A Dynamic Time Warping based Algorithm for Trajectory Matching in LBS

Peng Chen, Junzhong Gu*, Dehui Zhu and Fei Shao

Department of Computer Science and Technology, East China Normal University, 200241 Shanghai, China
*{pchen, dhzhu, fshao}@ica.stc.sh.cn; *jzgu@cs.ecnu.edu.cn*

Abstract

Advances in mobile computing, wireless communication, and positioning technology have flourished Location Based Service (LBS). This results in a large amount of trajectories accumulated in LBS applications. As a fundamental research spot, trajectory matching algorithm has drew much attention from image and transportation communities. While the various location update strategies in LBS make trajectory matching a more challenging task, since the different samples of one trajectory should not be falsely recognized as another trajectory. Besides this, LBS applications also require the trajectory matching algorithm to be sensitive to time sequence and tolerate to time scaling. A dynamic time warping based algorithm is proposed in this paper. Extensive experiments show the effectiveness, time sequence sensitiveness and time scaling toleration of the proposed algorithm.

Keywords: *Trajectory matching; dynamic time warping; location based service*

1. Introduction

With the development of Internet, Geographic Information System (GIS) and mobile technology, Location Based Service (LBS) plays an important role in various applications [1]. In LBS applications, more and more devices are capable of acquiring, processing, and storing location information of moving objects. Thus, huge amount of location information in the form of trajectories are accumulated. The trajectory study is a hot research area in ecology, biology, transportation, geographic information science, as well as social and behavioral science. The research of similarity between trajectories will help understanding the object's movement, as well as prediction, simulation, and knowledge discovery. To be more specific, in Intelligent Transportation Systems (IST), through the analysis of the similarity between trajectories, one can identify a path with a high probability of congestion. This will be used in future road planning or traffic directing. Wildlife scientists want to find out the migration pattern from animal moving trajectories. The migration pattern may further be used in animal protection.

The similarity problem has been studied in research fields such as image, motion tracking in videos and time series analysis. While the diverse definitions of similarity in different research fields make it infeasible to simply adopt existing matching methods into LBS scenarios. The weakness of existing matching algorithms are mainly four fold, (1) sensitive to noise, (2) cannot be performed on sequences with different lengths or sampling rates, (3) not sensitive to time sequence, and (4) not tolerate to time scaling. In this paper, an effective, time sequence sensitive, and time scaling tolerating algorithm based on dynamic time warping is proposed.

The rest of this paper is organized as follows: In Section 2, a brief introduction to related researches is given. In Section 3, the trajectory model, location update strategy, and the goals of trajectory matching algorithms for LBS are presented. The dynamic time warping based algorithm is described in Section 4. Experiment evaluations are reported in Section 5. The last section concludes this paper.

2. Related work

In this section, the classification of similarity is summarized and a brief survey of existing algorithms is given.

2.1. Classification of Similarity Calculation

The similarity calculation can be categorized as either motion based (Motion) or shape based (Shape) according to whether the temporal dimension is taken into consideration or not [2]. Similarity can be further categorized into more groups such as Rotation and Translation Free (RTF), Time Sequence Sensitive (TSS), Temporal Scaling Tolerating (TST) and Spatial Scaling Tolerating (SST).

Rotation and Translation Free (RTF) means when two trajectories are compared, they can be rotated and translated in any way to minimize the distance function [2]. Time Sequence Sensitive (TSS) means when the time sequence of one trajectory is changed to some extent (such as the time dimension is reversed), the new trajectory should be recognized as another trajectory. For Spatial Scaling Tolerating (SST), if the only difference between two samples is the spatial dimensions are amplified or minified to some extent, they should not be seen as different trajectories; For Temporal Scaling Tolerating (TST), if the time interval between the time stamps t_1 and t_2 in sample s_1 is only slightly larger than the time interval between the time stamps t_1' and t_2' in sample s_2, those two samples are from one trajectory.

