (R: every 200 sec, S: every 30 sec).

1 Local time shifting occurs when one element of one sequence is shifted along time axis to match an element of another sequence even if the two matched elements appear in different positions.

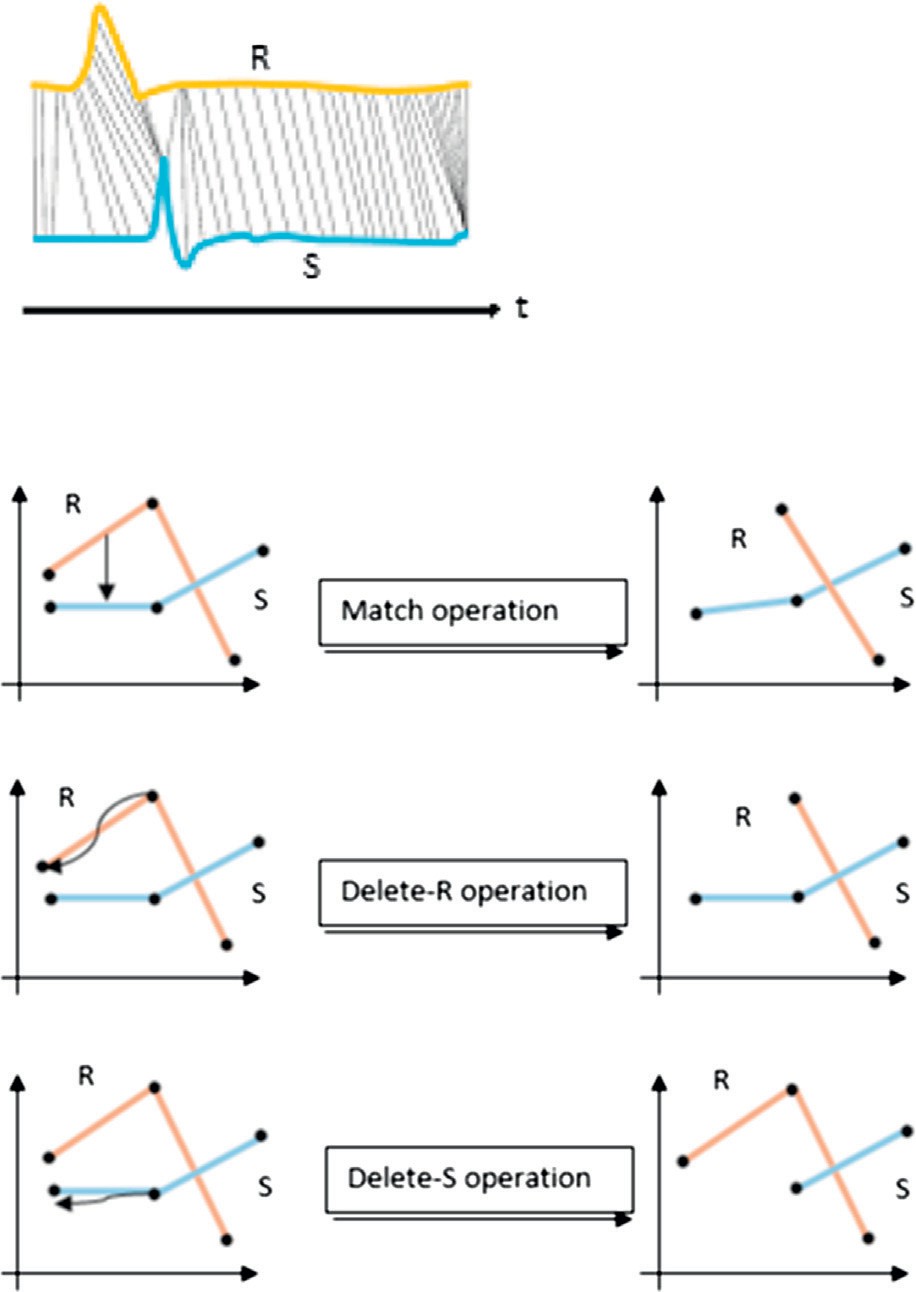




Fig. 2. TWED similarity measure.

1. TWEDistance discrete similarity based operator

TWEDistance operator [[5]](#_bookmark38) is a generic discrete similarity based operator that is built on top of one of the moving objects database

management systems which is called SECONDO [[20,21]](#_bookmark48). This oper-

differences are multiplied by a stiffness parameter c to control the elasticity. TWED, hence, accept four parameters *Rm*; *Sn*; k; c and returns the minimum of:

8> TWEDk;c(*Rm*—1; *Sn*)+ c · (*tr* — *tr* )

1

1

*m*

*m*—1

1 1

ator is proposed in our previous paper [[5]](#_bookmark38) and is based on the time warp edit distance measure (TWED) [[6]](#_bookmark39). TWED is favorable over

>

+*dlp*(*rm*; *rm*—1)+ k

*delete* — *R*

other discrete measures as it deals with local time shifting, a met- ric measure and an elastic metric as detailed in [[5]](#_bookmark38).

Given two trajectories *R* and *S*, as in [Fig. 2](#_bookmark4)(a), TWEDistance edits

TWEDk;c(*R*1 ; *S*1 )+ c · (*tsn* — *tsn*—1 )

>< +*dlp*(*sn*; *sn*—1)+ k

*m*

*n*—1

(1)

the two trajectories using three operations: *delete* , *delete* , and >

*delete* — *S*

*R S* TWEDk;c(*Rm*—1; *Sn*—1)+ *dlp*(*rm*; *sn*)

>

1

1

*match*. Each of the three operations has a penalty, and the distance between the two trajectories is the sum of penalties of the minimal-cost sequence of editing operations needed to transform one trajectory into the other.

:

A graphical illustration of the TWEDistance operations is given

>>>

+c · (*tsn* — *trm* )+ *dlp*(*rm*—1; *sn*—1)

+c · (*tsn*—1 — *trm*—1 )

### *Match*

Lambda and stiffness values are data dependent and can be

in [Fig. 2](#_bookmark4)(b). The *Delete* operation inside *R* or *S* involves dragging and dropping the sample to be deleted to its previous one. The cost associated with this delete operations is the length of the vector from the deleted sample to its previous one, using the *dlp* which is a user defined function. An extra associated constant penalty called Lambda k is added to this delete cost. So, the cost of deleting

element *m* from the series *R* would be *dlp*(*rm*; *rm*—1)+ k. The *Match*

operation involves dragging and dropping the segment between

two samples in the first trajectory to the matching segment in the second trajectory. The cost associated with the match opera- tion is the sum of lengths of two vectors connecting the start and

end of both segments, which is *dlp*(*rm*; *sn*)+ *dlp*(*rm*—1; *sn*—1).

To provide the controlled temporal elasticity, TWED includes

the time stamp difference in all its cost functions. These time

Algorithm 1. TWEDistance Algorithm

used for normalizing the value difference and the temporal differ- ence scales [[5]](#_bookmark38). The operator accepts trajectories of any meaning (e.g. the speed profile of a car, the spatial path, etc.) and the dis- tance function used can be any user defined function (i.e. L*n*- norm where n = 1, 2,.. .). TWEDistance operator, hence, supports genericness of both the meaning of similarity and the used dis- tance function.

[Algorithm 1](#_bookmark6) illustrates the evaluation of TWEDistance. It takes as an input two moving objects’ trajectories divided into two sequences the spatial values and the time-stamp values, TWED’s Stiffness and Lambda, and a user defined distance function. It returns as an output the TWEDistance that represents the sum of penalties of the minimal-cost sequence of editing operations needed to transform one trajectory into the other.

Input:

*R* ? The sequence of values of the first object

*S* ? The sequence of values of the second object

*tr* ? The sequence of time stamps for the first object

*ts* ? The sequence of time stamps for the second object

*dlp* ? The user defined distance function

k ? Lambda

c ? Stiffness

Output:

*TWED*[*RSize*; *SSize*] ? The cost/distance between two moving objects 1: *RSize* ← *Length*(*R*)

2: *SSize* ← *Length*(*S*)

3: Allocate *TWED*[0..*RSize*; 0..*SSize*] 4: for *i* ← 1; *SSize* do

5: *TWED*[0; *i*]←∞

6: end for

7: for *j* ← 1; *RSize* do

8: *TWED*[*j*; 0]←∞

9: end for

10: *TWED*[0; 0]← 0

11: for *i* ← 1; *RSize* do

12: for *j* ← 1; *SSize* do

13: *Delete* — *R* ← *TWED*[*i* — 1; *j*]+ *dlp*(*R*[*i* — 1]; *R*[*i*]) + c \* (*tr* [*i*]— *tr* [*i* — 1]) + k

14: *Delete* — *S* ← *TWED*[*i*; *j* — 1]+ *dlp*(*S*[*j* — 1]; *S*[*j*]) + c \* (*ts*[*j*]— *ts*[*j* — 1]) + k

15: *Match* ← *TWED*[*i* — 1; *j* — 1]+ *dlp*(*R*[*i*]; *S*[*j*]) + c \* (|*tr* [*i*]— *ts*[*j*]|))

