Quantitative Similarity Analysis among Three-dimensional Trajectories of Events

[1]M. I. Casillas-del Llano, [2]M. A. Mirón-Bernal and [3]J. Figueroa Nazuno

Centro de Investigación en Computación.
Instituto Politécnico Nacional
Unidad Profesional "Adolfo López Mateos" -Zacatenco- México, D.F.
[1]mcasillas@sagitario.cic.ipn.mx, [2]amirona06@sagitario.cic.ipn.mx, [3]jfn@cic.ipn.mx
Paper received on 08/08/08, accepted on 11/09/08.

Abstract. In many situations of several computational areas, the three-dimensional trajectory of an event has been measured in a precise way. However, there are not direct quantitative techniques that provide us an scalar value representing the similarity among several events trajectories. In this paper, a computational technique is presented that obtains a direct similarity value among the trajectories of two events in the three-dimensional space. Experimental results show that, similarity measures provided by this technique among the artificially generated trajectories, are an effective comparison indicator.

1 Introduction

1.1 The Similarity Problem

Humans deal with the similarity problem in an *apparently* simple way. They are able to distinguish *likeness* among different entities, such as two similar songs, stories referring to the same event, politic coincidences among candidates; characteristics that based on their judgement, become people more coincident in their opinions (see [1]). All of this involves physical processes (little known as today) that occur in the human brain

The first attempts to understand the similarity problem, in a quantitative way, were given by Tversky[2] in 1977. He stated that this problem can be dealt by four approaches (Fig. 1). The *common elements approach* (Fig. 1a) refers about comparison among entities using a common element criterion. The *template approach* (Fig. 1b) makes comparisons focusing in *pairing off*, or *aligning*, an entity with another one. The *geometric approach* (Fig. 1c) performs comparisons among entities based on their geometric structure. *Feature approach* (Fig. 1d) is similar to the *common elements approach*, although last one doesn't require a previous recognition of the properties involved in the comparison.

Tversky's appreciation[2] is that similarity problem has unique inherent-context characteristics.

© M. G. Medina Barrera, J. F. Ramírez Cruz, J. H. Sossa Azuela. (Eds.)
Advances in Intelligent and Information Technologies.
Research in Computing Science 38, 2008, pp. 175-184

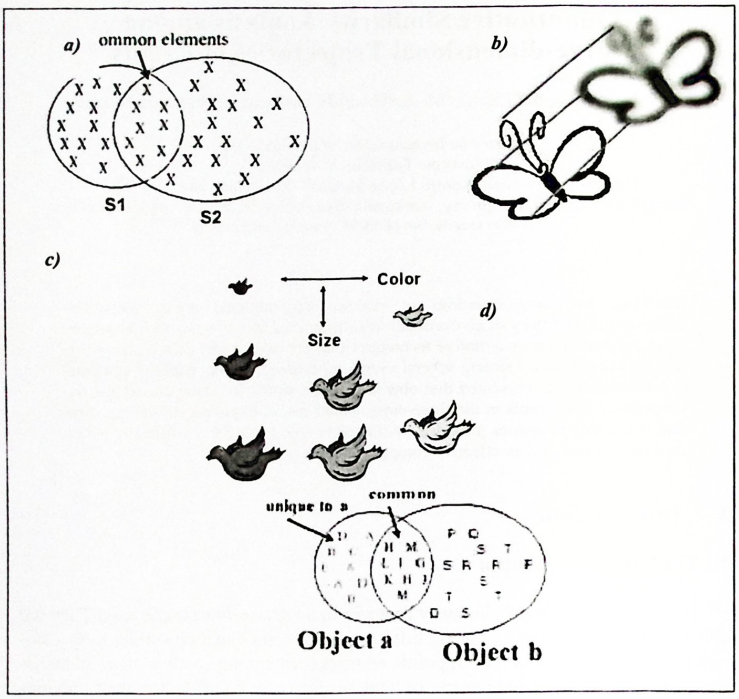

Fig. 1. The four approaches proposed by Tversky. Images extracted from [2].

1.2. The problem about finding similarity among trajectories of events.

There are many situations where it is desirable to find a scalar value of similarity among two events. For example, it is useful to know the registered trajectories among several pollutants in the atmosphere. The ability to detect similar trajectories has an important role in various applications, such as clustering and classification, k-Nearest-Neighbor queries, or indexing large 3D trajectory data repositories[10].

