Trajectory Voting and Classification based on Spatiotemporal Similarity in Moving Object Databases

Costas Panagiotakis[1], Nikos Pelekis[2], and Ioannis Kopanakis[3]

[1] Dept. of Computer Science, University of Crete, P.O. Box 2208, Greece, cpanag@csd.uoc.gr
[2] Dept. of Informatics, University of Piraeus, Greece, npelekis@unipi.gr
[3] E-Business Intelligence Lab, Dept. of Marketing, Technological Educational Institute of Crete, Greece kopanakis@e-bi.gr

Abstract. We propose a method for trajectory classification based on trajectory voting in Moving Object Databases (MOD). Trajectory voting is performed based on local trajectory similarity. This is a relatively new topic in the spatial and spatiotemporal database literature with a variety of applications like trajectory summarization, classification, searching and retrieval. In this work, we have used moving object databases in space, acquiring spatiotemporal 3-D trajectories, consisting of the 2-D geographic location and the 1-D time information. Each trajectory is modelled by sequential 3-D line segments. The global voting method is applied for each segment of the trajectory, forming a local trajectory descriptor. By the analysis of this descriptor the representative paths of the trajectory can be detected, that can be used to visualize a MOD. Our experimental results verify that the proposed method efficiently classifies trajectories and their sub-trajectories based on a robust voting method.

1 Introduction

Nowadays, there is a tremendous increase of moving objects databases due to location-acquisition technologies like GPS and GSM networks [1], and to computer vision based tracking techniques [2]. This explosion of information combines an increasing interest in the area of trajectory data mining and more generally the knowledge discovery from movement-aware data [3]. All these technological achievements require new services, software methods and tools for understanding, searching, retrieving and browsing spatiotemporal trajectories content.

A MOD consists of spatiotemporal trajectories of moving objects (e.g. humans, vehicles, animals, etc.). In general case, these trajectories encode the 2-D (two dimensional) or 3-D geographic location and the 1-D time information. Many of the existing approaches are interested in the trajectory shape analysis considering that the trajectory consists of sequential 2-D or 3-D spatial sampling positions ignoring the temporal dimension [4], [5]. In [6], a trajectory clustering algorithm is proposed that partitions a 2-D trajectory into a set of line segments, and then, groups similar line segments together into a cluster, while the

notion of the *representative trajectory* of a cluster is defined. The algorithm is based on geometrical distances between line segments taking into account position and orientation. These methods can be applied on trajectory segmentation, classifications, searching and retrieval problems using shape based descriptors. Based on the idea of partial trajectories, Lee et al. [7] proposed an algorithm for trajectory classification showing that it is necessary and important to mine interesting knowledge on partial trajectories rather than on the whole. However, both of these algorithms cannot be applied with complex time-aware trajectories considering the whole route of the moving objects.

In addition, temporal dimension is ignored by almost all computer vision based methods, that are interested in human action and activity recognition. Many of them use 2-D trajectories from specific human points that are tracked in video sequences with constant frame rate [8], [9]. Another class of methods use temporally annotated sequences [10], [1], performing mining tasks. In these methods, as temporal dimension is used the transition time between sequentially points of the trajectory. Therefore, a trajectory of $n+1$ points $S = (s_0, s_1, ..., s_n)$, is stored as $T = (S, \Delta t_1, \Delta t_2, ..., \Delta t_n)$, where Δt_i, denotes the transition time between the points s_{i-1} and s_i. The use of transition time takes into account that the sampling rate could be varied, providing information about speed. However, the format in temporal dimension changes and important information for some real world applications is missing. In real world, there are applications where the temporal dimension should be used unchanged. These applications concern traffic monitoring, security applications (e.g. identifying "illegal" trajectories under shape and space-time requirements), searching using space-time constraints, and so on.

In [11], distance-based criteria have been proposed for segmentation of object trajectories using spatiotemporal information. First, Minimum Bounding Rectangles (MBRs) is used in order to simplify the trajectories, taking advantage their tight integration with existing multidimensional indexes in commercial database management systems (such as R-trees). The use of R-trees reduces the computation cost of trajectory searching to $O(log(n))$, where n denotes the number of trajectories. The distance between two trajectories is defined used MBRs representation. Finally, the segmentation problem is given as a solution of a maximization problem, that attempts to create MBRs in such a way, that the original pairwise distances between all trajectories are minimized. In [12], a framework consisting of a set of distance operators based on primitive (space and time) as well as derived parameters of trajectories (speed and direction) has been introduced. They assume linear interpolation between sampled locations, so that a trajectory consists of a sequence of 3-D line segments, where each line segment represents the continuous development of the moving object during sampled locations. In [13], representative motion paths (frequently traveled trails of numerous moving objects) are detected in a distributed system under the assumption that the moving objects can communicate with a central unit (coordinator) and all processing must be performed in a single pass over the stream. The location measurements of each object is modeled with some