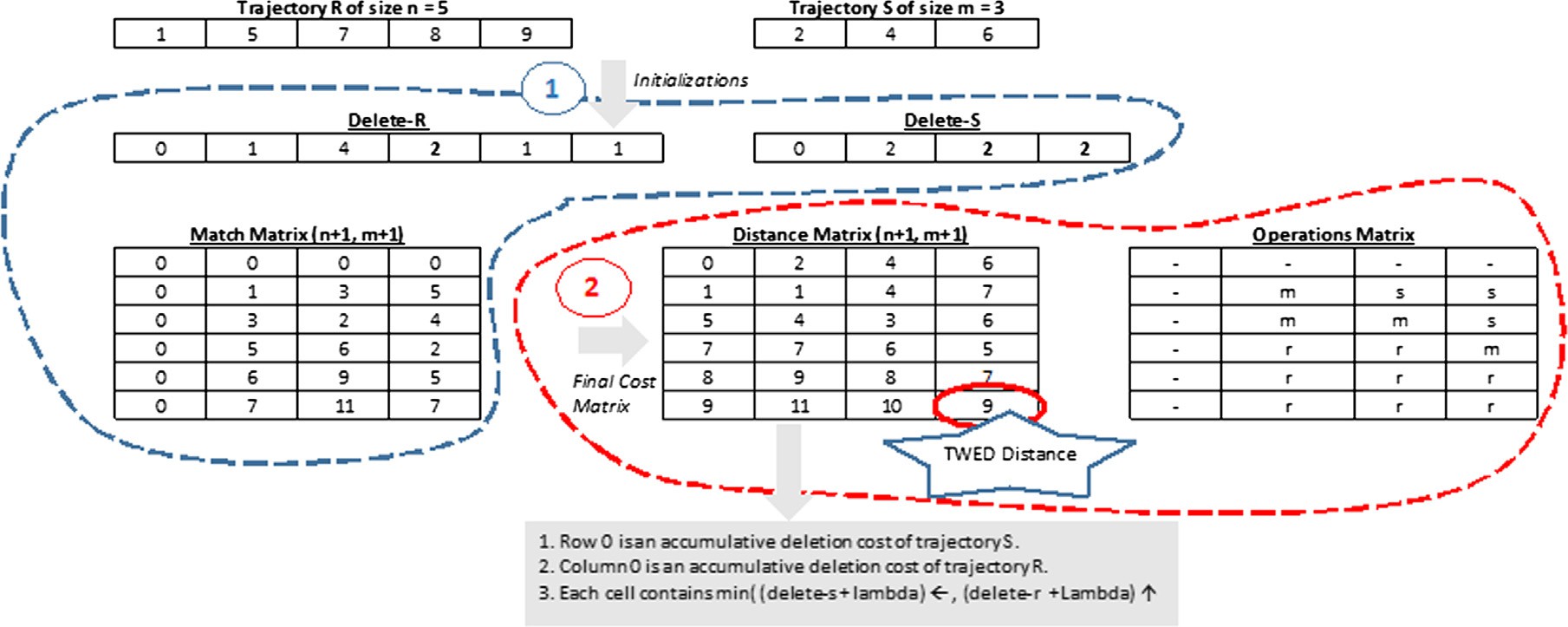
+*dlp*(*R*[*i* — 1]; *S*[*j* — 1]) + c \* (|*tr* [*i* — 1]— *ts*[*j* — 1]|)

16: *TWED*[*i*; *j*]← *Minimum*(*Delete* — *R*; *Delete* — *S*; *Match*)

17: end for

18: end for

19: return *TWED*[*RSize*; *SSize*]

Fig. 3. TWED example using *L*1 -norm, k = 1 and c = 0.0.

A TWEDistance example is shown in [Fig. 3](#_bookmark8) that shows how TWEDistance implementation works between two 1-D trajectories R and S, where four matrices are defined: Delete-R, Delete-S, Match and Final Distance Matrix [[5]](#_bookmark38).

Where Delete-S and Delete-R at position ‘index’ is measured as follows:

8>< 0; *if index* = 0

a is the intercept (The value of y when x equals zero).

b is the slope.

Given a set of *n* (*x*; *y*) pairs, the objective is to find the best fitting function *f* (*Xi*) with the minimum residual error that deviates from the observed data the least using least square method. Residual

error for a point at index *i* is evaluated as shown in Eq. [(3)](#_bookmark9), and for the entire set is as in Eq. [(4)](#_bookmark10). To achieve the objective, partial

*delete*[*index*]=

*dlp*(*samplei*—1; 0); *if index* = 1

>: *dlp*(*samplei*—1; *samplei*—2); *if index* > 1

derivative of the function *r*(*X*) is taken with respect to its coeffi- cients (e.g. a and b in Eq. [(2)](#_bookmark11)). After equalizing the partial deriva-

Match cost between the two trajectories R and S is measured as follows:

8> *dlp* (*Ri*—1; *Sj*—1)+ *dlp* (*Ri*—2 ; *Sj*—2 );

tives of the coefficients to zero, the result is a set of equations – where their number is equal to number of coefficients – to be solved using linear algebra and matrices.

*ri*(*xi*)= *yi* — *f* (*xi*) (3)

*Match*[*i*; *j*]= < *d* (*R* ; *S* );

>>: *lp i*—1 *i*—1

*if i* & *j* > 1

### *otherwise*

*n*—1

*r*(*X*)=

X

0

(*yi* — *f* (*xi*))2

(4)

where *dlp* is the used defined function. After building deletion and

match matrices, the final distance matrix holds the accumulative cost required (added to c ⁄ the time stamp differences) to superim-

pose both trajectories where *distanceMatrix*[*n* + 1; *m* + 1] is the

TWED distance [[5]](#_bookmark38).

where *yi* is the original value of *y* in the data and *f* (*xi*) is the pre- dicted value.

A linear regression model then can be solved using the follow- ing matrix:

*n* P*nxi* *a*

0

P*nyi*

P*n* P*n* 2 P*n*

0

=

4. Continuous based similarity measures

0*xi*

0*xi* *b*

0*xiyi*

In this paper, similarity evaluation is proposed to be based on the trajectory’s data as a continuous function. The proposed con- tinuous similarity evaluation was based on different approaches

regression [[22,23]](#_bookmark48) with function *Y* = *a*0 + *a*1*X* + *a*1*X*2 + *a*2*X*3+ The previous matrix could be generalized for polynomial

· + *apXp* of degree *p* as follows:

which are regression, interpolation and curve barcoding. In this

*n* P*nxi* P*nx*2

P*nx*3

·· · P*nxp*

*a*

P*ny*

section, these approaches and how they are utilized in evaluating 0

*x*

*x*

*nxi*

*n* 2

0 *i*

*n* 3

0 *i*

P

P

0 *i* 0 *i* 0 *i*

# P P P P

0

0 *i*

0

*x*

*x* + *a*1

*x yi*

*n* 4

0 *i*

·· ·

*n p* 1

0 *i*

*n*

0 *i*

P*nx*2 *n* 3

0 *i*

P *x*

the similarity are discussed.

0 *i*

*n* 4 *n* 5

P *x*

P *x*

0 *i*

0 *i*

0 *i* 0 *i*

·· · P*nxp*+2 *a*2 =

.

·· · P*nx*2*p* *ap*

P*nx*2*yi*

* 1. *Regression*

. . .

. . . . .

. . .

P *x*

P*nxp*

# P

*x*

Regression is one of the statistical curve fitting approaches in which an equation is predicted to describe discrete set of points.

*n p*+1

0 *i*

*n p*+2

0 *i*

0 *i*

. . . .

# P

P *x*

*n p*

*n p*+3

0 *i*

0 *xi yi*

.

0 *i*

One of its types is the linear regression that depends on one inde- pendent variable to predict the value of the dependent variable as described in Eq. [(2)](#_bookmark11).

*Y* = *a* + *bX* (2)

where

Y is the dependent variable that needs to be predicted. X is the independent variable.

Using Gauss-Jordan Elimination technique [[24]](#_bookmark48), the linear or polynomial system can be solved to get coefficient values by build- ing the augmented matrix as shown in Eq. [(5)](#_bookmark12) and converting it to its reduced echelon form using row operations (i.e. swapping rows, multiplication/division of rows by a non zero constant and adding/subtracting one row to a multiple of another row).

## (5)



Logarithmic functions could also be used for modeling a set of discrete points to a continuous function using linear transforma- tions [[25,26]](#_bookmark48) as shown in [Table 1](#_bookmark13). In this paper, similarity is eval- uated between trajectories using Linear, Logarithmic and Polynomial Regression.

* + 1. *Regression similarity*

in the form of a function *f* (*x*) with the minimum residual error In regression, the movement of the trajectory is approximated *RSS*. For evaluating the similarity between two trajectories R and

S, the following steps are followed and they are shown in [Fig. 4](#_bookmark15).

1. For a trajectory R represented as (x, y) pairs, a function *f* (*x*) is generated that approximates the movement with a residual

error *RSSR*.

1. The x value of the generated function *f* (*x*) is set with the x- values of the second trajectory S (*Sx*-*values* ) to generate the new *Sy*' -*values* values.
2. Then, the residual error is evaluated for the second trajectory

*RSSS* using sum of square differences between original y values

*Sy*-*values* and predicted y' values *Sy*' -*values* .

absolute difference between their RSS: |*RSSR* — *RSSS*|. 4. Finally the similarity between two trajectories R and S is the

Table 1

Linear regression transformation [[27]](#_bookmark49).

[Fig. 5](#_bookmark16), shows a numeric example for linear regression.

Where linear regression is used to approximate the trajectory R

1.323199212*x* + 1197.900894 with a residual error equal to points getting up with a linear function that equals to 258442.184. After substituting with the x values of S trajectory

in the linear equation, new y values are generated with a residual error equal to 293479514.9. After evaluating both error the simi- larity is the absolute difference between then that is equal to 293221072.716.

* 1. *Interpolation*

Interpolation is used for curve fitting and for constructing new

Having a discrete set of points represented as (*x*, *y*) pairs with the data points within the range of a discrete set of known data points. condition *x*0 < *x*1 < ... *x*N, it is required to estimate/interpolate the

diate *x* value where *xk* 6 *x* 6 *xk*+1 (k is an index inside the set of value of the function that the data points represent for an interme- pairs).

In this paper, two types of interpolation methods are considered:

* Linear Interpolation.
* Piecewise Cubic Interpolation.
  + 1. *Linear interpolation*

In Linear interpolation, If the value of the independent variable

Regression type

Function representation

Description

*x* falls between two data points (*x*0, *y*0 ) and (*x*1, *y*1) in the set of dis-

crete points, the value of the dependent variable *y* can be esti-

Logarithmic *y* = *a* + *bln*(*x*) Relation between *ln*(*x*) and *y* is linear. *a* and *b*

are evaluated using least square method on *y*

and *ln*(*x*) instead of *x*

mated using Eq. [(6)](#_bookmark14). For example, in [Fig. 6](#_bookmark17), the value of *y* at

*x* = 9000 is calculated to be *y* = 1.0342.

