01 Jan 2006

Generalized Distance Metric as a Robust Similarity Measure for Mobile Object Trajectories

Garima Pathak

Sanjay Kumar Madria
Missouri University of Science and Technology, madrias@mst.edu

Spandan Tiwari

Recommended Citation

G. Pathak et al., "Generalized Distance Metric as a Robust Similarity Measure for Mobile Object Trajectories," *Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06)*, Institute of Electrical and Electronics Engineers (IEEE), Jan 2006.
The definitive version is available at https://doi.org/10.1109/SUTC.2006.1636172

Generalized Distance Metric as a Robust Similarity Measure for Mobile Object Trajectories

Garima Pathak, Sanjay Madria
Department of Computer Science
University Of Missouri-Rolla
Missouri-65401, USA
{madrias}@umr.edu

Spandan Tiwari
Department of Computer Engineering
University Of Missouri-Rolla
Missouri-65401, USA
{stfff}@umr.edu

Abstract

In this paper, we propose a novel generalized distance metric based on a model that incorporates the time axis explicitly. The proposed metric is based fundamentally on the Mahalanobis distance metric, which eliminates the correlation and scaling errors in similarity searches on trajectory databases. We propose the incorporation of a weight matrix in the proposed distance metric, which allows for easy manipulation of the degree of significance of the different spatial and or temporal dimensions.

1. Introduction

In mobile computing, users mobile devices move in space and register their location at different time instances to spatio-temporal databases via a wireless link stored in the form of trajectories. A trajectory of a moving object [VG02, WX98], usually consists of series of vector values in a multidimensional Euclidean space. There has been an increasing interest in data analysis techniques for the extraction of interesting trajectory patterns. For example, by analyzing trajectories of users who travel in taxicabs, one can find similar important routes or paths being taken by other taxicabs to places, restaurants, shopping malls and other facilities that are frequently visited by people or tourists. Taxicabs or travel agents can use the results of this analysis to provide a higher number of taxies in those paths that are frequently visited.

An efficient way to retrieve interesting information from data repositories with certain motion patterns are similarity and distance-based queries. Thus, the problem becomes that of finding a metric, which can quantitatively describe the degree of similarity between two trajectories. There have been many distance metrics proposed as a measure of similarity between moving object trajectories [VG02,YA03]. Most of the techniques based on distance metrics embed an n-point trajectory in n-dimensional Euclidean space [YA03] and use L_p-norm or its modifications as the similarity measure.

It has been found that there is a high degree of correlation amongst the various dimensions of the resultant data when we map the trajectory data to Euclidean space. This high degree of correlation inherent due to characteristics of the spatial data results in erroneous clustering of similar trajectories (explained in section 3). There is also the common problem of irregular scaling of various dimensions of the data in analyzing due to the disparity of sources from where the data is acquired. It has also been found that most of these metrics are typically suited to a particular application and perform poorly in terms of accuracy for other applications. One common problem is that of incapacity of these metrics in assigning different degrees of significances to various dimensions as per the demands of the application. For example, consider the case of the following similarity based query. Suppose a travel agency wants to find out all similar paths taken by cabs which passed within a radius of one mile along the north or south (single dimension) direction of a hotel. The conventional distance metric will result in equal distance value (and hence similarity value) to objects within one mile radius in east and west direction too since it gives equal significance to all dimensions, which is not required by the application. This drawback is the manifestation of giving equal significance to all dimensions by most of the distance metrics proposed in the past. Our explicit incorporation of the time axis along with weighted significance of various dimensions alleviates this problem and increases the suitability of the proposed distance metric to a variety of applications.

In this paper, we design and experiments with a different technique to retrieve a trajectory most similar to the query trajectory. The method uses the proposed Mahalanobis distance based metric and produces exact similarity search results. Our metric is used for finding distances between trajectories of same lengths. The result shows that the proposed metric performs better than all other distance metrics.

2. Related Work

The simplest approach to defining a similarity measure between two trajectories is to define the n-point trajectory to be a vector in the n-dimensional Euclidean

Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06)
0-7695-2553-9/06 $20.00 © 2006 IEEE

space and define the L_p norm to be the similarity measure. For n-dimensional vectors a and b, the L_p-norm distance is defined as:-

$$L_p(a,b) = \left(\sum_{i=1}^{n} |a_i - b_i|^p \right)^{1/p} \quad (1)$$

It is evident that for p=1, L_p norm is the Manhattan or city block distance and for p=2, it is the simple Euclidean distance between the vectors. Also for $p=\infty$, L_p norm is:

$$L_P(a,b) = \max_i |a_i - b_i| \quad (2)$$

Techniques based on these types of metrics allow efficient indexing by a dimensionality reduction technique [GI99] and were originally proposed for scalar valued series. It was extended in [YA03] for vector-valued series. They tackle the problem of vector of vectors by defining the distance metric as Euclidean distance of L_2-norm between individual points. Additionally, there have been many useful extensions proposed to these Euclidean distance based metrics that render them insensitive to various linear transformations like translation, scaling, normalization and moving average [RM99]. Time warping is another technique that was first used in speech recognition for the matching of signals [SC78]. In the field of data mining, by allowing stretching in time, the same technique was used by Berndt and Clifford [BC94] to get a better distance measure for measuring similarity of time series. Dynamic time allows acceleration-deceleration of signals along the time dimension. The basic idea in Dynamic time warping considers two sequences $x = x_1, x_2, ...x_n$, and $y = y_1, y_2, ...y_n$ and extends each sequence in time by repeating elements. The Euclidean distance is then calculated between the extended sequences x' and y'. There were various restrictions to the given theory. Recent results in [PC00] scale up the dynamic time warping techniques for efficient searching. To find longest common subsequence (LCS) of two sequences [AL95] and then define the distance using length of this subsequence is a similar kind of technique. Dynamic Time wrapping has to pair all elements of the sequences. There has been similar work done on finding similar time series and matching sequences of different lengths [BY97] too. There are various other techniques that are based on extracting certain peculiar features from each time series and using them to define time-based similarity [FJ97,R99]. Benetis et al. [BJ02] proposed Nearest Neighbor and Reverse Nearest Neighbor search algorithm for moving query as well as data points. But they all employ the Euclidean distance between a query and a given point as the measure of closeness between the two.

3. Problem Overview

There have been some models proposed for defining trajectories that take the time values as an additional co-ordinate to the spatial co-ordinates. But it has been found that most distance metrics do not explicitly use the time value at a given instance as another coordinate in the position vector of the object. For example, in [VG02] although time has been taken as a third coordinate along with two spatio-temporal dimensions in the position vector, the distance measure doesn't include the time coordinate in its computation. Instead, time is used as a secondary field in algorithmic searches like "Find trajectory for which t_i>10:00am and t_f<11:00 am" and so on. This not only provides a semi-formal structure but also prevents us from manipulating various dimensions of the trajectory data based on the demands of the application.

Another problem is that of errors in accuracy of nearest neighbor clustering due to correlation and scaling of various dimensions of the data. The assumption of the Euclidean distance is that the dimensions are taken as independent, i.e. there is no influence among the components. This assumption reflects that there is no correlation between features. For example, consider the simple case of a trajectory space for all two-point trajectories. The trajectories are being plotted on a 2D trajectory space where x point of a trajectory is being mapped to x-axis of trajectory space and y point of trajectory is being mapped to y-axis of trajectory space. Let us say that the database has stored trajectories, which are clustered in two main clusters manifesting themselves in the trajectory space as shown in figure 1. There are various methods like spatio temporal range query or a spatial temporal nearest neighbor query to retrieve objects in a mobile object data environment. The queries are defined as the distance between the trajectory of a mobile object and an indicated point in a space. These distance-based queries are very useful in location management of mobile objects; however, queries like range and nearest neighbor do not have sufficient power to analyze the pattern of the object's motion.

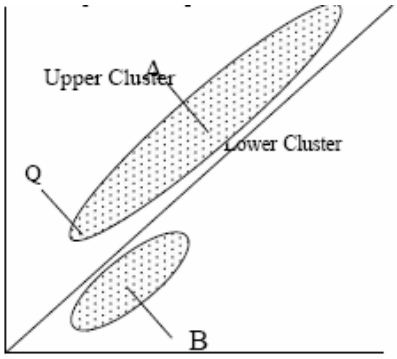

Figure 1. Two clusters of trajectories in the space

For the extraction of individual moving patterns of each object from the trajectories we need to develop tools which can analyze trajectories that follow certain

patterns. Example of a similarity based query is, "Retrieve all objects that follow a path similar to the user who walked in a building." We now quote an example to highlight a few major issues of concern with conventional distance metrics. Figure 1 has many trajectory points forming ellipsoidal clusters. Now let us do a similarity based query for the search query trajectory Q. A can be thought of as the prototype trajectory for the upper cluster and B for the lower one. As they manifest themselves in the trajectory space, it is quite clear that Q and A are from the same cluster and hence are more similar to each other than Q and B, which are not from the same cluster. But the Euclidean distance between Q and B is smaller than Q and A and hence the metrics based on Euclidean distance will reflect greater similarity between Q and B than between Q and A, which will give erroneous result for the similarity search. This kind of error is generated due to the tilted elliptical shape of the cluster, which can be attributed to high correlation amongst the two dimensions and unequal variance of the data along the two principal axes. This correlation is particularly high in spatial data, as objects don't change their positions very rapidly in one time instance to another.

The group of trajectory points forms a tilted cluster shape due to positive correlation or negative correlation. A positive correlation is defined as a relation where the values of two variables increase or decrease together. The tilted structure in figure 1 has slope of the line positive for small values of *X* corresponding to small values of *Y*; large values of *X* correspond to large values of *Y*, so there is a positive "co-relation" (that is, a positive correlation) between *X* and *Y*. A negative correlation is a pattern formed from data points where the values of one variable increase as the values of the other variable decrease, correlation in which large values of one variable are associated with small values of the other, where the slope of the line is negative for small values of X correspond to large values of Y or vise versa. Also the Euclidean metric is sensitive to scaling of co-ordinates. The lack of scale invariance suggests that the trajectory data be standardized by dividing each data point by the standard deviation of the coordinate to which that data point belongs.