uncertainty tolerance ϵ and a one-pass greedy algorithm, termed RayTrace, is running on each object independently. They have proposed a one-pass greedy algorithm, termed RayTrace, running on each object independently. The coordinator utilizes a lightweight index structure, termed MotionPath, which stores the representative motion paths. The goal of this work is in the same context with the aim of our research. However, they ignore segments' orientation taking into account only the points of the trajectories. Moreover, they propose to the use of a step function to formulate the closeness of two points. On the other hand the use of a continuous decision function derived robust and smooth results (in our approach).

Most of the above mentioned approaches propose different similarity metrics which they utilize either for introducing indexing structures for vast trajectory retrieval, or for clustering purposes, focusing either on space criteria, ignoring temporal variation, minimizing predefined metric criteria on feature domain, simplifying the given trajectories or applying simple clustering-based techniques. We argue that all of the above approaches, as well as those which are dealing with vast volumes of trajectory datasets would benefit if they would be applied in a representative subset (consisting of the representative trajectories) that best describes the whole dataset. Consider for example the domain of visual analytics on movement data [14] in which it is meaningless to visualize datasets over a certain small size, as the human eye cannot distinguish any movement pattern due to the immense size of the data. On the contrary, in this paper, we don't simplify the given trajectories, as we use the original data unchanged. Moreover, the temporal information is taken into account.

We are proposing a global voting method that is applied for each segment of trajectory without any simplification. Then, we analyze the voting descriptor in order to detect the representative paths of the trajectory, that followed by many objects at almost the same time and space. Moreover, we classify the trajectories and the trajectory segments. The results of classification have been used to visualize a MOD. The proposed methodology can be applied under different distance metrics (e.g. non Euclidean) and higher trajectory dimensions.

The rest of the paper is organized as follows: Section 2 gives the problem formulation describing the proposed modelling. Section 3 presents the proposed method for trajectory voting and classification. The experimental results are given in Section 4. Finally, conclusions and discussion are provided in Section 5.

2 Problem Formulation

In this section the problem formulation is given. Let us assume a MOD $D = \{T_1, T_2, \cdots, T_n\}$, of n trajectories, where T_k denotes the k-trajectory of the dataset, $k \in \{1, 2, ..., n\}$. We assume that the objects are moving in the xy plane. Let $p_k(i) = (x_k(i), y_k(i), t_k(i))$, be the i-point, $i \in \{1, 2, ..., L_k\}$ of k-trajectory, where L_k denotes the number of points of k-trajectory. $x_k(i), y_k(i)$ and $t_k(i))$ denote the 2-D location and the time coordinate of point $p_k(i)$, respectively.

Similar to the work of [12], [6], we consider linear interpolation between successive sampled points $p_k(i)$, $p_k(i+1)$, so that each trajectory consists of a sequence of 3-D line segments $e_k(i) = p_k(i)p_k(i+1)$, where each line segment represents the continuous moving of the object during sampled points. The goal of this work is to detect representative paths[4] and trajectories, that followed by many objects at almost the same time and space. A method to detect them is to apply a voting process for each segment $e_k(i)$ of the given trajectory T_k. This means that $e_k(i)$ will be voted by the trajectories of MOD, according to the distance of $e_k(i)$ to each trajectory. The sum of these votes is related to the number of trajectories that are close to $e_k(i)$. If this number is high, means that the segments is representative, followed by many objects at almost the same time and space. Thus, the voting results will be used to detect the representative paths and trajectories. First, we have to determine the distance $d(e_k(i), T_m)$ between $e_k(i)$ and a trajectory T_m of the dataset that consists of line segments. $d(e_k(i), T_m)$ is defined as the distance between $e_k(i)$ and the closest line segment of T_m to $e_k(i)$:

$$d(e_k(i), T_m) = min_j d(e_k(i), e_m(j)) \tag{1}$$

So, we have to compute distances between 3-D line segments ($d(e_k(i), e_m(j))$). In this framework, the meaning of ($d(e_k(i), e_m(j))$) is equal to the minimum energy of transportation of line segment $e_k(i)$ to $e_m(j)$, or line segment $e_m(j)$ to $e_k(i)$. Between these two choices, the transportation of minimum energy is selected. This idea has been introduced on Earth Movers Distance (EMD) framework [15] and it has been successfully applied on pattern recognition and computer vision applications [16]. In our case, this energy can be defined by the sum of two energies:

- translation energy $d_\perp(e_k(i), e_m(j))$ and
- rotation energy $d_\angle(e_k(i), e_m(j))$,

that depend on the Euclidean distance and on angle between the line segments, respectively. Therefore, taking into account the orientation of line segments, we have added an expression $d_\angle(e_k(i), e_m(j))$ to the distance formula related to the angle θ between the line segments,

$$d(e_k(i), e_m(j)) = d_\perp(e_k(i), e_m(j)) + d_\angle(e_k(i), e_m(j)) \tag{2}$$

$$d_\angle(e_k(i), e_m(j)) = min(|e_k(i)|, |e_m(j)|) \cdot sin(\theta) \tag{3}$$

where $d_\perp(e_k(i), e_m(j))$ denotes the Euclidean distance between the 3-D line segments and $|e_k(i)|$ the Euclidean norm (length) of 3-D line segment $e_k(i)$. In order to minimize the energy according to EMD definition, we select to rotate the line segment of minimum length, see Equation 3. Moreover, $d_\angle(e_k(i), e_m(j))$ has been expressed in "Euclidean distance" units, measuring the maximum distance that a point of line segment of minimum length will cover during rotation. If $d_\angle(e_k(i), e_m(j))$ was expressed in rads, we should introduce a weight to make $d_\perp(e_k(i), e_m(j))$ and $d_\angle(e_k(i), e_m(j))$ comparable (similar with Equation 4). Fig.

[4] In this framework, "path" is used for a trajectory part.

1 illustrates the Euclidean distance (red dotted line) between the 3-D line segments $p_k(i)p_k(i+1)$ and $p_m(j)p_m(j+1)$.

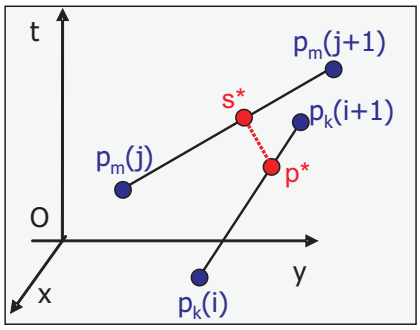


Fig. 1. The (closest) points p^* and s^* of 3-D line segments $p_k(i)p_k(i+1)$ and $p_m(j)p_m(j+1)$ define their distance.

The distance $d(e_k(i), e_m(j))$ cannot be expressed by a single formula, but it can be estimated in $O(1)$ [17]. This computation cost is not affected by the $d_\angle(e_k(i), e_m(j))$, since it a constant value for (each pair of points of the) two line segments. In order to estimate the Euclidean distance $d_\perp(p, s)$ between two points $p = (x, y, t)$ and $p' = (x', y', t')$, weights (w_1, w_2) can be used (Equation 4), making comparable location and time differences.

$$d_\perp(p, p') = \sqrt{w_1 \cdot (x - x')^2 + w_1 \cdot (y - y')^2 + w_2 \cdot (t - t')^2} \tag{4}$$

The weights can be defined by the user. The ratio $\frac{w_2}{w_1}$ determines the spatial difference (e.g. how many meters) that "is equivalent" with one unit time difference (e.g. one second). This ratio can be estimated my the mean speed.

3 Global Voting and Classification

3.1 Voting Method

This section describes the proposed algorithm, the Global Voting Algorithm (GVA). The input of the algorithm is a MOD $D = \{T_1, T_2, \cdots, T_n\}$, a trajectory $T_k \in D$ and an intrinsic parameter σ of the method. The output of the method is the vector V_k of $L_k - 1$ components that can be considered as a trajectory descriptor along the T_k line segments. Each component of the vector $V_k(i)$ corresponds to the number of votes (representativeness) of $e_k(i), i \in \{1, 2, ..., L_k - 1\}$ of T_k.