Finding similarity among two or more trajectories is a non-analytical soluble problem, both in two and three-dimensional spaces. Thus, it is necessary to develop techniques that could be useful in situations where obtaining a scalar value representing similarity among two or more trajectories is needed.

2 Algorithm's Description

The algorithm presented in this paper is based on the *Fast Dynamic Time Warping* (FDTW) algorithm and other techniques used to find similarity among data sets: refer to [3-5, 7-9, 11]. FDTW is an algorithm presented by Jang[6] in 2000, which minimizes a *base distance* between two sequences Q and C of size $|Q|$, $|C|$, respectively; that is, $FDTW(Q,C) \leq D_{base}$. It also performs a temporal "alignment" among the two sequences, that is, it constructs a point relation denominated *warp path*, where the ith point of the Q matrix is related to the jth point of the C matrix.

The following algorithm calculates the FDTW distance:

FDTW (Q,C)
//First element
1. $M[0,0] = D_{base}(q_0, c_0)$
//First row
2. $M[0,j] = M[0,j-1] + D_{base}(q_0, c_j)$
//First Column.
3. $M[i,0] = M[i-1,0] + D_{base}(q_i, c_0)$
//Rest of the Matrix
4. $M[i,j] = D_{base}(q_i, c_j) + min\{M[i-1,j-1], M[i-1,j], M[i,j-1]\}$

Where the *base distance* (D_{base}) is the Euclidean distance.

The FDTW distance has several properties:

- Similarity between sequences decreases if the FDTW approaches to infinity
- Similarity between sequences increases if the FDTW approaches to 0.
- FDTW distance between sequences Q and C is the same as the distance between the sequences C and Q

Warp path (W) is a bidimensional array that contains the point relation (mapping) among Q and C sequences. This Warp Path is constructed doing a backtracking of the matrix M previously calculated. Starting from the last element of M (that is, the element containing the FDTW distance), we choose the least of the three neighbors of the actual element that minimizes the cost of the distance, that is, the elements M[i–1, j-1], M[i-1, j] and M[i, j-1].

2.1 Method Proposed

The algorithm presented in this paper deal with the data matrices that represent the artificially generated trajectories of several events in the three-dimensional

space. The basic idea of the algorithm also consists in minimizing a *base distance* between two matrices, in this case named $Q_{m \times n}$ and $C_{n \times o}$. In order to calculate the *base distance*, the two matrices are combined into a single *three-dimensional data cube*, of size $m \times n \times o$. This data cube will map each element of the Q matrix with each element of the C matrix. The idea behind this is to make an "alignment" between these matrices, and calculate the cost of this alignment.

As well as FDTW, the (x, y, z) element of the data cube will be calculated from the sum of the Euclidean distance among an element of Q and an element of C, and the least of the seven neighbors of the element. Such neighbors are the elements found in the same level as the actual element from 1) 45°, 2) 0°, 3) 90°; and the level below from 4) 45°, 5) 0°, 6) 90° and 7) 360°.

Once the data cube has been constructed, the last of its element will be defined as the *base distance* among the two matrices.

We conclude that:

$$FDTW3D(Q,C) \rightarrow \infty \quad \text{if } Q \text{ and } C \text{ are different} \tag{1}$$

$$FDTW3D(Q,C) = 0 \quad \text{if } Q \text{ and } C \text{ are equal} \tag{2}$$

$$FDTW3D(Q,C) = FDTW3D(C,Q) \tag{3}$$

Warp Path 3D is a three-dimensional array that contains the point relation between the Q and C matrices. As well as FDTW, this Warp-Path is constructed starting from the last element of the three-dimensional data cube. This element contains the FDTW3D distance between Q and C. From there, we look for the seven neighbors that also minimizes the cost of the distance. This neighbors are the same as the ones considered in the construction of the data cube, that is, the neighbors located from 1) 45°, 2) 0°, 3) 90°; and the level below from 4) 45°, 5) 0°, 6) 90° and 7) 360°.

The Warp Path 3D has the following properties:

$$W = \{ w_1, w_2, w_k, \ldots, w_K \} \tag{4}$$

$$max(|Q|, |C|) \leq K \leq (|Q| + |C|) - 1 \tag{5}$$

$$w_k = (i, j, z), \quad 1 \leq i \leq |Q|, \quad 1 \leq j \leq |C|, \quad 1 \leq z \leq |C| \tag{6}$$

This Warp Path 3D is subjected to the following constraints:

1. **Begin / End**: This constraint requires that the first element of the Q matrix must be mapped with the first element of the C matrix, and the last element of the Q matrix must be mapped with the last element of the C matrix.