Another problem faced by the conventional Euclidean distance based metrics is that every dimension contributes equally towards the final value of the metric, which might not be required by the application. Consider queries like "Find all taxicabs having trajectories similar to the trajectory path being followed by taxicab passing Madison Avenue" or "Find all the football players with trajectories similar to the trajectory of the football player taking the ball at this instance".

Suppose that the unit for the time axis is hour (hr) and for the spatial co-ordinates it is meter. Say Madison avenue is taken as a trajectory T (shown in red color) and is fed into the system as query trajectory, T_a (shown in blue color) is a trajectory which is exactly similar in shape to the query trajectory (T) but is unit time, as shown in figure 2(b). Consider another trajectory T_b (shown in blue color) that is exactly similar in shape but is displaced by 1 meter on the x-axis in space, on all points, as shown in figure 2(a).

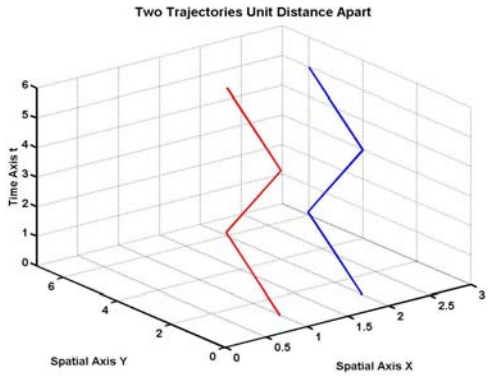

Fig 2(a). Two trajectories unit distance apart in space

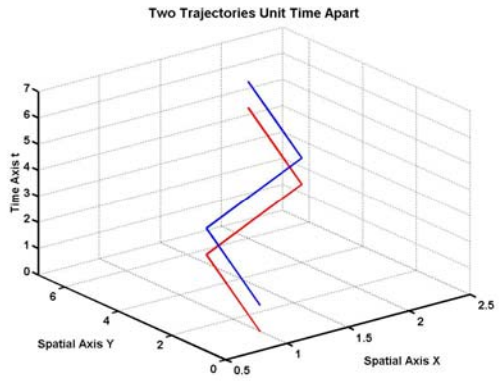

Fig. 2(b). Two trajectories a unit apart in time

Now, if we find the Euclidean distance between T and T_a and that between T and T_b, it will come out to be equal. Suppose an application desires to find the taxicabs closest to Madison Avenue at a given instance of time. Traditional metrics would not resolve the purpose for this application as they will give the unit time (hr) apart trajectory to have same degree of similarity as unit distance (meters) apart trajectory, whereas the application requires the latter as the O/P result because the former is way back in time. Therefore for this particular application there is a need to magnify the contribution of the difference in time to the value of distance between two trajectories more than the contribution of the difference in spatial axes. That is the distance between T and T_a needs to be smaller than between T and T_b. We therefore somehow have to magnify the contribution of the time difference to the final distance value more than the difference on the spatial axes.

4. Proposed Model

In this section, we present some important definitions and the proposed distance metric for similarity search.

4.1 Definitions

A trajectory of a mobile object is actually a continuous line in a 2-D real space R^2 and is in a closed time interval I = [t, t']. In most of the devices, the time at which the object was at various locations is also recorded. We incorporate this time value as the third coordinate with this continuous line.

Definition 1: *Trajectory*- A trajectory Λ is the image of a continuous mapping Λ: I→ R^3.

Here, we have included time as the third co-ordinate in each data point along with the spatial locations, which is an extension of the definition in [RM00].

Definition 2: *Discrete Trajectory*- A discrete trajectory is the image of a discrete mapping $\Lambda : T_\Lambda \rightarrow R^3$.

A discrete trajectory can be represented as a vector sequence $\lambda = \{\overrightarrow{X_1}, \overrightarrow{X_2}, \ldots\ldots \overrightarrow{X_n}\}$ where every point in the sequence itself is a 3-D vector $\overrightarrow{X_i} = (x_i, y_i, t_i)$. We also denote the number of vectors involved in a discrete trajectory Λ as | Λ | called the cardinality of the trajectory Λ.The distance metric proposed in [YA03] is defined at two levels. They first define the distance between two 2-D spatial points as the Euclidean distance between them and then define the distance metric between two trajectories (string of 2-D spatial points) X and Y as: $X = \{\overrightarrow{X_1}, \overrightarrow{X_2}, \ldots\ldots \overrightarrow{X_n}\}$ $Y = \{\overrightarrow{Y_1}, \overrightarrow{Y_2}, \ldots\ldots \overrightarrow{Y_n}\}$

$$D_{XY} = \sqrt{d(\overrightarrow{X_1}, \overrightarrow{Y_1})^2 + d(\overrightarrow{X_2}, \overrightarrow{Y_2})^2 + d(\overrightarrow{X_3}, \overrightarrow{Y_3})^2 \ldots\ldots\ldots\ldots d(\overrightarrow{X_n}, \overrightarrow{Y_n})^2} \quad (3)$$

where $d(\overrightarrow{X_j}, \overrightarrow{Y_j}) = \sqrt{(X_{jx} - Y_{jx})^2 + (X_{jy} - Y_{jy})^2}$