Table 1 shows similarity calculation requirements of each application.

Table 1. Application Requirements

Applications	Motion	Shape	RTF	TSS	TST	SST
Vehicle	Y	N	N	Y	Y	N
Animal	Y	N	N	Y	Y	N
Motion	Y	N	Y	Y	Y	N
Image	N	Y	Y	N	N	Y
LBS	Y	N	N	Y	Y	N

Here “Bus” represents bus route schedule applications. “Motion” means motion analysis applications (*e.g.*, NBA players’ shouting and dribbling in games). “Animal” are animal tracking applications (*e.g.*, migrant bird tracking). “Image” denotes the image matching applications. Since we focus on the trajectory matching in LBS scenarios, motion, time sequence sensitive, temporal scaling tolerating are the three criteria we emphasized.

2.2. Existing Algorithms

In essence, the trajectory similarity calculation is a multi-dimensional sequence data clustering problem, which is a trajectory matching problem of mobile objects [3]. And HD, MHD, IMHD, and OWD are four well known trajectory matching algorithms. In this section, brief introductions of those four are given.

HD (Hausdorff Distance): HD is a shape comparison metric based on binary image. It is a max-min distance defined between two point sets [4]. In HD, there is no one-to-one correspondence between one point (in the model) and the other (in the test image). Since no explicit point correspondence is calculated in HD, this method is simple yet efficient. If the Hausdorff distance between two point sets A and B is *d*, this means every point in A is within distance *d* of some point in B and vice versa.

$$H(A, B) = \max(h(A, B), h(B, A)) \quad (1)$$

$$h(A, B) = \max_{a_i \in A}(\min_{b_j \in B} d(a_i - b_j)) \quad (2)$$

$$h(B, A) = \max_{b_j \in B}(\min_{a_i \in A} d(b_j - a_i)) \quad (3)$$

MHD (Modified Hausdorff Distance): The Hausdorff Distance is sensitive to outlier points. A few outlier points, even only a single one, can perturb the distance greatly [5]. To alleviate this situation, the Modified Hausdorff distance (MHD) is introduced by Dubuisson and Jain in [6]. MHD achieves a very high performance over images with various levels of noise. Meanwhile, MHD is also robust to outlier points.

$$h(A, B) = \frac{1}{m_a} \sum_{a_i \in A} (\min_{b_j \in B} d(a_i - b_j)) \quad (4)$$

IMHD (Interpolation based Modified Hausdorff Distance): Since various location update strategies make the trajectory matching algorithm a challenging task, an Interpolation based MHD algorithm (IMHD) is introduced in [3]. IMHD aims at reducing the side effect of various location update strategies, sampling granularities, and initial positions. As reported in [3], it has a higher discriminating power than any other Hausdorff Distance based trajectories matching algorithms. The distance function of IMHD is defined as follows:

$$h(A, B) = \frac{1}{m_a} \sum_{a_i \in A} (\min_{\overline{b_{j-1} b_j} \in \overline{B}} d(a_i - \overline{b_{j-1} b_j})) \quad (5)$$

OWD (One Way Distance): OWD is a simple and effective method which compares the spatial shapes of trajectories. The OWD distance between point p and trajectory Tr_i is defined as follows:

$$D_{point}(p, Tr_i) = \min_{q \in Tr} D_{Euclid}(p, q) \quad (6)$$

where $D_{Euclid}(p, q)$ denotes the Euclidean distance between point p and q.

The OWD distance between a trajectory Tr_1 and another trajectory Tr_2 is defined as the integral of the distance from points of Tr_1 to trajectory Tr_2 divided by the length of Tr_1:

$$D_{owd}(Tr_1, Tr_2) = \frac{1}{|Tr_1|} \left(\int_{p \in Tr_1} D_{point}(p, Tr_2) dp \right) \quad (7)$$

The distance between two trajectories Tr_1 and Tr_2 is the average of their one way distances:

$$D(Tr_1, Tr_2) = \frac{1}{2} (D_{owd}(Tr_1, Tr_2) + D_{owd}(Tr_2, Tr_1)) \quad (8)$$

2.3. Summary of Existing Algorithms

After the introductions of both the classification of similarity calculation and existing trajectory matching algorithms, we categorize those algorithms into groups. Table 2 shows the classification of existing algorithms.