2. **Monotonocity:** The elements in the Warp Path 3D must follow the condition that $w_{k-1} \leq w_k$, within their respective (i, j, k) index.
3. **Local Restrictions:** The local restrictions are used to reduce the range search of the elements using the neighborhood of a matrix cell.

The general procedure used to calculate similarity among two trajectories of events is presented as follows. Let Q and C be two matrices that represent the values of a pair of trajectories:

FDTW3D (Q, C)

//FIRST ELEMENT

1. $Cube[0,0,0] = D_{base}\left(q_{0,0}, c_{0,0}\right)$

//FIRST AXIS OF THE CUBE

2. $Cube[0,j,0] = D_{base}\left(q_{j,0}, c_{y,0}\right) + Cube[0,j-1,0]$

//SECOND AXIS OF THE CUBE

3. $Cube[i,j,0] = D_{base}\left(q_{j,0}, c_{j,i}\right) + Cube[i-1,j,0]$

//THIRD AXIS OF THE CUBE

4. $Cube[0,j,k] = D_{base}\left(q_{j,k}, c_{j,0}\right) + Cube[0,j,k-1]$

//REST OF THE CUBE

5. $$Cube[i,j,k] = D_{base}\left(q_{j,k}, c_{j,i}\right) + \min \begin{cases} Cube[i-1, j-1, k-1] \\ Cube[i-1, j-1, k] \\ Cube[i, j-1, k-1] \\ Cube[i, j-1, k] \\ Cube[i-1, j, k-1] \\ Cube[i-1, j, k] \\ Cube[i, j, z-1] \end{cases}$$

$$FDTW3D = Cube[m,n,o]$$

That is, the FDTW 3D distance is the last element of the data cube. Then, in order to construct the Warp Path 3D that maps the matrix Q with the matrix C, we proceed as follows:

1.1. We find the least of the seven neighbors of the actual element, and the coordinate (x,y,z) of that element is stored in the WarpPath3D register.
1.2. Repeat step 3.1. with each element of the three-dimensional data cube, until we reach the first element of the cube.

Since it is necessary to construct each of the axis of the three-dimensional data cube in an accumulative manner, the complexity of this algorithm is $O(n^3)$. An analogous work presented in [12] mentions that their algorithm look for a total of fifteen neighbors. It is worth to mention that also has a complexity of $O(n^6)$.

3 Experimental Results

A set of one hundred trajectories of events in the three-dimensional space were artificially generated, each of one consisting of 500 points and were stored in 500 x 3 matrices, as shown in Figure 2, where the three columns represent the axis of the three-dimensional space, and the 500 rows represent each point of the trajectory. These trajectories were generated by randomly choosing some mathematical functions (such as sine, cosine, etc.) or simply incrementing the values by a random value; as well as random values were used in order to change the direction of the trajectories. Figure 3 shows some of these trajectories.

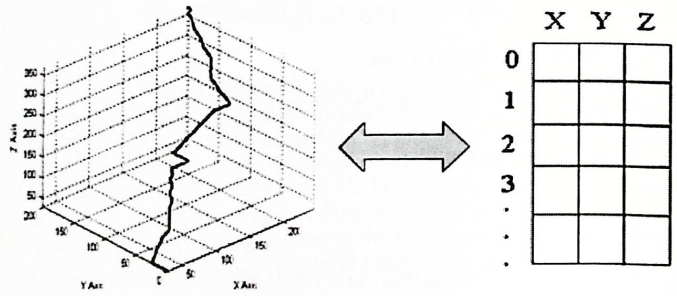

Fig. 2. Trajectories are stored in two-dimensional matrices. Columns represent the X, Y and Z axis, while rows represent the time stamp of the trajectory.

The algorithm performance, implemented by a computational tool, was tested with the set of one hundred trajectories. Each trajectory were compared with the rest of them, and the base distances obtained were stored in a table, as shown in Table 1 (example of one fraction of the whole table).