*x* — *x*

*y* = *y*0 + (*y*1 — *y*0 ) 0

1 — 0

*x x*

(6)

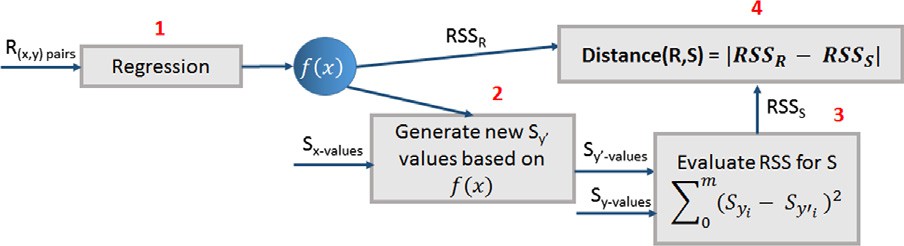


Fig. 4. Regression approach steps.

Linear interpolation on a discrete set of data points

{(*x*0, *y*0 ), (*x*1, *y*1), ··· , (*xn*, *yn* )} of length n is the concatenation of linear interpolants between each pair of data points. This results

in a continuous curve.

* + 1. *Piecewise cubic interpolation*

In this section, types of interpolation based on piecewise cubic Hermite interpolation [[28]](#_bookmark49) such as Shape-Preserving [[29,30]](#_bookmark49) and

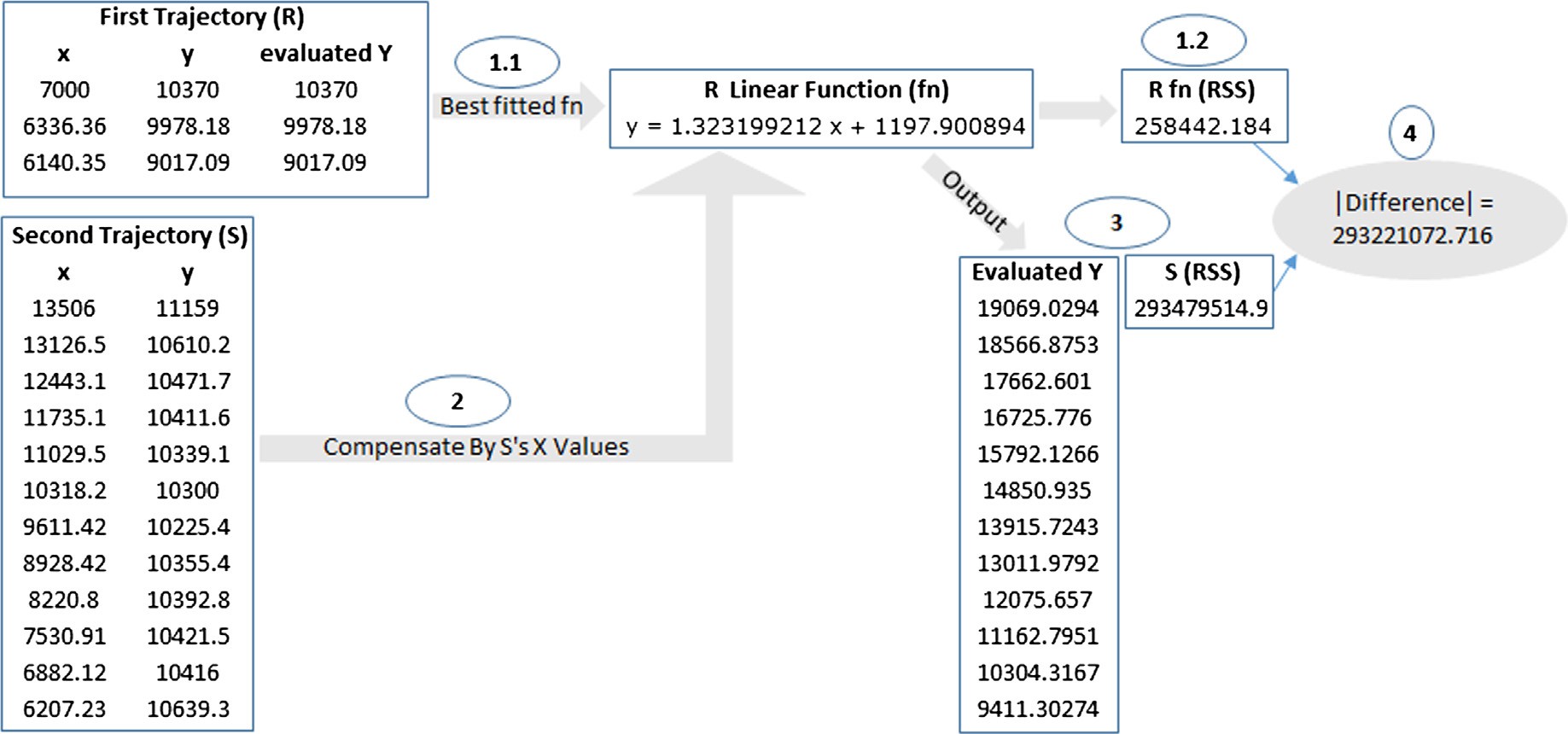


Fig. 5. Linear regression example.

1.2

1.18

1.16

1.14

1.12

1.1

y

1.08

1.06

1.04

y = 1.0342

1.02

1

x 104



(x0,y0) Interpolated (x,y)

(x1,y1)

data

fitted curve

0.7 0.8

x = 0.9 1 1.1 1.2 1.3 1.4

1.2

1.18

1.16

1.14

1.12

1.1

y

1.08

1.06

1.04

y = 1.0347

1.02

1

x 104



data

fitted curve

Interpolated (x,y)

0.7 0.8

x = 0.9 1 1.1 1.2 1.3 1.4

0.6

x x 104

0.6

x x 104

Fig. 6. Interpolation of *x* = 9000 that falls between [*x*0 , *x*1 ] using linear interpolation.

Fig. 7. Shape preserving cubic interpolation example where the *y* = 1.0347 at

*x* = 9000.

nomial of degree 3 for an interval [*xk*, *xk*+1]. cubic spline [[31,30,32]](#_bookmark49) are discussed. Both types generate a poly-

The polynomial is defined as follows

*x* = 9000 on the approximated curve which is found to be *y* = 1.0347. [Fig. 7](#_bookmark18) shows an example of interpolating the value of *y* at

*Cubic Spline Interpolation*

*P*(*x*)=

3*hs*2 — 2*s*3

*h*3 *yk*+1 +

*s*(*s* — *h* 2

)

*h*3 — 3*hs*2 + 2*s*3

*h*3 *yk* +

*s*2(*s* — *h*) *d*

*h*2 *k*+1

In cubic spline, the following system of equations are solved to get the slopes *d*:

*Ad* = *r*

+ *h*2 *dk* (7)

where *s* = *x* — *xk*, *h* = *xk*+1 — *xk*(interval length), *yk* = *P*(*xk*), *yk*+1 =

where *A* is a tridiagonal matrix represented as follows:

## *h*2 *h*1 + *h*2 0 0 .. . 0

*P*(*xk*+1

), *dk*

= *P*'(*xk*), *d*

*k*+1

= *P*'(*x*

*k*+1

) and *dk*

and *d*

*k*+1

are the slopes at

## *h*2 2(*h*2 + *h*1) *h*1 0 .. . 0

*xk* and *xk*+1 respectively. Both types differ in the way they compute

## 0 *h*3 2(*h*3 + *h*2) *h*2 .. . 0

the slopes, *dk*. Once the slopes have been computed, the interpolant

can be evaluated using the power form with the local variable s: .

.

.

*P*(*x*)= *yk* + *s dk* + *s*2 *ck* + *s*3 *bk*, where the coefficients of the quadratic

0 0 0 *h*

2(*h*

*n*—2

+ *h*

*n*—1

) *hn*—2

. . .

. . .

. . .

. . . .

and cubic terms are

0 0 0 0 *hn* 1 + *hn* 2) *hn* 2

*ck* = 3 d*k* — 2 *dk* — *dk*+1

*n*—1

### *h*

And *r* is

*r*1

— — —

*bk* =

*dk* — 2 d*k* + *dk*+1

*h*2

*y y*

*h*1d2 + *h*2d1

## *h*2d3 + *h*3d2

3 .

.

where d is measured as d*k*

= *k*+1 — *k* . In the following subsections,

*k*

*h*

how the slopes are measured for each type are presented.

*Shape-Preserving Piecewise Cubic Interpolation*

*hn*—2d*n*—1 + *hn*—1d*n*—2

In Shape-Preserving piecewise cubic interpolation, to evaluate *rn*

slopes there are two main cases:

* Slopes for the interior points
  1. If d*k* \* d*k*+1 6 0, then *dk* = 0.
  2. If d*k* \* d*k*+1 > 0 and the two intervals have the same length

where *r*1 and *rn* is evaluated using the following equations

(*h*1 + 2(*h*2 + *h*1)) \* *h*2 \* *y*2 —*y*1 + *h*2 \* *y*3 —*y*2

*r*1 = *x*2—*x*1 1 *x*3—*x*2

*h*2 + *h*1

2

then *d*

= 2 d*k*—1 d*k* .

*h* \* *yn*—1 —*yn*—2 + (2(*hn*—1 + *hn*—2 )+ *hn*—1)\* *hn*—2 \* *yn* —*yn*—1

*k* d*k*—1 +d*k*

*rn* = *n*—1 *xn*—1 —*xn*—2 *xn* —*xn*—1

* 1. If d*k* \* d*k*+1 > 0 and the two intervals have different lengths *hn*—1 + *hn*—2

d*k* d*k*—1 (x1 +x2 )

then *dk* =

x1 d*k* +x2 d*k*—1

x2 = *hk* + 2*hk*—1.

* Slopes for the data end points

where x1 = 2*hk* + *hk*—1 and

[Fig. 8](#_bookmark19) shows an example of interpolating the value of *y* at

*x* = 9000 that results in *y* = 1.0343.