Table 2. The Classification of Existing Algorithms

Algorithms	Motion	Shape	RTF	TSS	TST	SST
HD	N	Y	N	N	Y	N
MHD	N	Y	N	N	Y	N
IMHD	Y	N	N	N	Y	N
OWD	Y	N	N	N	Y	N

Since none of HD, MHD, IMHD, OWD can fit in with LBS requirements. A novel trajectory matching algorithm which considers both spatial and temporal dimensions, tolerates to temporal scaling, and be sensitive to time sequence is required.

3. Formal Definition of the Problem

In this section, the frequently used symbols are presented. Then the trajectory model used throughout this paper is introduced. Finally, the goals of trajectory matching algorithms for LBS are given. In Table 3, the symbols used in our discussion are summarized.

Table 3. Frequently used Symbols

Symbol	Description
Tr_i	Trajectory i, with total length $\|Tr_i\|$
S_{ij}	A sample of Tr_i acquired through location update strategy j, with length $\|S_{ij}\|$
$\mathrm{d_k}(Tr_i, Tr_j)$	The distance measured by algorithm k between trajectory i and j.
reverse(S_{ij})	The temporal sequence is reversed.
time_scaling(Tr_i, α)	The spatial dimensions remain unchanged, while the temporal dimension is scaled by α (amplified $\alpha>1$ or minified $\alpha<1$).
time_span(Tr_i)	The total time elapsed of trajectory Tr_i

3.1. Trajectory Model

A trajectory Tr_i is a series of location updates with according time stamps. $Tr_i = \{(p_1, t_1), (p_2, t_2),\ldots,(p_m, t_m)\}$, where p_i is the location recorded at time stamp t_i. A sample S_{ij} is a sample of trajectory Tr_i with location update strategy j. Normally, S_{ij} is a subset of Tr_i.

In this paper, the two location update strategies introduced in [7] are considered, namely time based update strategy and distance based strategy. In time based update strategy, a mobile user updates his location every Δt units of time (*e.g.*, Δt = 3 minutes). Similarly, in distance based strategy, a new location is reported to server when it derived Δd from the last reported location (*e.g.*, Δd = 1 kilometer). When Δt or Δd is approaching 0, the sampling S_{ij} equals to Tr_i.

3.2. The Goals of Trajectory Matching Algorithms for LBS

According to the requirements of real life LBS applications, three goals, effectiveness, time sequence sensitiveness and time scaling toleration, are proposed.

Effectiveness:

d(S_{ij}, S_{ik}) ≈ 0

On one side, the distance between different samples of one trajectory should be as small as possible.

(2) d(Tr_i, Tr_j) >> 0

On the other side, the distance between two different trajectories should be as large as possible.

In experiments, max{d(S_{ij}, S_{ik})} < min{d(Tr_i, Tr_j)} is used.

Time sequence sensitiveness:

(3) d(Tr_i, reverse(Tr_i)) >> 0

When the temporal sequence is reversed, the new trajectory reverse(Tr_i) should be recognized as a different trajectory, d(Tr_i, reverse(Tr_i)) > max{d(S_{ij}, S_{ik})}.

Time scaling toleration:

d(Tr_i, time_scaling(Tr_i, α)) ≈ 0

The distance between Tr_i and time_scaling(Tr_i, α) should be as small as possible. time_scaling(Tr_i, α) means the spatial dimensions of Tr_i remain unchanged, while the temporal dimension is slightly amplified or minified, d(Tr_i, time_scaling(Tr_i, α))< min{d(Tr_i, Tr_j)}.