We, like [Y03], present the distance metric between two points in the trajectories, which is a Mahalanobis distance between two vectors with weighted dimensions. We then go on to extend the distance metric for a string of vectors i.e. trajectories, in the Mahalanobis framework. We also note that although we have not extended our metric for finding distances between trajectories of different lengths, there are several techniques reported in literature [AL95, BC94] for overcoming such a problem.

4.2 Distance between two vector points on trajectories

Let the set of all the vector points (constituting all the trajectories) in the database be S. Let C be the covariance matrix of the dataset S.

$$C = \underset{S}{E}\left[(\vec{x} - \overrightarrow{m_x})^T (\vec{x} - \overrightarrow{m_x})\right] \quad (4)$$

where E [.] denotes the expectation operator over a given dataset (in this case, S) and m_x is the mean value. Let $\overrightarrow{X_1} = (x_1, y_1, t_1)$ and $\overrightarrow{X_2} = (x_2, y_2, t_2)$ be two vectors in the dataset S. Then the Mahalanobis distance between the two vector points $\overrightarrow{X_1}$ and $\overrightarrow{X_2}$ is given as: $D_m(\overrightarrow{X_1}, \overrightarrow{X_2}) = (\overrightarrow{X_1} - \overrightarrow{X_2})C^{-1}(\overrightarrow{X_1} - \overrightarrow{X_2})^T$ (5)

If the various coordinates of the data points, namely x, y and t are independent then the covariance matrix reduces to only a diagonal matrix with the elements of the diagonal as the variance of the data along the axes x, y, t . That is:

$$C = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_t^2 \end{bmatrix} \rightarrow C^{-1} = \begin{bmatrix} 1/\sigma_x^2 & 0 & 0 \\ 0 & 1/\sigma_y^2 & 0 \\ 0 & 0 & 1/\sigma_t^2 \end{bmatrix}$$

where, $\sigma_x^2, \sigma_y^2, \sigma_t^2$ are the variance of spatial coordinates x, y and time co-ordinate t, respectively.

Typically for an application, the contribution of the spatial axes towards the notion of similarity (or dissimilarity) is different to that of the temporal axis. In general, the contribution can be dissimilar even for the various spatial axes too. It is thus required that provisions for incorporating different levels of significance to various spatial and time axes be given in the distance metric. For the ease of customization of the metric to various applications it also becomes imperative that this should be done with the change in a few parameters and not in the general Mahalanobis framework. We thus modify the Mahalanobis distance metric between two vectors to incorporate a weight matrix as follows:

$$D_g^{\overrightarrow{X_1}\overrightarrow{X_2}} = (\overrightarrow{X_1} - \overrightarrow{X_2})WC^{-1}W^T(\overrightarrow{X_1} - \overrightarrow{X_2})^T \quad (6)$$

where W= $\begin{bmatrix} \sqrt{f_x(x_1, x_2)} & 0 & 0 \\ 0 & \sqrt{f_y(y_1, y_2)} & 0 \\ 0 & 0 & \sqrt{f_t(t_1, t_2)} \end{bmatrix}$

is the weight matrix and f_x, f_y, f_t are the weighting functions for the three axes. Let's take a simplistic case where the covariance matrix C is a diagonal matrix as defined above, though in most of the real-world applications C will not be a diagonal matrix. Also, let the weighting functions in the matrix W be constant functions as shown below: $W = \begin{bmatrix} \sqrt{k_x} & 0 & 0 \\ 0 & \sqrt{k_y} & 0 \\ 0 & 0 & \sqrt{k_t} \end{bmatrix}$

Then the distance between two vectors $\overrightarrow{X_1}$ and $\overrightarrow{X_2}$ reduces to:

$$D_g^{\overrightarrow{X_1}\overrightarrow{X_2}} = \sqrt{\frac{k_x(x_1 - x_2)^2}{\sigma_x^2} + \frac{k_y(y_1 - y_2)^2}{\sigma_y^2} + \frac{k_t(t_1 - t_2)^2}{\sigma_t^2}} \quad (7)$$

The above metric normalizes the variances of the various axes to unity thus eliminating the errors due to irregular scaling as discussed in section 3. Also, the weighting constants govern the contribution of the various dimensions to the final distance. Different

weighting functions for different applications and their effects are discussed in section 5.