1. At Point 0, *d*1 = ((2 *h*1 +*h*2 )d1)—(*h*1 d2 ) with the following cases

*h*1 +*h*2

## *d* = 0, *d*1 \* d1 6 0

1 3 d1, d1 \* d2 6 0 && | *d*1 | > 3 |d1 |

1. At Point n, *dn* = ((2 *hn*—1 +*hn*—2 )d*n*—1 )—(*hn*—1 d*n*—2 ) with the following

*n*—1 *n*—2

*h* +*h*

cases

*d* = 0, *dn* \* d*n*—1 6 0

## 3 d*n*—1, d*n*—1 \* d*n*—2 6 0 && | *dn* | > 3 |d*n*—1|

*n*

* + 1. *Interpolation similarity*

In interpolation, a trajectory is represented as ((x, y), t) pairs or (x, t) pairs where they are divided into intervals based on a sorted independent variable. Therefore, the time dimension is chosen as our independent variable for predicting (x, y) pair or x. For evalu- ating similarity between two trajectories R and S, the following steps are followed and they are shown in [Fig. 9](#_bookmark20):

1. For a trajectory R, the functions *f* (*t*) for each data interval are generated ((n — 1) functions).

1.2

1.18

1.16

1.14

1.12

1.1

y

1.08

1.06

1.04

y = 1.0343

1.02

1

x 104

0.7 0.8

x = 0.9 1 1.1 1.2 1.3 1.4

Algorithm 2. Radon Barcode Algorithm

Input:

*IMG* ? The input image

*numOfProj* ? Number of projections

*RN* ? Number of rows for

image normalization

*CN* ? Number of columns for

image normalization

Output:

*BRBC* ? Binary Radon Barcode

0.6

x x 104



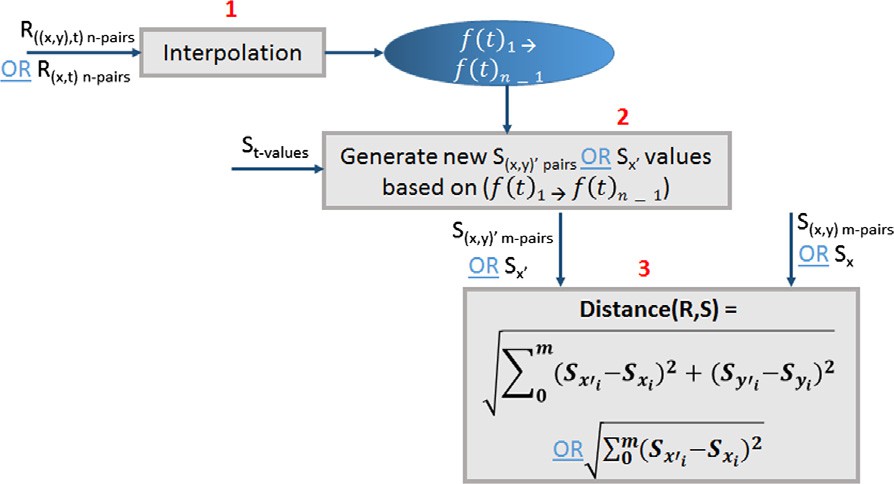
data

fitted curve

Interpolated (x,y)

1: *BRBC* ← £

Fig. 8. Cubic spline interpolation example where *y* = 1.0343 at *x* = 9000.



2: Initialize h ← 0

3: *IMG* ← Normalize(*IMG*, *RN*, *CN*) 4: while h < 180 do

5: Get projection q for the angle h

*Ttypical* ← *Median*q*i* –0 (q) 6: Find typical value: *b* ← q P *Ttypical* 7: Binarize projection:

8: Append the new binary

projection barcode at h to *BRBC*: Append(*BRBC*, *b*)

180

9: h ← h +

*numOfProj*

10: end while

11: return *BRBC*

Fig. 9. Interpolation based similarity approach.

erate corresponding new (x, y) pairs (*S*(*x*,*y*)' ) or new X (*Sx*' ) for S. 2. Then, *St*-*values* are interpolated in the generated functions to gen-

dean distance between the new predicted values (*S*(*x*,*y*)' or *Sx*' ) 3. Finally, the distance between R and S is evaluated using Eucli- and the original ones (*S*(*x*,*y*) or *Sx*).

* 1. *Radon barcoding*

Barcoding was proposed in medical image retrieval where it is used with feature based retrieval for better accurate results. One of the approaches that is used for image barcoding is the Radon Transform. Applying the Radon transform on an image f (x, y) for a given set of angles can be thought of as computing the projection of the image along the given angles. The resulting projection is the sum of the intensities of the pixels in each direction (i.e. a line inte-

gral). The result is a new image *R*(q, h). The new image can be

q = *x* cos h + *y* sin h. Then, Radon Transform of the image can be expressed using its projections which is defined as follows expressed as defined in Eq. [(8)](#_bookmark22).

* + 1. *Barcode similarity*

Similarity evaluation using curve barcoding deals with a trajec- tory as an image instead of a sequence of points. To evaluate sim- ilarity between two trajectories R and S, the following steps are followed:

1. After converting both trajectories (R and S) from (x, y) pairs into an image, barcode sequences are generated for them (*Rbarcode* and *Sbarcode*). [Fig. 10](#_bookmark23) shows an example of a trajectory image and its corresponding barcode.
2. Then, similarity between trajectories’ barcodes is evaluated using Hamming distance that can be defined as

*n*

*Distance*(*Rbarcode* , *Sbarcode* )= *Rbarcodei* – *Sbarcodei*

X

0

where n is the length of the barcodes. [Fig. 11](#_bookmark24) shows an example for Hamming distance evaluation between two trajectories’ barcodes.

*R*(q

## , h)=

Z +∞ Z +∞ *f x y*

) d(q —

*x* cos h —

*y* sin

### *dxdy*

(8)

—∞ —∞

( ,

where d(·) is the Dirac delta function. In [[33]](#_bookmark49), the authors proposed

to binarize the resulted radon transform. Therefore, after generating

radon transform of an image, the projections are thresholded by cal- culating a typical value using the median operator. This median operator is applied to all non-zero values of each projection. After predicting median value it is used for building the binarized radon barcode where values P median is replaced with 1, otherwise it is

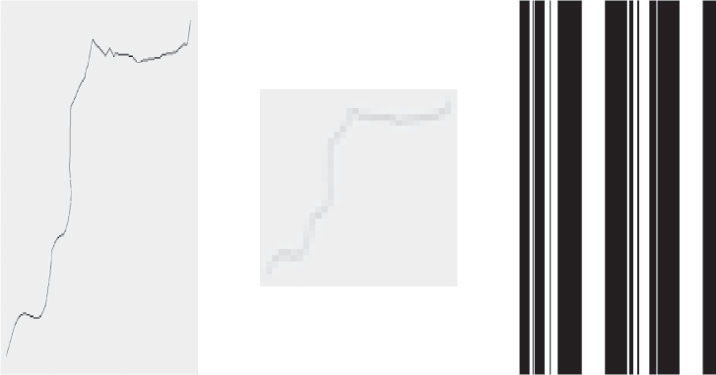
are normalized and resized into *RN* × *CN* images (i.e. replaced with 0. To receive barcodes with the same length, images

*RN* = *CN* = 2*n*; *n* ∈ N+).

Original Image

Radon Barcode

0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1



Normalized Image

[Algorithm 2](#_bookmark21) describes how the binary radon barcodes are generated.

Fig. 10. Example of a 32 × 32 normalized trajectory image using 4 projections barcoding.

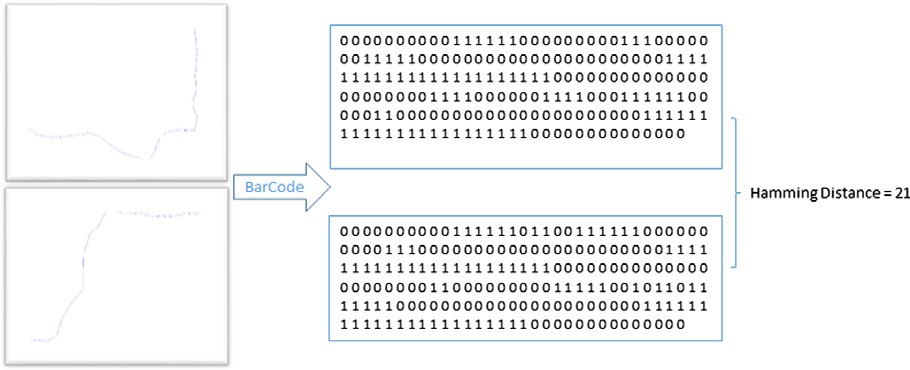


Fig. 11. Hamming distance between two trajectories’ barcodes example.

* 1. k-nearest neighbor classification

The k-nearest neighbors (k-NN) [[34]](#_bookmark49) is a method that is used for classification in our experiments. The input consists of set of trajec- tories and a query trajectory that needs to be classified. Classifica- tion is done by evaluating distance between the query trajectory and the other trajectories. The predicted class is evaluated based on the K specified value. If k = 1 then the predicted class is the class of the first nearest neighbor based on the evaluated distance. If k > 1 then the predicted class is the most frequent class in the k near- est neighbors. The distance can be evaluated using any distance metric such as Euclidean distance (ED).

[Algorithm 3](#_bookmark25) describes how K-NN Works.

Algorithm 3. K-NN Classification

Berlintest dataset contains the *Trains* relation. This relation was created by simulating the underground trains in Berlin. Simulation was based on the real train schedule and the real underground net- work of Berlin. The period of simulation is about 4 h in one day. It contains 9 lines identified by an ID (Line attribute) and group of trains travel per each line and identified also by an ID. Each train’s trajectory is stored in a *Trip* attribute. This relation contains 562 trajectories, consisting of a total of 54,595 observations. Trucks dataset consists of 1100 trajectories of 50 trucks delivering con- crete to several construction places around Athens with a total of 188,197 observations. Geo-Life is a GPS trajectory dataset that was collected in the Geolife project (Microsoft Research Asia) by

182 users in china in a period of over three years (from April 2007 to August 2012). It contains 17,621 trajectory with different sampling rates. Each trajectory is represented by a sequence of time-stamped points, each of which contains the information of latitude, longitude and altitude. The dataset were filtered to year 2008 that contains 7334 trajectory with 5,474,814 observations.