4. Dynamic Time Warping based Algorithm

Dynamic Time Warping (DTW) [8] has been successfully applied in speech recognition community. It is currently used in gesture recognition, handwriting matching, wildlife monitoring, and *etc*. The introduction of DTW makes it possible to compare sequences with different lengths, with and without time information. Through stretch of sequences, the minimum of distance between sequences is achieved.

For two trajectories $Q= q_1, q_2, \ldots, q_i, \ldots, q_n$ and $C = c_1, c_2, \ldots, c_j, \ldots, c_m$, of length n and m respectively, in order to align those two trajectories, an n-by-m matrix M is used. The M[i][j] is the distance $d(q_i, c_j)=(q_i-c_j)^2$. M[i][j] is the alignment between the points q_i and c_j as shown in Figure 1.

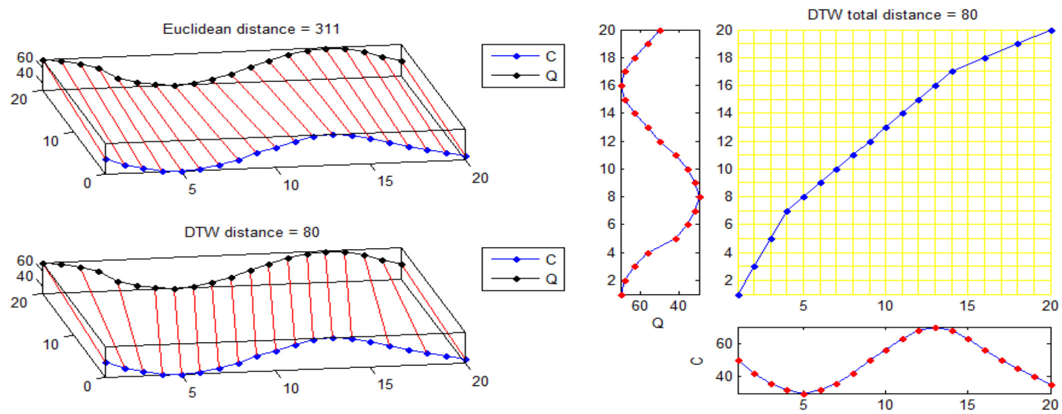


Figure 1. Dynamic Time Warping

The warping path p is a sequence of points, $p = (p_1, p_2, \ldots, p_K)$ where $p_l = (q_i, c_j)$. In DTW the following three restrictions should be satisfied:

Boundary: $p_1 = (1,1)$ and $p_K = (m, n)$, this means the warping path starts at the first points of aligned trajectories and ends at the last.

Continuity: for $p_k = (q_i, c_j)$ and $p_{k-1} = (q_i', c_j')$, $q_i - q_i' \leq 1$ and $c_j - c_j' \leq 1$. This means the warping path is composed by adjacent cells.

Monotonicity: for $p_k = (q_i, c_j)$ and $p_{k-1} = (q_i', c_j')$, $q_i - q_i' \geq 0$ and $c_j - c_j' \geq 0$. This preserves the time sequence in trajectory matching.

A warping path should also satisfy:

$$DTW(Q,C) = \min\left(\sqrt{\sum_{k=1}^{K} M_k}\right) \tag{9}$$

The dynamic programming is an optimal solution for finding the warping path. The cumulative distance dtw[i][j] used is defined as:

$$\text{dtw[i][j]} = \text{d[i][j]} + \min\{\text{dtw[i-1][j-1], dtw[i-1][j], dtw[i][j-1]}\} \tag{10}$$

The time complexity of DTW is O(nm). The Euclidean distance between two trajectories can be seen as a special case of DTW where $i = j$.