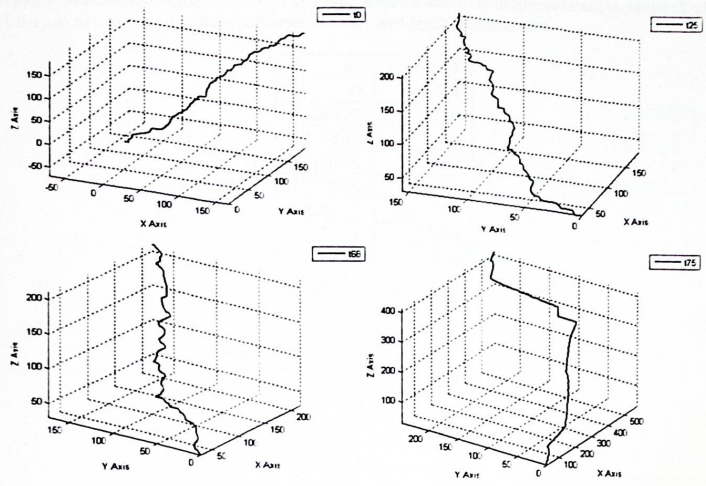

Fig. 3. Example of four trajectories artificially generated.

Table 1. Base distances obtained by the comparison of the hundred trajectories set (the results of the first five trajectories are shown).

	t0	**t1**	**t10**	**t11**	**t12**
t0	0	4,568.37	5,396.91	6,865.13	32,720.76
t1	4,568.37	0	6,177.75	6,274.20	27,520.46
t10	5,396.91	6,177.75	0	4,107.44	17,042.95
t11	6,865.13	6,274.20	4,107.44	0	13,290.97
t12	32,720.76	27,520.46	17,042.95	13,290.97	0

The similarity analysis process is made by identifying the trajectories with the least base distances. As mentioned before, this distance represent the cost or the "effort" in order to perform the warping, i.e. the cost of the "alignment" between the two matrices; the less the base distance between two trajectories is, the similarity value increases. Similarity value decreases while the FDTW 3D distance increases among the trajectories. For example, Table 2 shows nine trajectories with the least base distance between the a) t0 trajectory and the b) t1 trajectory.

Table 2. Comparison results of trajectories a) t0 and b) t1. It is shown, in order, the trajectories with the least base distances calculated.

Query :	t0
t0	0
t67	1708.65
t6	2651.88
t64	2667.85
t2	2893.51
t90	3335.94
t28	3343.87
t63	3392.29
t5	3526.79

a)

Query :	t1
t1	0
t5	1961.14
t97	2187.15
t61	2360.85
t9	2636.9
t23	2655.23
t22	2834.52
t60	2955.06
t29	3100.21

b)

As expected, the distance between the query trajectory with itself is zero, since there is no cost while mapping the two trajectories when they are the same. To confirm the results, a trajectory with their four most similar trajectories were plotted. Figure 3 shows, the comparison results of trajectory *t*0 (3.a.) and trajectory *t*1 (3.b). This figure contains the query trajectory, as well as their four trajectories that were considered to be the most similar.

4 CONCLUSIONS

Results obtained by this technique indicate that it is possible to obtain a similarity scalar value of three-dimensional trajectories in the three-dimensional space. This kind of analysis is very important, since there are so many conditions that rule the movement of an event (and not only by the First, Second and Third Newton's laws in Classical Mechanics). Therefore, it is affected by many other phenomena; for example, a pollutant in the atmosphere is affected by temperature, pressure, wind direction, etc., and its behavior is going to be more alike a *random walk*. Even in the

extreme case that its trajectory would consist of a quasi-randomly behavior, our technique allow to obtain similarity values among trajectories.