4.3 Distance between two trajectories

Now we define our distance metric D between two n-point trajectories Λ, Λ' where each point is a 3-D vector. The trajectories are:

$$\Lambda = \left\{\overrightarrow{X_1}, \overrightarrow{X}_{2,}\text{----}, \overrightarrow{X}_n\right\} \text{and } \Lambda' = \left\{\overrightarrow{X'_1}, \overrightarrow{X'}_{2,}\text{----}, \overrightarrow{X'_n}\right\}.$$

The distance metric is then given as:

$$D^{\Lambda\Lambda'} = dQ^{-1}d^T \qquad (8)$$

where

$$\vec{d} = \left[D_g^{\overrightarrow{X_1}\overrightarrow{X_1}'}, D_g^{\overrightarrow{X_2}\overrightarrow{X2}'} \text{.....} D_g^{\overrightarrow{X_n}\overrightarrow{X_n}'} \right] \text{where}$$

$D_g^{XX'}$ is the generalized Mahalanobis distance between two vectors X, X' described in section 4.2. And Q is given as:

$$Q = \begin{bmatrix} \sigma_1^2 & 0 & 0\text{……..0} \\ 0 & \sigma_2^2 & 0\text{…..……..0} \\ 0 & 0 & \sigma_3^2\text{.....0} \\ 0 & 0 & 0\text{.....}\sigma_n^2 \end{bmatrix}$$

where σ_i^2 is the standard deviation of the data of values of the distance D_g amongst the i^{th} vector points of all possible pairs of trajectories.

The matrix Q normalizes the various axes of the vector "d" based on their respective variances. This normalization prevents the distance between two trajectories to be high if there's a large distance between single or a few points on the two trajectories. Those high distances alone will not govern the final distance between the trajectories but the final distance value would truly be a measure of the similarity between the two trajectories as wholes. Thus, in a sense this normalization makes the metric immune to outliers. We can consider the above distance metric to be the Mahalanobis distance of the vector "d" from the origin with the cross-correlation terms zero. Those terms would signify the correlation between the distances between i^{th} and j^{th} points on all the possible pairs of trajectories (where $i \neq j$) in the trajectory database. Since our metric is based only on point to point (i^{th} to i^{th} point) distances between the various points of the trajectories, we can safely assume these cross-correlation terms in Q to be zero. Though it is not necessary, but would bias the distance metric erroneously, if they are not. Also, this assumption reduces the cost of calculating Q from prohibitively high otherwise, to reasonably manageable.

5. Discussion

In this section, we present the analysis of the proposed distance metric and strengthen our argument in its support by quoting some real world examples. Given the nature of the spatio-temporal data, there will always be high correlation between the spatial dimensions. This correlation will be markedly higher in case of slow moving trajectories but small for agile (fast direction changing) object trajectories and can be theoretically zero only for objects executing Brownian motion. It could, however, be that correlation between the spatial and temporal dimensions is small and can be neglected if found to be too small as it could just be the result of the inaccuracy in the estimation of the covariance matrix C, due to finite data.

We are essentially capturing the distances between various trajectories and hence, correction of correlation and scaling errors will certainly give a higher degree of accuracy to the similarity search results. For example, consider three points A, B, C in the Euclidean space with A={1,0,1}, B={0,1,1}, C={1,1,0} (corresponding to three trajectories). Euclidean distances between all three points are $d_{ab} = d_{bc} = d_{ca} = \sqrt{2}$ .The points are plotted in figure 3. Note that in the actual trajectory, there will be a vector of points. Suppose we want that A and B should appear closer than BC or AC (since A and B are in the same temporal plane). We then have to reduce the significance of time axis by setting $w_t > 1$, say $w_t = \sqrt{2}$.

We assume x, y, t to be mutually independent having unity standard deviation each i.e.

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ \& } W = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \sqrt{2} \end{bmatrix}$$

Then using our proposed distance metric:-

$$D_g^{AB} = (X_A - X_B)WC^{-1}W^T(X_A - X_B)^T \qquad (10)$$

$$D_g^{AB} = \sqrt{1(X_a - X_b)^2 + 1(Y_a - Y_b)^2 + 2(t_a - t_b)^2} = \sqrt{2}$$

Where as, $D_g^{BC} = D_g^{CA} = \sqrt{3}$, which is required by the demands of the application. Similarly we can vary the significance of the various dimensions to suit the need of the application. There can be applications in which we might need the weight functions to be other than a constant. Consider the following query."*Find all similar trajectory paths taken by taxi cabs near Plaza hotel at 10:00 am*". We thus want all the objects, which are not only near to Plaza hotel spatially but also in time. Let the query be T (consider figure 2(a)). Consider a taxicab trajectory 1 meter apart in space and parallel to T, says T_A. Also another taxicab trajectory, 1 minute apart (behind) in time from T, say T_B(figure 2(b)). Now consider another similar taxicab trajectory, 60 meters apart in space from T, say T_C ,and finally another trajectory, 60 minutes apart (behind) in time from T, say T_D. We want that in a "k" most similarity based search we should get T_A, T_B and T_C as output. If we take C as the identity matrix shown above and the weighting functions in W to be constants as below:

$W = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix}$ then the distances between various trajectories are:

$$D_g^{TT_A} = \sqrt{2}\ ,\ D_g^{TT_B} = \sqrt{1}\ ,\ D_g^{TT_C} = 60\sqrt{2}\ ,\ D_g^{TT_D} = 60$$

It can be understood that $D_g^{T\,TB}$ should be small as 1-minute difference is not as significant as 1-meter difference, which is achieved by the constant weights in W. But it is also desired that $D_g^{TT_D}$ should be much higher than $D_g^{TT_C}$ as 60 minutes difference in time is considerably huge (because a taxi might be anywhere in 60 minutes) than 60 meters difference in space.