The procedure of DTW can be summarized as follows:

DTW(Q, C, D)// Q and C are the two trajectories. D is the distance matrix, each d[i][j] in D is the distance between q_i and c_j.

```
dtw[]= new double [n][m];// initialize matrix dtw
dtw[0][0]= 0;
for i = 1: n;
  dtw[i][1]= dtw[i-1][1] + d[i][1];
for j= 1:m;
  dtw[1][j]= dtw[1][j-1] + d[1][j];
for i = 1: n;
  for j= 1: m;
    dtw[i][j]= d[i][j]+min{dtw[i-1][j-1], dtw[i-1][j], dtw[i][j-1]}
return dtw;
```

5. Experimental Evaluation

In this section, we evaluate the DTW algorithm by comparing it with the well known IMHD and OWD. The main findings from the experiments are:

(1) DTW and IMHD algorithms are effective.

(2) Neither OWD nor IMHD is time sequence sensitive.

(3) OWD will falsely recognize a sample of one trajectory as anther trajectory when Δt, the update time interval, getting larger.

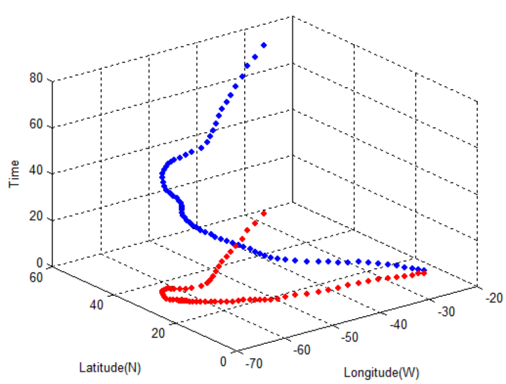
(4) OWD cannot distinguish the sample of one trajectory from another trajectory when Δd, the update distance, getting larger.

(5) DTW is time sequence sensitive and time scaling tolerating.

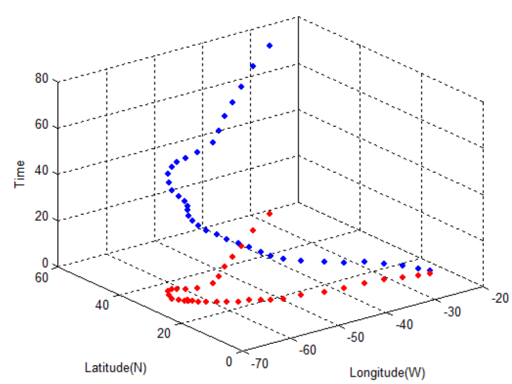
5.1. Experiments Setup

In order to evaluate the proposed algorithm, two real world datasets are used in our experiment. The first dataset records hurricanes of eastern Pacific from 1949 to 2011 containing 879 trajectories composed by 24768 points. The second dataset records hurricanes of Atlantic from 1851 to 2011 containing 1476 trajectories composed by 45580 points. Both datasets are available on the website [9]. The experiments were conducted on a Lenovo laptop V460 with Intel(R) 2.27GHz processor and 4GB RAM.

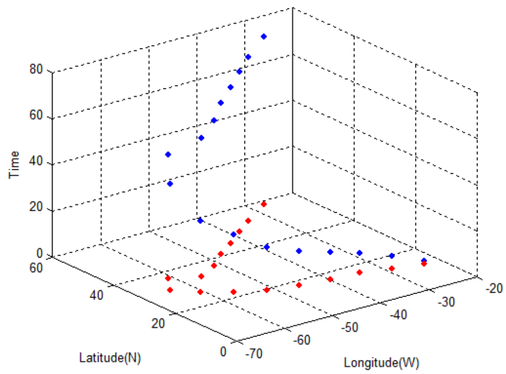
We study the behavior of 3 distance measurements (DTW, IMHD and OWL) on different trajectories. (a) 03-20 JUL 2008, Atlantic, Hurricane-3 BERTHA, where locations are updated every 6 hours; (b) the sampling time interval is bigger (12 hours); (c) the sampling distance is 6% of the whole trajectory length; (d) the sampling distance is 12% of the whole trajectory length; (e) the temporal sequence reversed trajectory of (a); (f) the time scaling of (a), where time dimension is amplified by 1.15. Figure 2 depicts trajectories (a) to (f) and their projections in spatial dimensions in blue and red respectively.