* 1. *Spatial based similarity*

In this experiment, the accuracy of the four approaches is eval- uated based on the spatial similarity and how it is affected by dif- ferent sampling rates. Trains dataset is filtered by picking up 9 trains one from each line then resampling these trains to ten differ- ent samples coming up with 90 trajectories. Same for Trucks data- set, they are filtered into 205 trajectory and each trajectory was resampled to 6 samples yielding 1230 trajectories. These final tra-

jectories contain from 4 to 287 samples per each. [Table 2](#_bookmark27) concludes

Input:

[*x*, *class*] ? the input trajectories with their class label

*q* ? the query trajectory

*k* ? the k value for

nearest neighbors

Output:

*c* ? The predicted class

label

1: *XSize* ← *Length*(*X*)

2: Distances[0..*XSize*] ? Contains distance

with the class label

3: for *i* ← 1 to *XSize* do

4: Compute distance

(*di*([*x*, *class*]*i* , *q*),class) Distances.add

5: end for

6: sort(Distances) Ascending

7: if *k* == 1 then c ← the trajectory that has *d*0 with

q

8: else if *x* > 1 then

9: c ← majority class of Distances from index 0 to k

10: end if

11: return c

* 1. Experiments and results

In the experiments, the similarity between trajectories is evalu- ated using the four approaches. Three datasets are used, SECON- DO’s berlintest synthetic dataset [[35]](#_bookmark49), the Trucks real dataset

[[36]](#_bookmark49) and the Geo-Life real dataset [[37–39]](#_bookmark49).

the datasets description that was used in this experiment.

In regression similarity, for all trajectories, linear, logarithmic and polynomial regression – degree 10 or less if trajectory has less than 9 samples – are used to predict the function that best fits tra- jectories’ points with the minimum residual error. For each trajec-

tory’s function *f* (*n*), the x variable are set with the x values of the

much the function *f* (*n*) fits the points. After that the Residual sum other trajectories’ points to evaluate the *Y* values to evaluate how ence between *f* (*n*) RSS and other trajectories’ RSS are computed of squares RSS as defined in Eq. [(4)](#_bookmark10) is measured. Then the differ- and the results are sorted according to the evaluated difference.

In TWED similarity, the generic TWEDistance operator that is implemented in SECONDO [[5]](#_bookmark38) is used. The Similarity evaluation is based on spatial proximity with stiffness = zero, Lambda = 1 and Euclidean Distance as our distance function [[5]](#_bookmark38). The Query used is as follows

let SpatialDist=

Trucks feed

extend[Dist: TWEDistance (

.Trip\_a,

.Trip\_b, 0.0,

1.0,

fun(x: point, y: point)

sqrt( pow( getx(x) - getx(y), 2 ) +

pow( gety(x) - gety(y), 2 )))] sortby[ID\_a asc, Dist asc]

consume;

where *Tripa* is the first trajectory and *Tripb* is the second one. After evaluating TWED distance between all pairs of trajectories, results were sorted with respect to the computed distance.

In radon barcode similarity, trajectories are represented as a curve image. Radon transform approach is then applied to the

Table 2

Description of trucks and trains datasets.

Parameters Dataset Trucks Trains

Number of trajectories 1100 562

After evaluating accuracy per each trajectory, the total accuracy is computed using the following formula:

## *TN* Trajectory Accuracy

P

### *Total Accuracy* = 0

*T*

*N*

Number of Observations/points

Number of Filtered Trajectories

Number of versions per trajectory (*VN* )

Total number of trajectories including versions (*TN* )

188,197 54,595

205 9

6 10

1230 90

[Fig. 13](#_bookmark30) shows the results of the four approaches using the head nearest neighbor approach where interpolation gives higher accu- racy on both datasets.

* 1. *Direction based similarity*

In this experiment, accuracy of the approaches is based on the

Sampling rates Every (100, 50,

900, 80, 200) sec

Every (100, 50, 80, 200, 60,

30, 70, 20 and 15) sec

direction similarity. The *Trains* relation of the *berlintest* dataset

was used as each trajectory is labeled with its direction. It contains

Number of samples per trajectory

Attributes defines a trajectory

4–287 sample

ID, Version\_ID and Trip(Trajectory)

trajectories of trains travelling at different lines. The subset of trains that travel on Line 1 that has a total of 58 trajectories are selected, from which 29 are heading on one direction, and the rest are heading on the opposite direction. One train is picked from

each group[2](#_bookmark28) and then re-sampled into five versions. The sampling

image after normalization (32 × 32) and the result is thresholded

to generate the binarized barcode. For both datasets, radon trans-

form is applied using different values of projections (i.e. 4, 8, 16 and 32) for all trajectories’ images then binary barcode is gener- ated and hamming distance is used to measure how much the bar- codes are similar. The hamming distance results are then sorted.

In interpolation similarity, the two types of previously dis- cussed interpolation are used. First, the approximated curve of each trajectory are generated then the *x* values of the other trajec- tories’ points are interpolated to generate the *y* values (i.e. The *x*

values in this experiment are the time and *y* values are the (*x*, *y*)

coordinates). Then, the distance is measured using Euclidean Dis-

tance between the interpolated points and the original one. Finally, the results are sorted according to the evaluated distance.

* + 1. *Accuracy evaluation*

Accuracy of the spatial similarity experiment is evaluated based on Head Nearest Neighbor approach. After evaluating distance between all trajectories, for each trajectory, the results are sorted from the least distance to the maximum. After that, the head k tra- jectories are evaluated if they contain trajectories from the same class under the assumption that trajectory and its versions form a class. In the trains dataset, each trajectory has a total number of versions (*VN*) equals to 10 with a total number of trajectories (*TN*) equals to 90. In the trucks dataset, each trajectory has a total number of versions (*VN*) equals to 6 with a total number of trajec- tories (*TN*) equals to 1230. First, accuracy per each trajectory is evaluated as follows:

rates were every 42, 50, 80, 100, 200 sec.

In interpolation, regression and binary barcode, the time was considered as the x-axis and direction as the y-axis. The results should be as follows

* + - * Each train should have the minimum distance with all other trains travelling at the same direction (i.e. Head distance results

should be the other trains (train’s versions) that travel at the same direction then comes the ones travelling at the opposite direction).

In TWED, the following query is used with lambda = 1.0 and stiffness = 0.0:

let DirectionDist= TrainsLine1 feed extend[Dist: TWEDistance (

direction(.Trip\_a), direction(.Trip\_b), 0.0,

1.0,

fun(x: real, y: real) abs( x - y))]

sortby[ID\_a asc, Dist asc] consume;

where TrainsLine1 is the relation that hold trajectories of all trains

### *Trajectory Accuracy*

*Nsameclass*

= *VN*

travelling at Line 1. *Tripa* and *Tripb* are the trajectories of two trains in each tuple where distance is measured between their direction using absolute difference as our defined function. After that, the

where *Nsameclass* is the number of trajectories of the same class that appears at the Head K. For each trajectory, k is determined based on the distance value. K is increased by 1 while reaching the follow- ing condition: if the next trajectory to compare with is not of the same class and previous trajectory’s distance is different from the current one (i.e. stops when a different id appears). [Fig. 12](#_bookmark29) shows two illustrative examples for evaluating accuracy of a trajectory with *VN* equals to 7. In the first example, trajectory with id equal to 1 has head four trajectories with zero distance. The ones having same id are two. Therefore, k here is equal to 2 as there are other trajectories with different ids appears in zero distance. In second example, k is equal to 4 as in the zero distance there are three tra- jectories with the same id (initially k is equal to three), no different ids appears in that distance so go to the next distance (1717.17). In the next distance there is one trajectory with the same id so k now is 4, stop here as there is different id appears in that distance.

results are sorted ascending according to the computed distance.

Same approach followed in the spatial experiment is used in this experiment for evaluating the accuracy of the approaches using *VN* equal to 5 and *TN* equal to 10 (i.e. the stopping condition is when a different direction appears). [Fig. 14](#_bookmark31) shows the accuracy results for this experiment where cubic interpolation, linear inter- polation and TWEDistance gives higher accuracy.

* 1. *Speed based similarity*

In this experiment, the Geo-life dataset is used as each trajec- tory is labeled by a transportation mode. These modes include car, bus, train, bike, walk, etc. This experiment is applied on 46

2 Only one train is picked from each group because all trains in a group will have the same direction values as they travel at the same line and direction.