In essence, we want that trajectories with huge distances in time from query trajectory should have very large values of distance metrics in comparison to values of same value of differences in spatial axis, but trajectories with small differences in time should have values of distance from the query trajectory, almost similar to the values of distance metrics for equal value of differences on the spatial axes. We can thus choose $f_t(t,t')$ to be $f(t,t') = (t-t')^2$ where $W = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & (t-t')^2 \end{bmatrix}$ Going back to same example above and finding the distance metrics again: -

$$D_g^{TT_A} = \sqrt{(X_t - X_{tA})^2 + (Y_t - Y_{tB})^2 + (t-t_{tA})^2(t-t_{tA})^2} = \sqrt{2} \quad (11)$$

$$D_g^{TT_B} = \sqrt{1}\ ,\ D_g^{TT_C} = 60\sqrt{2}\ ,\ D_g^{TT_D} = 60^2 = 360$$

Thus, D_g between T and T_D is much larger than previous value and also $D_g^{TT_C}$. This T_C will appear more similar to T than T_D, which is exactly what is required by the application.

A weighting function for a dimension maps the difference in the values of the dimension between two points to the desired weight value, which reflects the significance for that dimension (for that given value of difference). A weighting function for a given dimension governs the weight assigned for the values of the dissimilarity between two points on that dimension. A typical application for moving objects will not require different weighting functions for X and Y spatial dimensions, unless there is a directional query like in the example given in section 1 for similarity query along North-South or East-West direction. But typically temporal dimensions are given different weighting functions (even if constant function) than spatial based on the needs of the application as shown in the example in the last section. If however we wish to discriminate between smaller and larger differences on a dimension then we make the weighting functions a function of the difference. For discriminating between the difference values of different dimensions, we need to have their weighting functions with different slopes.

A specific case was quoted in the example in the last section where the difference between a constant and the time difference dependent function was brought up. For small values of time differences (note that time difference cannot be less than 1 unit) the weighting function results in same weights as for spatial dimensions but for large differences it results in much higher values due to the non linear weight function. We can thus design weighting functions for various axes taking into account what significance needs to be given to a dimension at a given value of difference between two vectors values for that dimension. Though, we understand that most applications typically not require any other weighting functions besides the constant and the square of difference weighting functions.

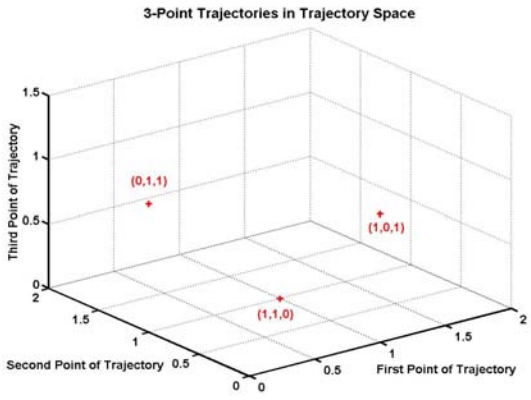

Figure 3. Trajectories A, B, C in trajectory space

Coming to the formulation of the distance metric as a similarity measure for two trajectories; the normalization of the distances between various points based on their variances in the trajectory distance calculation also produces more meaningful values of the distances for the similarity searches. The calculation of the covariance matrix C (section 4) requires some intensive processing. but assuming that we have a reasonable large database of the trajectories, we have a reasonable unbiased estimate of the co-variance matrix. Thus, it is not imperative that we recalculate the entire covariance matrix every time a new trajectory is appended in the database. In fact, going by the thumb rule of statistical analysis, for an n-point trajectory similarity search, we need to recalculate the covariance matrix only after 5n new trajectories have been appended and we can still safely assume our covariance matrix to be the representative of the current data. The calculation of the covariance matrix is entirely independent of the similarity search process and can be executed in parallel.

6. Experimental Evaluation

We used a database of 10000 trajectories, which is generated using the City Simulator software offered by

IBM Alpha Works. We have implemented various popular distance metrics to draw the comparative results on the accuracy of the various distance metrics to find the most similar trajectory. Among these, we have implemented the Euclidean distance metric [YA03], Manhattan distance metric and also the L-∞ norm based distance metric. We have extended the Manhattan distance metric and L-∞ norm from scalar-valued series to vector-valued series for incorporating trajectory data. In these results we have taken the weight matrix as identity matrix i.e. weights of both the spatial dimensions and the time dimensions are equal. This weight matrix is used for calculating Mahalanobis distance metric with the aim of highlighting the improvement in the accuracy of similarity search owing to the elimination of scaling and correlation errors. To speed up the calculations, we have used the reduced form of the covariance matrix as discussed in section 4.3 (similar to matrix Q).