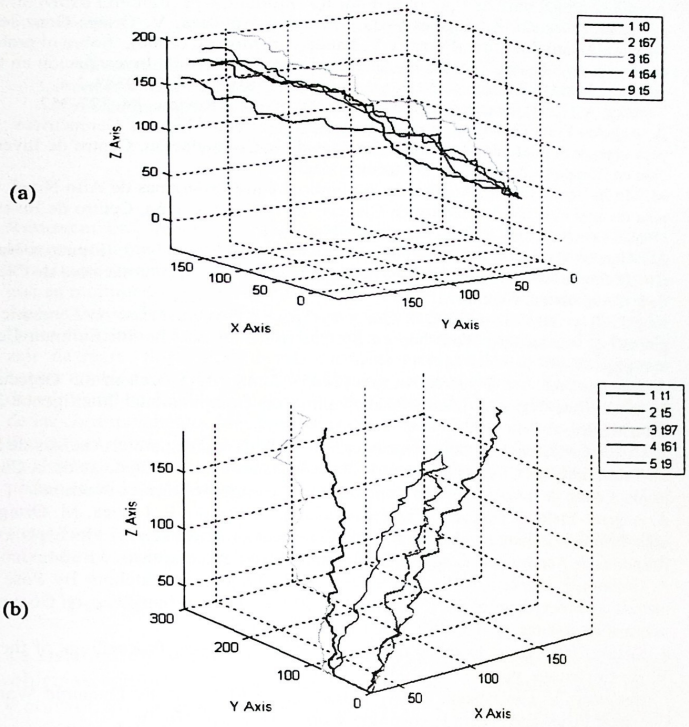

Fig. 3. Example trajectory plot with their four most similar considered trajectories. a) t0, b) t1

Generally, there are situations where we have to analyze the behavior or similarity of a phenomenon in the three-dimensional space in the system, such as: the movement of a dot in a lady's dress, the trajectory of an object simulated in real time (such as an airplane), similarity between trajectories of particles, similarity of several trajectories of a person running in a movie, etc. In all of these situations, it would be useful to count on a technique that could provide us a scalar value of similarity in order to analyze the behavior of an event, beyond the Newton's laws. Results indicate that similarity values obtained among trajectories artificially generated

by the presented technique are an effective comparison indicator. Besides, this procedure could be generalized to other situations in the modern computing.

References

1. J. Figueroa-Nazuno, A. Angeles-Yreta, J. Medina-Apodaca, V. Ortega-González, K. Ramírez-Amaro, M. Mirón-Bernal y V. Landassuri-Moreno (2006). Sobre el problema de Semejanza. Reporte Técnico No. 223, Serie Azul, Centro de Investigación en Computación. Instituto Politécnico Nacional, D.F. 2006. ISBN: 970-36-0343-2
2. Tversky, A. (1997). Features of similarity. Psychological Review, 84, 327-352.
3. A. Angeles-Yreta (2006). Cómputo de la Similitud entre Figuras Geométricas. Tesis para obtener el grado de Maestro en Ciencias de la Computación, Centro de Investigación en Computación, Instituto Politécnico Nacional.
4. M. Mirón-Bernal (2008), Análisis de la Similitud entre Programas de Alto Nivel. Tesis para obtener el grado de Maestro en Ciencias de la Computación, Centro de Investigación en Computación, Instituto Politécnico Nacional.
5. A. Angeles-Yreta, H. Solís-Estrella, V. Landassuri-Moreno y J. Figueroa-Nazuno (2004), Similarity Search In Seismological Signals, Encuentro Internacional de Ciencias de la Computación, Colima, México.
6. Jang J.S.R y Gao M.Y (2000). A Query-by-Singing System based on Dynamic Programming. International Workshop on Systems Resolutions (the 8th Bellman Continuum), pp. 85-89.
7. A. Ángeles-Yreta y J. Figueroa-Nazuno (2005). Similarity Search in 3D Objects and Dynamic Time Warping, International Seminar on Computational Intelligence 2005, D.F., México.
8. V. Ortega-González, J. Figueroa-Nazuno (2008). Una Técnica para el Análisis de Similitud entre Imágenes. Tesis para obtener el grado de Maestro en Ciencias de la Computación, Centro de Investigación en Computación, Instituto Politécnico Nacional.
9. A. Ángeles-Yreta, S. García, J. Figueroa-Nazuno, M. Romo, F. Correa, M. Ortega, H. Solís-Estrella, K. Ramírez-Amaro (2005), Clasificación y Semejanza de Espectros de Respuesta de Aceleración, Congreso Nacional de Ingeniería Sísmica, D.F., México.
10. A. Croitoru, P. Agouris, A. Stefanidis (2005). 3D Trajectory Matching By Pose Normalization. Proceedings of the 13th annual ACM international workshop on Geographic information systems, pp. 153-162, ISBN:1-59593-146-5
11. E. Keogh (2002). Exact Indexing of Dynamic Time Warping. Proceedings of the 28th VLDB Conference, pp. 406-417, 2002.
12. L. Hansheng, V. Govindaraju (2004). Direct Image Matching by Dynamic Warping, Computer Vision and Pattern Recognition Workshop, 27, pp. 76-76.