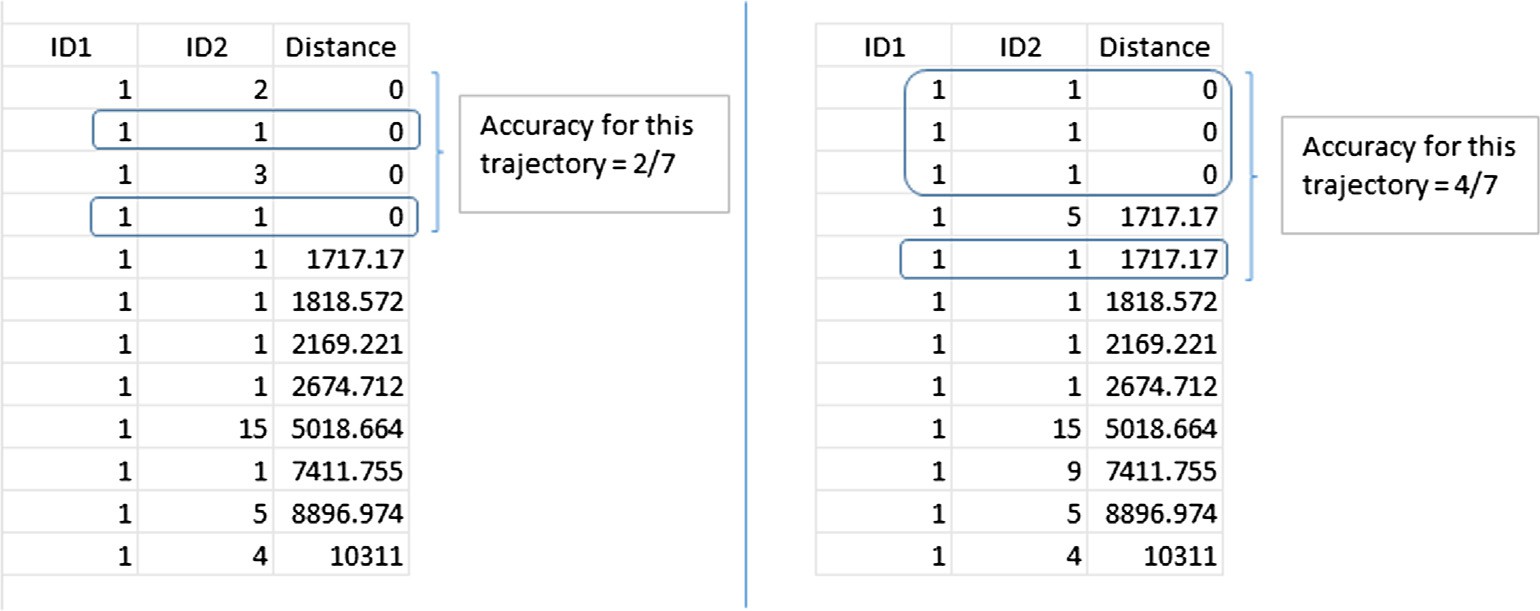


Fig. 12. Trajectory accuracy evaluation examples with a total number of versions *VN* equal to 7.

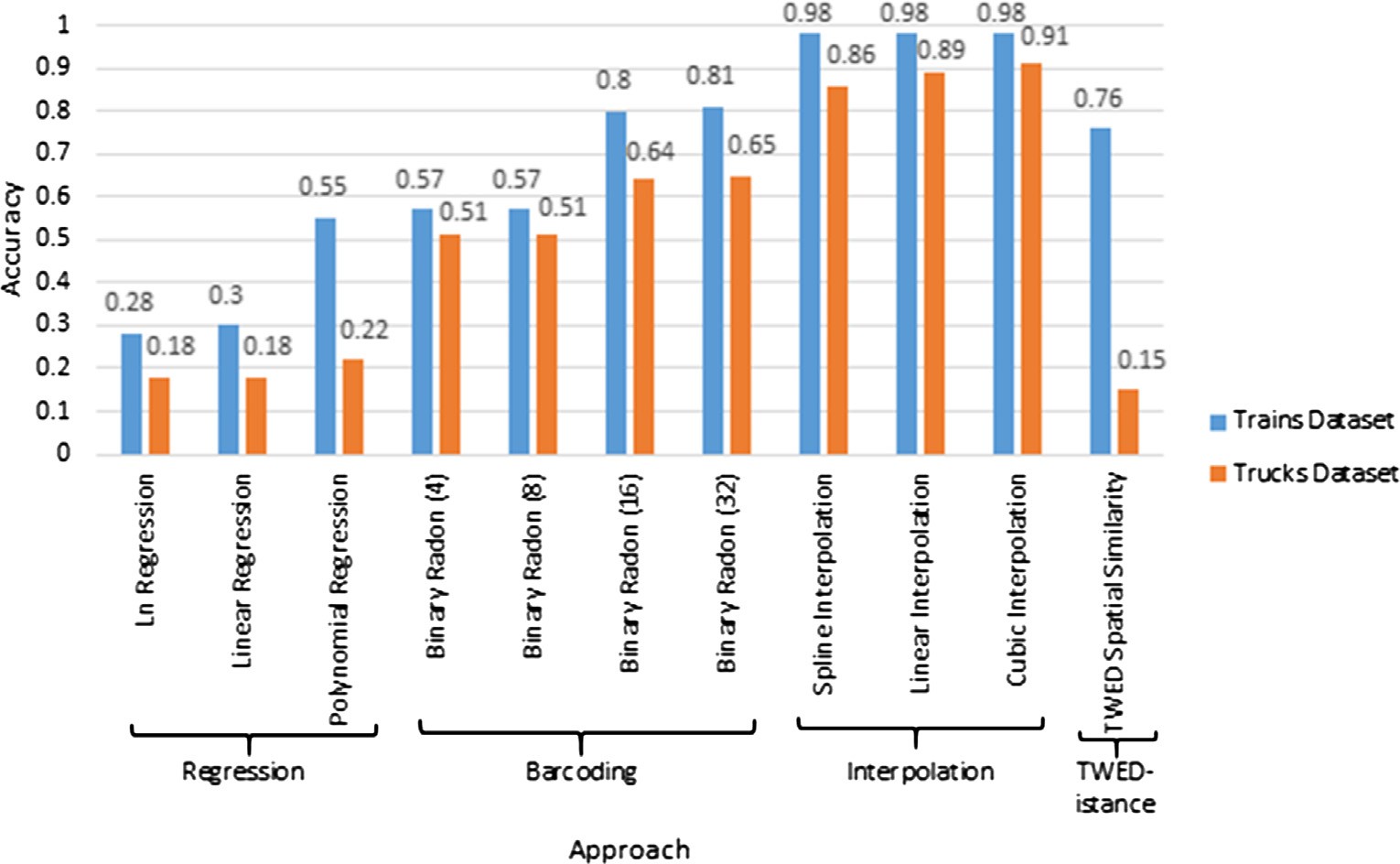


Fig. 13. Head nearest accuracy results where cubic interpolation in the experiments is the shape preserving type.

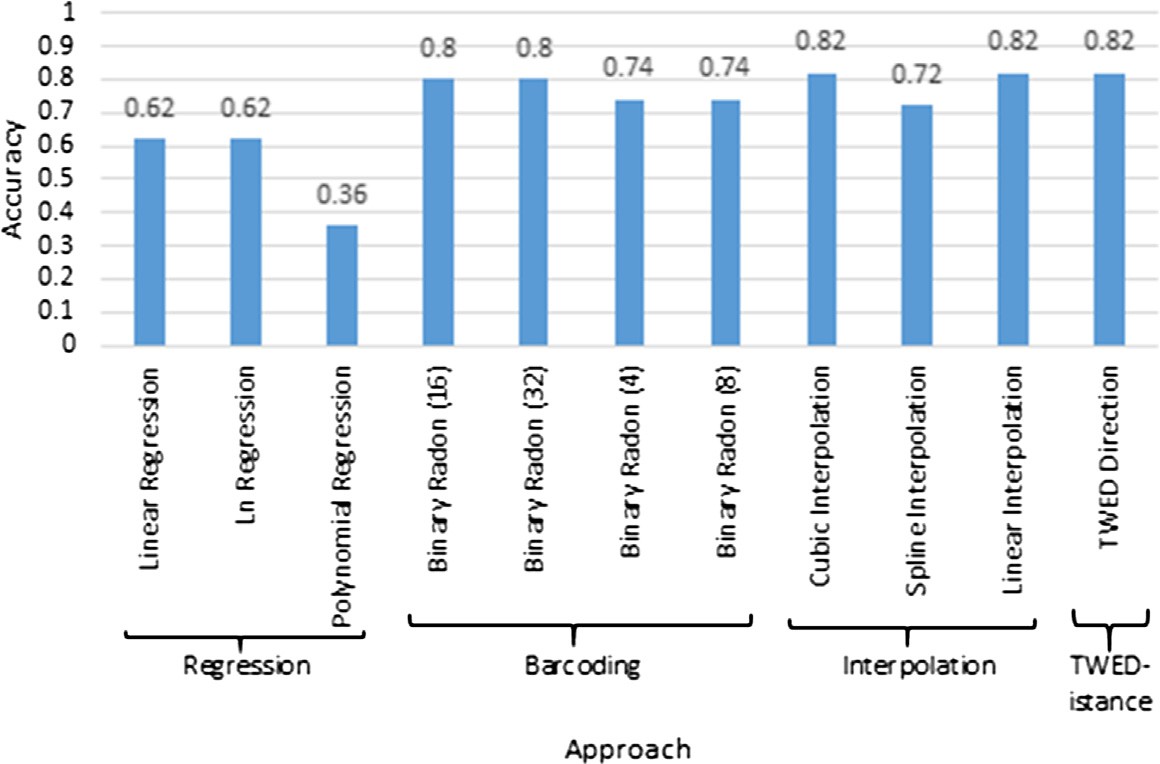


Fig. 14. Direction based similarity accuracy.

trajectories including original trajectories and sampled versions from them. The sampling rates differs based on the transportation mode as some sampling rates causes number of observations to be less that what is needed (i.e. in cubic interpolation at least four observations are needed).

In interpolation, regression and binary barcode, the time was considered as the x-axis and speed as the y-axis. The results should be as follows

* Each train should have the minimum distance with all other trains having the same transportation mode.

In TWED experiment, the following is the used query with lambda = 50.0 and stiffness = 0.0:

let SpeedDist=

Trucks feed

extend[Dist: TWEDistance ( speed(.Trip\_a),

speed(.Trip\_b),

where Trucks is the relation that hold trajectories of all trains trav- elling at Line 1. *Tripa* and *Tripb* are the trajectories of two trains in each tuple where distance is measured between their speed using

absolute difference as our defined function. After that, the results are sorted ascending according to the computed distance.

Same approach used for evaluating the accuracy in the spatial and direction based similarity is used here with *TN* equal to 46 and different *VN* for each transportation mode (i.e. the stopping condition is when a different mode appears). [Table 3](#_bookmark32) shows each mode with its *VN*.

[Fig. 15](#_bookmark34) shows the accuracy results for all approaches where regression and interpolation gives higher accuracy.

* 1. *Results*

In this section, the overall accuracy for all approaches is evalu- ated based on the average accuracy of all experiments. For evaluat- ing the average accuracy, the following formula is used:

P *CExpersameclass*

0.0,

50.0,

### *Avg Accuracy* =

P *TN*

*Exper*

fun(x: real, y: real) abs( x - y))]

sortby[ID\_a asc, Dist asc] consume;

Table 3

Transportation mode and *VN* .