Since there is no metric to judge which metric gives the most "similar" trajectory as the result of the similarity search, we have used visual inference for this like used in [VG02]. Since we are dealing with real world trajectories the notion of similarity is based mainly on the distance between a resultant trajectory and the query trajectory and also the similarity in their shapes. The latter contributes less to the final result as the real world trajectories of mobile users have a very small component of Brownian motion and hence don't differ greatly in two consecutive points. Thus, the distances between two trajectories are mainly governed by their average separation in the 3-D space and less by their internal shape distinctions. This however is very specific to slow moving mobile device user trajectories and the factors can vary significantly based on what amount of the movement can be characterized as Brownian motion. In figure 4(a), we show the results of similar trajectory search in the database of 10000 trajectories, using all the four metrics. The query trajectory is shown in red, the result of Mahalanobis distance based metric is shown in blue, the result of Euclidean distance based metric is shown in green, the result of Manhattan distance based metric is shown in cyan and the result of L-∞ distance based metric is shown in magenta color. It can be clearly seen that the Mahalanobis distance based metric gives the output trajectory, which is closest to the query trajectory in the 3-D space-time. To show the result more clearly the 3-D space is shown from the top in figure 4(b). In figure 4(b) it can be seen how far the various results are placed from the actual query trajectory and apparently Mahalanobis distance based metric result is the closest to the query trajectory in space too. Note that in the figure 4 (b) we show only y-axis and time axis to show the similarity (similar notation used in other figures).

In figure 5(a), we depict the results of all the four distance metrics. Again it can be seen that Mahalanobis distance metric outperforms the other three distance metrics. Euclidean and Manhattan distance based metric give the same trajectory and hence they overlap. L-∞ Norm based metric gives a different output trajectory from the other three and, in fact this particular case gives the least similar result. This is evident from figure 5(b), which takes a view in the 3-D space along the time axis.

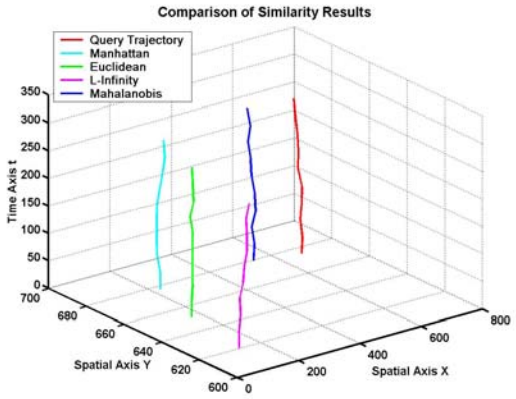

Figure 4 (a). Case 1: Results of various Distance metrics

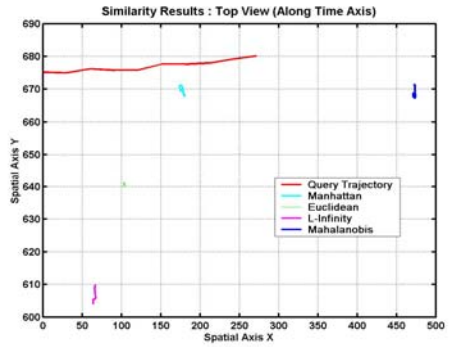

Figure 4 (b). Case 1: Trajectories viewed along the Time axis

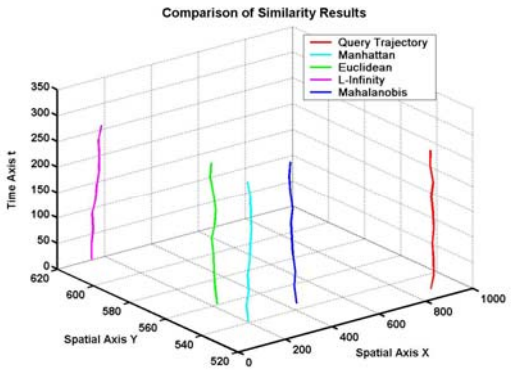

Figure 5 (a). Case 2: Results of various distance metrics

From the results shown in figures 6(a,b), Mahalanobis distance based metric gives the most similar trajectory where as all others give a trajectory different from Mahalanobis metric but the same amongst themselves. The distance between the

Mahalanobis and other metrics trajectory from the query trajectory is significant and shows substantial improvement with the use of Mahalanobis distance based metric. We have compared the performance of the proposed metric with other metrics on a database of 1000 and 10000 and have found that Mahalanobis based metric gives outright better results in most of the cases.

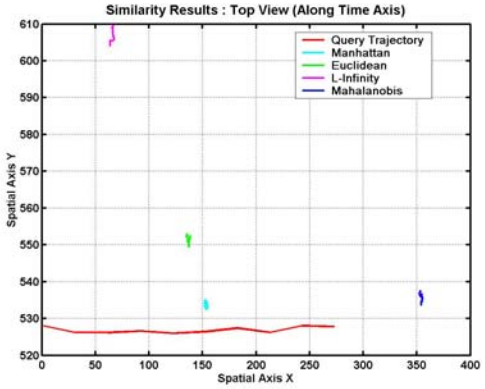

Figure 5 (b). Case 2: Trajectories viewed along the time axis.