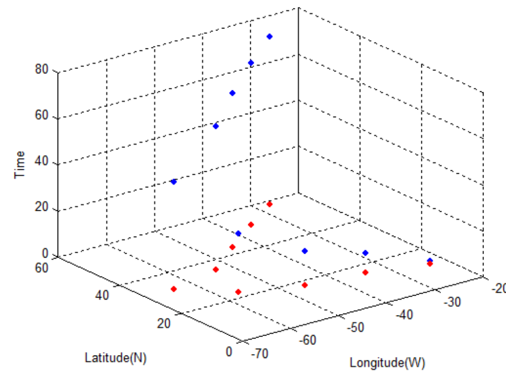
(a) 03-20 JUL 2008, Atlantic, Hurricane-3 BERTHA (6 hours)



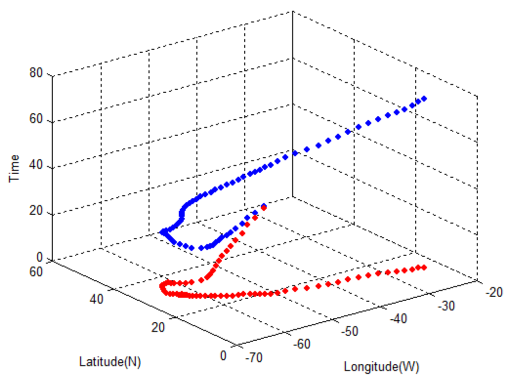
(b) The sampling time interval is bigger (12 hours)



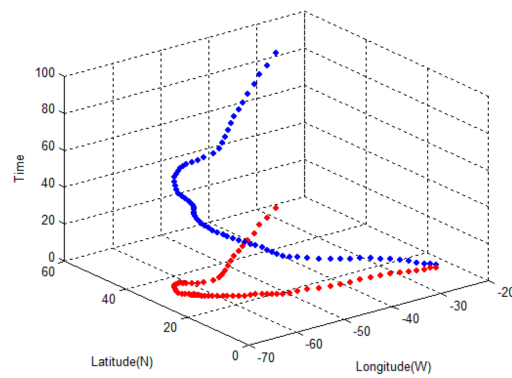
(c) The sampling distance is 6% of the whole trajectory length



(d) The sampling distance is 12% of the whole trajectory length



(e) The temporal sequence is reversed



(f) Time dimension is amplified by 1.15

Figure 2. Trajectories and their Projections on Spatial Dimensions

5.2. Similarities Calculations

In the following discussions, two criteria, *min* and *avg*, are used. Where *min* is the minimum distance between trajectory (a) and any other trajectory in the dataset, and *avg* is the average distance between trajectory (a) and another trajectory in a dataset.

Table 4. The Similarities Calculations

	(a) (6h)	(b) (12h)	(c) (6%)	(d) (12%)	(e) (reverse)	(f)Time scaling	min	avg
DTW	0	0.78	2.64	7.47	**182.93**	**19.22(33.21*)**	**31.18**	361.38
IMHD	0	0.49	2.31	6.92	**0**	**0**	23.10	194.13
OWD	0	21.64	63.80	**176.10**	**0**	**0**	**161.32**	1852.06

* is the time scaling of (a), where time dimension is amplified by 1.2 (20%).

It can be seen from Table 4 that both IMHD and DTW are effective. OWD falsely recognizes (d) as another trajectory, since the distance between (a) and (d) is larger than *min* (176.10>161.32). Neither IMHD nor OWD is time sequence sensitive, the distance between (a) and (e) is 0. DTW takes (e) as another trajectory, this makes DTW a time sequence sensitive algorithm. DTW can tolerate an almost 20% time scaling. When time dimension is amplified by 20%, DTW false recognizes the trajectory as another trajectory (33.21 is slightly larger than 31.18).