Mode *VN*

Train 11

Taxi 4

where *CExpersameclass* is the number of trajectories of the same class that appears at the Head K per each experiment and *TNExper* is the total number of trajectories per each experiment as shown in [Table 4](#_bookmark33).

After evaluating the average accuracy, the results are as shown in [Fig. 16](#_bookmark42).

In the results of average accuracy, interpolation shows the high- est accuracy results with the existence of sampling rate differences with an average accuracy up to 90% as it generates an equation for each data interval.

Table 4

Total number of trajectories per each experiment.

|  |  |  |  |
| --- | --- | --- | --- |
| Bus  Walk | 7  4 | Experiment | *TN* |
| Bike | 4 | Spatial similarity | 1320 (i.e. 1230 + 90) |
| Subway | 6 | Direction similarity | 10 |
| Car | 10 | Speed similarity | 46 |

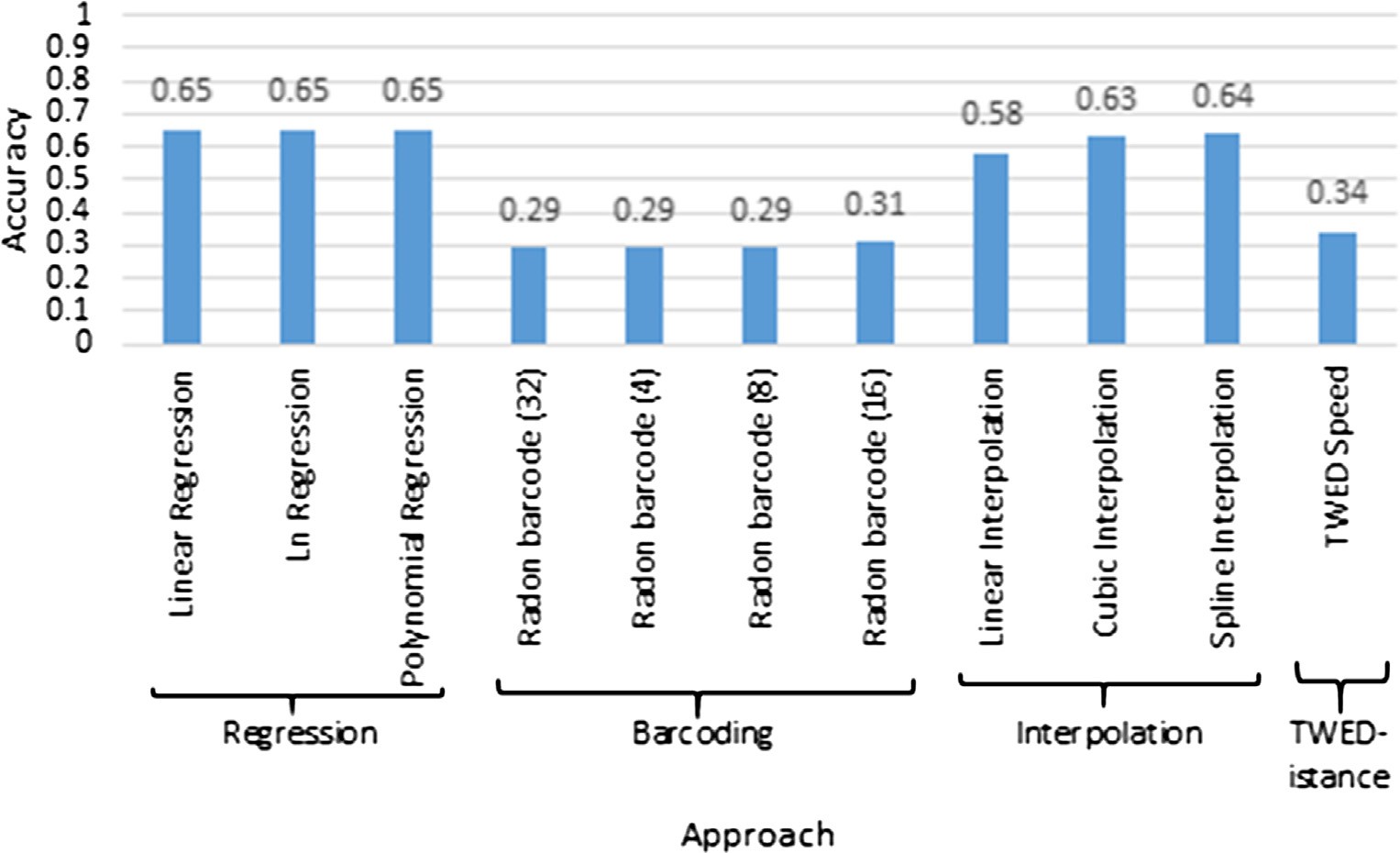


Fig. 15. Accuracy of speed similarity.

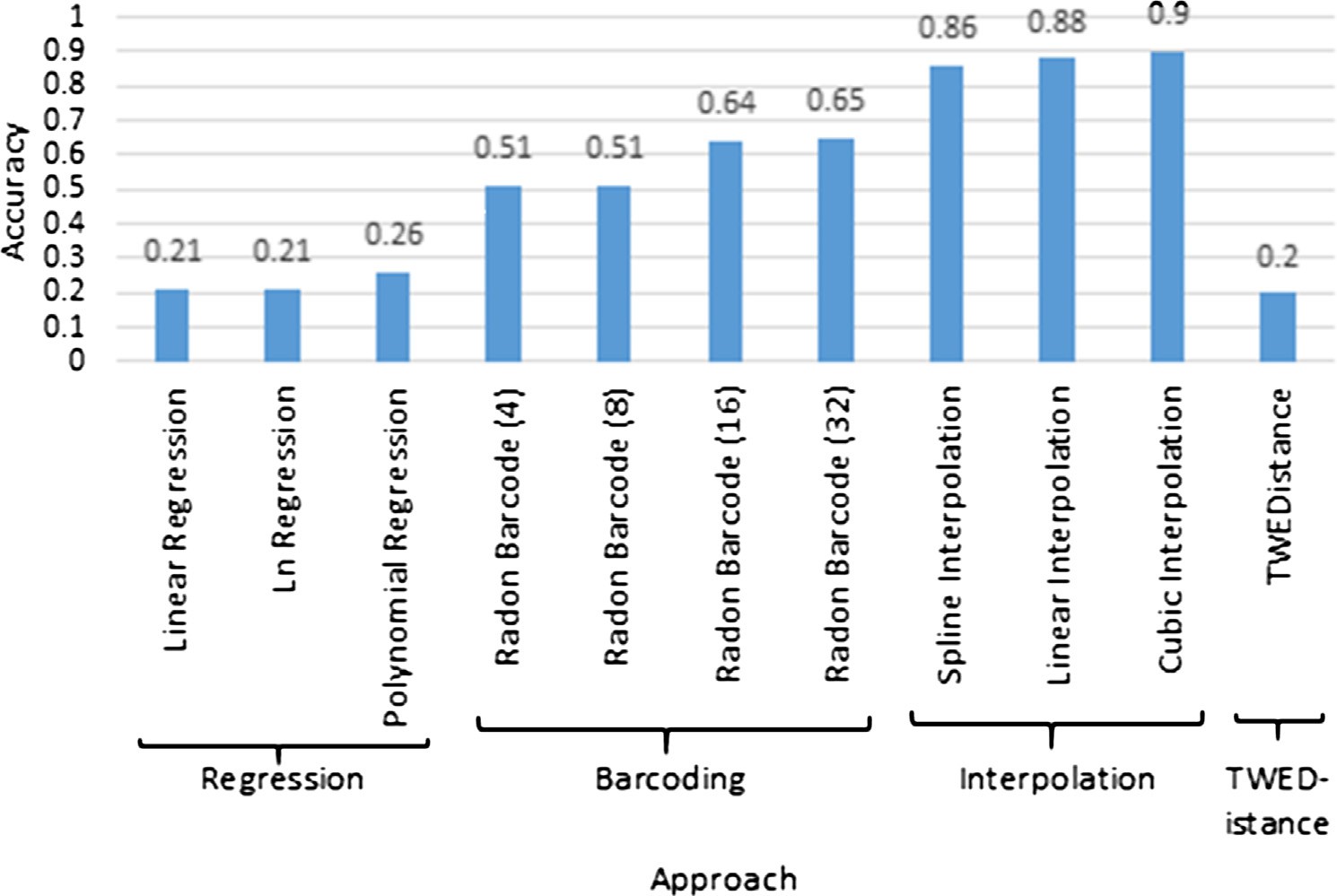


Fig. 16. Average accuracy.

Table 5

Complexity in big O notation.

Approach Complexity Description

TWED O(m ⁄ n) m and n are the length of the two trajectories Barcoding O(l) l is the length of the trajectories’ barcodes Regression O(t3) t is the size of the matrix to be solved

Interpolation O(n ⁄ m) m and n are the length of the two trajectories

* 1. *Approaches complexity*

In this section, complexities of the four approaches are pre- sented in a tabular format in a Big O notation as shown in [Table 5](#_bookmark43). The complexity of TWED is O(n⁄m) as based on its previously mentioned algorithm there is a nested loop over the two trajecto- ries with different lengths (m and n) to compute the distance. The barcoding compare barcodes of the same length (l), so there is only one loop over the two trajectories’ barcodes and compare points at the same index to compute the hamming distance. There for the barcode complexity is O(l). Interpolation evaluates the distance over different trajectories length (n and m), it interpolates one tra- jectory’s points into the approximated functions of the other tra- jectory to evaluate the distance as explained in the interpolation section. Therefore, interpolation complexity is O(n ⁄ m). Regression complexity is the complexity of solving the matrix which is O(t3).

* 1. Conclusion and future work

Similarity evaluation between moving objects’ trajectories is a very important aspect for many application domains. There are variety of similarity measures that evaluate similarity based on the discrete form of a trajectory. These measures are affected by the sampling rate differences. We used different approaches that build up the continuous form of a trajectory’s data (Interpolation, Regression and Curve barcoding) and compared them with a dis- crete similarity based operator which is TWEDistance. We evalu- ated the four approaches on the trains synthetic dataset, the trucks and Geo-life real datasets. We measured their accuracy over

the datasets based on spatial, direction and speed similarity. After evaluating the accuracy per each experiment, we evaluated the average accuracy and the results show that interpolation yields the highest accuracy results against the other approaches with an average accuracy up to 90%. For a future work, there is a need for an auto feature prediction generic continuous similarity opera- tor that predicts what features to be used as a similarity criteria based on the input data.