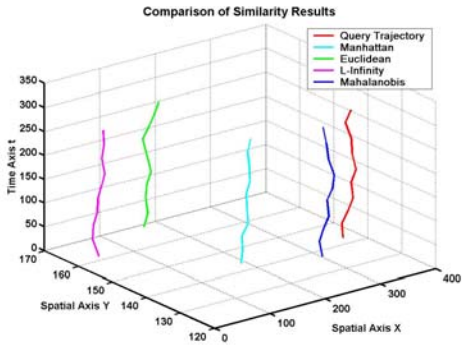

Figure 6 (a). Case 3: Results of various Distance metrics

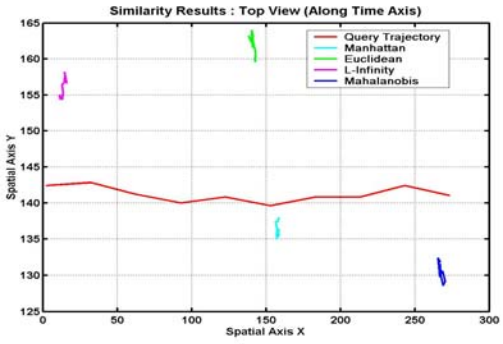

Figure 6 (b). Case 3: Trajectories viewed along the time axis

Comparison Over Queries	Mahalanobis Outright Winner	Mahalanobis Overlapped Winner	Mahalanobis Second to some!
1000	67%	29%	4%
10000	62%	35%	3%

7. Conclusions

In this paper, we have presented a novel generalized distance metric for capturing similarity between trajectories of moving objects including the time aspect as an explicit co-ordinate along with the spatial co-ordinates. The proposed metric being based on Mahalanobis distance metrics, corrects inaccuracies in the similarity searches occurring due to correlation between various spatio-temporal dimensions and their irregular scaling. This renders the proposed metric more customizable for various applications.

Reference

[AL95] R. Agrawal, K. Lin, H. S. Sawhney, and K. Shim, "Fast Similarity Search in the Presence of Noise, Scaling and Translation in Time-Series Databases", *In Proc of VLDB*, pages 490–501, Sept. 1995.

[BJ02] R. Benetis, C. S. Jensen, G. Karciauskas, and S. Saltenis, ``Nearest Neighbor and Reverse Nearest Neighbor Queries for Moving Objects," In *Proceedings of the 2002 International Data Engineering and Applications Symposium*, Edmonton, Canada, July 17-19, 2002, pp. 44-53.

[BC94] D. Berndt and J. Clifford. Using Dynamic Time Warping to Find Patterns in Time Series. *In Proc. of KDD Workshop*, 1994.

[BY97] T. Bozkaya, N. Yazdani, and M. Ozsoyoglu."Matching and Indexing Sequences of Different Lengths.", *In Proc.of the CIKM, Las Vegas*, 1997.

[FJ97] C. Faloutsos, H. Jagadish, A. Mendelzon, and T. Milo. "Signature technique for similarity-based queries", *In SEQUENCES 97*, 1997.

[GI99] A. Gionis, P. Indyk, and R. Motwani. "Similarity search in high dimensions via hashing". *In Proc. of 25th VLDB*, pages 518–529, 1999.

[PC00]S. Park, W. Chu, J. Yoon, and C. Hsu. "Efficient Searches for Similar Subsequences of Different Lengths in Sequence Databases". *In Proceedings of ICDE*, pp. 23–32, 2000.

[R99] D. Rafiei, "On similarity-based queries for time series data". *Proc of the 15th IEEE International Conference on Data Engineering*.,1999, Sydney, Australia.

[RM00] D. Rafiei and A. Mendelzon. "Querying Time Series Data Based on Similarity". *IEEE Transactions on Knowledge and Data Engineering, Vol. 12, No 5.*, pages 675–693, 2000.

[SC78] H. Sakoe and S. Chiba. "Dynamic programming algorithm optimization for spoken word recognition". *IEEE Trans. Acoustics, Speech and Signal Processing*, ASSP-26(1):43– 49, Feb. 1978.

[VG02] M. Vlachos, D. Gunopulos, G. Kollios: "Robust Similarity Measures for Mobile Object Trajectories", *Proc. of MDDS 2002*, p. 721-728, Aix-en-Provence, France.

[YA03] Yutaka Yanagisawa ,Jun-ichi Akahani ,Tetsuji Satoh "Shape-based Similarity Query for Trajectory of Mobile Objects", *4th International Conference on Mobile Data Management*, Jan 2003.

[WX98]O. Wolfson, B. Xu, S. Chamberlain, L. Jiang, "Moving Objects Databases: Issues and Solutions", *Proceedings of the 10th International Conference on Scientific and Statistical Database Management,* Capri, Italy, July 1-3, 1998, pp. 111-122.