5.3. The Effect of Δt

Table 5. The Effect of Δt

	6h	12h	24h	48h	min	avg
DTW	0	0.78	2.44	6.91	31.18	361.38
IMHD	0	0.49	1.39	4.91	23.10	194.13
OWD	0	21.64	54.59	**285.09**	**161.32**	1852.06

It can be seen from Table 5 that when the update time interval getting larger, OWD cannot distinguish the sample of one trajectory from another trajectory. The distance between (a) and one of its samples (48h, 285.09) is bigger than *min* (161.32).

5.4 The Effect of Δd

Table 6. The Effect of Δd

	3%	6%	9%	12%	min	avg
DTW	1.05	2.64	9.97	7.47	31.18	361.38
IMHD	1.02	2.31	10.40	6.92	23.10	194.13
OWD	14.55	63.80	111.11	**176.10**	**161.32**	1852.06

Table 6 shows that when update distance getting larger, OWD will falsely recognize the sample of one trajectory as another trajectory. The distance between (a) and one of its samples (12%, 176.10) is bigger than *min* (161.32).

6. Conclusions

In this paper, through the comparisons of existing algorithm and the analysis of the requirements of LBS applications, the goals of the trajectory matching algorithm are introduced and quantified. Then a trajectory matching algorithm based on dynamic time warping is proposed. It can be seen from extensive experiments that the proposed algorithm is effective, time sequence sensitive, and time scaling tolerating.

Acknowledgements

This work is supported by a National Natural Science Foundation of China under Grant No. 60903169, two Shanghai Science and Technology Development Funds under Grant No. 075107006 and 09510703000, and a Shanghai Scientific Development Foundation (Grant No. 11530700300).

References

[1] J. Z. Gu, L. He, J. Yang and Z. Lu, “Location Aware Mobile Cooperation-Design and System”, International Journal of Signal Processing, Image Processing and Pattern Recognition, vol. 2, no. 4, **(2009)**, pp. 49-60.
[2] B. Lin and J. W. Su, “Geoinformatica”, vol. 12, no. 117, **(2008)**.
[3] S. Fei and J. Z. Gu, China communication, vol. 6, no. 128, **(2010)**.
[4] W. J. Rucklidge, International Journal of Computer Vision, vol. 3, no. 251, **(1997)**.

[5] S. Q. Zhao and Y. S. Gao, "Textural Hausdorff Distance for Wider-Range Tolerance to Pose Variation and Misalignment in 2D", Proceedings of the 22th IEEE Conference on Computer Vision and Pattern Recognition, Miami, USA, **(2009)** June 20-25.
[6] M. P. Dubuisson and A. K. Jain, "A modified Hausdorff distance for object matching", Proceedings of the 12nd International Conference on Pattern Recognition, Jerusalem, Israel, **(1994)** October 9-13.
[7] S. K. Sen, A. Bhattacharya and S. K. Das, Wireless Networks, vol. 5, no. 313, **(1999)**.
[8] P. Senin, "Dynamic Time Warping Algorithm Review", Information and Computer Science Department, University of Hawaii at Manoa, **(2008)**.
[9] http://weather.unisys.com/hurricane/index.html.

Authors

Peng Chen received his BE degree from the Department of Computer Science and Technology, East China Normal University, in 2008. He is currently working toward the PhD degree in the Department of Computer Science and Technology, East China Normal University. His research interests include uncertain data, location based service and context aware computing.

Junzhong Gu Professor of Computer Science, Head of Institute of Computer Applications, East China Normal University, China. He received the MS degree in Computer Science from East China Normal University in 1982. He works at East China Normal University since 1982. He worked as visit professor at GMD and University Mannheim, Germany (1987-1989, and 1991-1993). His research interests now include context aware computing, distributed data management, and multimedia information processing.