References

1. [Sakr MA, Güting RH. Group spatiotemporal pattern queries. GeoInformatica](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0005) [2014;18(4):699–746](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0005).
2. [Vlachos M, Gunopulos D, Das G. Rotation invariant distance measures for](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0010) [trajectories. In: Proceedings of the tenth ACM SIGKDD international conference](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0010) [on knowledge discovery and data mining. ACM; 2004. p. 707–12](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0010).
3. [Magdy N, Sakr MA, Mostafa T, El-Bahnasy K. Review on trajectory similarity](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0015) [measures. In: 2015 IEEE seventh international conference on intelligent](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0015) [computing and information systems (ICICIS). IEEE; 2015. p. 613–9](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0015).
4. [Wang H, Su H, Zheng K, Sadiq S, Zhou X. An effectiveness study on trajectory](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0020) [similarity measures. Proceedings of the twenty-fourth Australasian database](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0020) [conference, vol. 137. Australian Computer Society, Inc.; 2013. p. 13–22](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0020).
5. [Magdy N, Sakr MA, El-Bahnasy K. A generic trajectory similarity operator in](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0025) [moving object databases. Egypt Inform J 2017;18(1):29–37](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0025).
6. [Marteau P-F. Time warp edit distance with stiffness adjustment for time series](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0030) [matching. IEEE Trans Pattern Anal Mach Intell 2009;31(2):306–18](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0030).
7. [Ranacher P, Tzavella K. How to compare movement? A review of physical](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0035) [movement similarity measures in geographic information science and beyond.](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0035) [Cartogr Geogr Inform Sci 2014;41(3):286–307](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0035).
8. [Faloutsos C, Ranganathan M, Manolopoulos Y. Fast subsequence matching in](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0040) [time-series databases, vol. 23(2). ACM; 1994](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0040).
9. [Chen Y, Nascimento MA, Ooi BC, Tung AK. Spade: on shape-based pattern](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0045) [detection in streaming time series. In: 2007 IEEE 23rd international conference](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0045) [on data engineering. IEEE; 2007. p. 786–95](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0045).
10. [Nakamura T, Taki K, Nomiya H, Seki K, Uehara K. A shape-based similarity](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0050) [measure for time series data with ensemble learning. Pattern Anal Appl](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0050) [2013;16(4):535–48](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0050).
11. [Alt H. The computational geometry of comparing shapes. In: Albers S, Alt H,](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0055) [Nher S, editors. Efficient algorithms. Springer; 2009. p. 235–48](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0055).
12. [Alt H, Behrends B, Blömer J. Approximate matching of polygonal shapes. Ann](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0060) [Math Artif Intell 1995;13(3–4):251–65](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0060).
13. [Chen L, Özsu MT, Oria V. Symbolic representation and retrieval of moving](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0065) [object trajectories. In: Proceedings of the 6th ACM SIGMM international](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0065) [workshop on multimedia information retrieval. ACM; 2004. p. 227–34](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0065).
14. [Li JZ, Ozsu MT, Szafron D. Modeling of moving objects in a video database. In:](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0070) [Proceedings, IEEE international conference on multimedia computing and](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0070) [systems’ 97. IEEE; 1997. p. 336–43](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0070).
15. [Keogh E, Ratanamahatana CA. Exact indexing of dynamic time warping. Knowl](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0075) [Inform Syst 2005;7(3):358–86](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0075).
16. [Chen L, Ng R. On the marriage of lp-norms and edit distance. Proceedings of](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0080) [the thirtieth international conference on very large data bases, vol. 30. VLDB](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0080) [Endowment; 2004. p. 792–803](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0080).
17. [Vlachos M, Kollios G, Gunopulos D. Discovering similar multidimensional](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0085) [trajectories. In: Proceedings. 18th International conference on data](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0085) [engineering, 2002. IEEE; 2002. p. 673–84](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0085).
18. [Chen L, Özsu MT, Oria V. Robust and fast similarity search for moving object](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0090) [trajectories. In: Proceedings of the 2005 ACM SIGMOD international](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0090) [conference on management of data. ACM; 2005. p. 491–502](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0090).
19. [Little JJ, Gu Z. Video retrieval by spatial and temporal structure of trajectories.](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0095) [In: Photonics west 2001-electronic imaging. International Society for Optics](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0095) [and Photonics; 2001. p. 545–52](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0095).
20. [Güting RH, Böhlen MH, Erwig M, Jensen CS, Lorentzos NA, Schneider M, et al. A](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0100) [foundation for representing and querying moving objects. ACM Trans](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0100) [Database Syst (TODS) 2000;25(1):1–42](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0100).
21. [Forlizzi L, Güting RH, Nardelli E, Schneider M. A data model and data](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0105) [structures for moving objects databases, vol. 29(2). ACM; 2000](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0105).
22. Que A. Polynomial regression; 2014. <[http://polynomialregression.drque.](http://polynomialregression.drque.net/math.html) [net/math.html](http://polynomialregression.drque.net/math.html)> [accessed: 2017-01-14].
23. Lutus P. Polynomial regression; 2009. <[http://arachnoid.com/sage/polynomial.](http://arachnoid.com/sage/polynomial.html) [html](http://arachnoid.com/sage/polynomial.html)> [accessed: 2017-01-14].
24. Gaussian elimination; 2016. <[http://pdfonpoint.com/PDF/LEVEL%20200/CSC%](http://pdfonpoint.com/PDF/LEVEL%20200/CSC%20202%20Matlab/NUMERIC%20METHODS/GAUSS-JORDAN/Gauss_Jordan.pdf) [20202%20Matlab/NUMERIC%20METHODS/GAUSS-JORDAN/Gauss\_Jordan.pdf](http://pdfonpoint.com/PDF/LEVEL%20200/CSC%20202%20Matlab/NUMERIC%20METHODS/GAUSS-JORDAN/Gauss_Jordan.pdf)> [accessed: 2017-01-14].
25. Que A. Non-linear functions that can be linearized for least-square regression; 2014. <[http://www.drque.net/index.php?ArticleNumber=2872#\_2872](http://www.drque.net/index.php?ArticleNumber=2872&_2872)>

[accessed: 2017-01-14].

1. Trek S. Transformations to achieve linearity; 2016. <[http://stattrek.com/](http://stattrek.com/regression/linear-transformation.aspx?Tutorial=AP) [regression/linear-transformation.aspx?Tutorial=AP](http://stattrek.com/regression/linear-transformation.aspx?Tutorial=AP)> [accessed: 2017-01-14].
2. Roberts D. Linear regression models; 2016. <[http://mathbits.com/MathBits/](http://mathbits.com/MathBits/TISection/Statistics2/IntroStat2.htm) [TISection/Statistics2/IntroStat2.htm](http://mathbits.com/MathBits/TISection/Statistics2/IntroStat2.htm)> [accessed: 2017-01-14].
3. Matlab. Interpolation; 2016. <<https://www.mathworks.com/moler/interp.pdf>>

[accessed: 2017-01-14].

1. [Fritsch FN, Carlson RE. Monotone piecewise cubic interpolation. SIAM J Numer](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0145) [Anal 1980;17(2):238–46](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0145).
2. Iwashita Y. Opengamma quantitative research: piecewise polynomial interpolations; 2013. <[https://developers.opengamma.com/quantitative-](https://developers.opengamma.com/quantitative-research/Piecewise-Polynomial-Interpolation-OpenGamma.pdf) [research/Piecewise-Polynomial-Interpolation-OpenGamma.pdf](https://developers.opengamma.com/quantitative-research/Piecewise-Polynomial-Interpolation-OpenGamma.pdf)> [accessed: 2017-01-14].
3. [De Boor C, De Boor C, Mathématicien E-U, De Boor C, De Boor C. A practical](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0155) [guide to splines, vol. 27. New York: Springer Verlag; 1978](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0155).
4. Jcooper. Splines; 2016. <[http://www.math.umd.edu/jcooper/amsc460/spline.](http://www.math.umd.edu/jcooper/amsc460/spline.pdf)

[pdf](http://www.math.umd.edu/jcooper/amsc460/spline.pdf)> [accessed: 2017-01-14].

1. [Tizhoosh HR. Barcode annotations for medical image retrieval: a preliminary](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0165) [investigation. In: 2015 IEEE international conference on image processing](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0165) [(ICIP). IEEE; 2015. p. 818–22](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0165).
2. [Han J, Pei J, Kamber M. Data mining: concepts and techniques. Elsevier; 2011](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0170).
3. Güting RH, Behr T, Düntgen C. Secondo: a platform for moving objects database research and for publishing and integrating research implementations. Fernuniv., Fak. für Mathematik u. Informatik; 2010.
4. Pelekis N. Trucks revised; 2012. Available at <[http://chorochronos.datastories.](http://chorochronos.datastories.org/?q=node/10) [org/?q=node/10](http://chorochronos.datastories.org/?q=node/10)> [accessed: 2017-01-14].
5. [Zheng Y, Zhang L, Xie X, Ma W-Y. Mining interesting locations and travel](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0185) [sequences from gps trajectories. In: Proceedings of the 18th international](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0185) [conference on world wide web. ACM; 2009. p. 791–800](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0185).
6. [Zheng Y, Li Q, Chen Y, Xie X, Ma W-Y. Understanding mobility based on gps](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0190) [data. In: Proceedings of the 10th international conference on ubiquitous](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0190) [computing. ACM; 2008. p. 312–21](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0190).
7. [Zheng Y, Xie X, Ma W-Y. Geolife: a collaborative social networking service](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0195) [among user, location and trajectory. IEEE Data Eng Bull 2010;33(2):32–9](http://refhub.elsevier.com/S1110-8665(17)30081-6/h0195).