According to the problem formulation, the algorithm for each line segment $e_k(i)$ of T_k and $T_m \in D, m \neq k$ computes the distance $d(e_k(i), T_m)$. This distance will be used to define the voting function $V(e_k(i), T_m)$. In literature, a lot

of voting functions have been proposed, like the step functions or continuous functions [18]. In this work, we have selected to use the continuous function of gaussian kernel getting,

$$V(e_k(i), T_m) = e^{-\frac{d^2(e_k(i), T_m)}{2 \cdot \sigma^2}} \tag{5}$$

The gaussian kernel is widely used in a variety of applications of pattern recognition [19]. The control parameter σ shows how fast the function ("voting influence") decreases with distance. According to Equation 5, it holds that $0 \leq V(e_k(i), T_m) \leq 1$. If $d(e_k(i), T_m)$ is close to zero, the voting function gets its maximum value, giving 1.0. This means, that there exists a line segment of T_m that is being very close (in time and space) to $e_k(i)$. Otherwise, if $d(e_k(i), T_m)$ is high, e.g. greater than $5 \cdot \sigma$, the voting function gets almost 0, meaning that T_m is very far away (in time or space) from $e_k(i)$.

The use of a continuous voting function, like the gaussian kernel, gives smooth results for small changes on parameters (σ), and the possibility to get decimal values as results of voting process increasing the robustness of the method. Finally, $V_k(i)$ is estimated by getting the sum of votes for all of trajectories $T_m \in D, m \neq k$. Given the above discussion, a nice property that holds is that the proposed local trajectory descriptor V_k changes continuously over the trajectory segments. The pseudo-code of the above procedure is depicted at the end of the section (see Algorithm 1). The next subsection discusses the using of local trajectory descriptor V_k to classify trajectories and to detect representative paths.

input : The moving objects database $D = \{T_1, T_2, \cdots, T_n\}$ and a trajectory $T_k \in D$, parameter σ for voting.
output: The voting vector of T_k, V_k.

for $i = 1$ **to** $L_k - 1$ **do**
  $V_k(i) = 0$
  for $m = 1$ **to** n **do**
    if $m \neq k$ **then**
      $V_k(i) = V_k(i) + e^{-\frac{d^2(e_k(i), T_m)}{2 \cdot \sigma^2}}$
    end
  end
end

Algorithm 1: Global Voting Algorithm (GVA).

3.2 Trajectory Classification

In this section, we describe the analysis of local trajectory descriptor V_k in order to detect the representative paths of the trajectory and to classify the trajectories. Representative paths or representative trajectories are followed by many

objects at almost the same time and space. In order to identify the representative trajectories, we will introduce E_k that is defined by the mean value of V_k over the line segments of T_k.

$$E_k = \frac{1}{L_k - 1} \sum_{i=1}^{L_k - 1} V_k(i) \tag{6}$$

This value is a measurement of trajectory representativeness. Therefore, a classification of the trajectories can be done using this value. Another trajectory feature is the maximum value of V_k, $M_k = max_i V_k(i)$. By the analysis of M_k, the representative line segments can be detected.

The trajectory classification results can be used for an efficient visualization and sampling of large datasets. The visualization of a large MOD suffers from the problem that the space-time density of the trajectories is extremely high (see Fig. 2(b)). A solution on this problem is given by an efficient sampling of the MOD, that can be provided by the classification results, using the detected representative trajectories (see Fig. 2(c)). In Section 4, we present experimental results concerning trajectory classification and visualization. The next subsection discusses the computational complexity issues of the proposed algorithm.

3.3 Computational Complexity Issues

Concerning the Global Voting Algorithm (GVA) complexity, the computational cost for each line segment $e_k(i)$, of T_k is $O(n)$. The computation cost of GVA (estimation of V_k) is $O(L_k \cdot n)$. If we perform GVA for each trajectory of the database, then the total computation cost is $O(\bar{L} \cdot n^2)$, where $\bar{L}$ denotes the mean number trajectory points (samples). Therefore the polynomial cost of the algorithm makes the algorithm efficient for large databases (i.e. more than 1000 trajectories needed few seconds).

However, it is possible to reduce this computation cost, in order to be able to execute the algorithm in even larger databases. MBRs can be used as initialization step, and the indexing of the line segments to MBRs should be stored. We have proposed the using MBRs because of their tight integration with multidimensional indexes (R-trees). Then, the cost of searching step of voting algorithm will be reduced in an MBR (or some MBRs). Thus, the use of R-trees will reduce the cost of GVA execution to $O(log(\bar{L} \cdot n))$, and the total cost to $O(n \cdot log(\bar{L} \cdot n))$.

4 Experimental Results

The method has been implemented using Matlab without any code optimization or using of R-trees structures. For our experiments, we used a Core 2 duo CPU at 1.5 GHz. A typical processing time of GVA execution, when $n = 1000$ and $\bar{L} = 100$, is about 3 seconds.

We have tested the proposed algorithm on the 'Athens trucks' MOD containing 1100 trajectories. The dataset is available online on [20]. In most of the figures

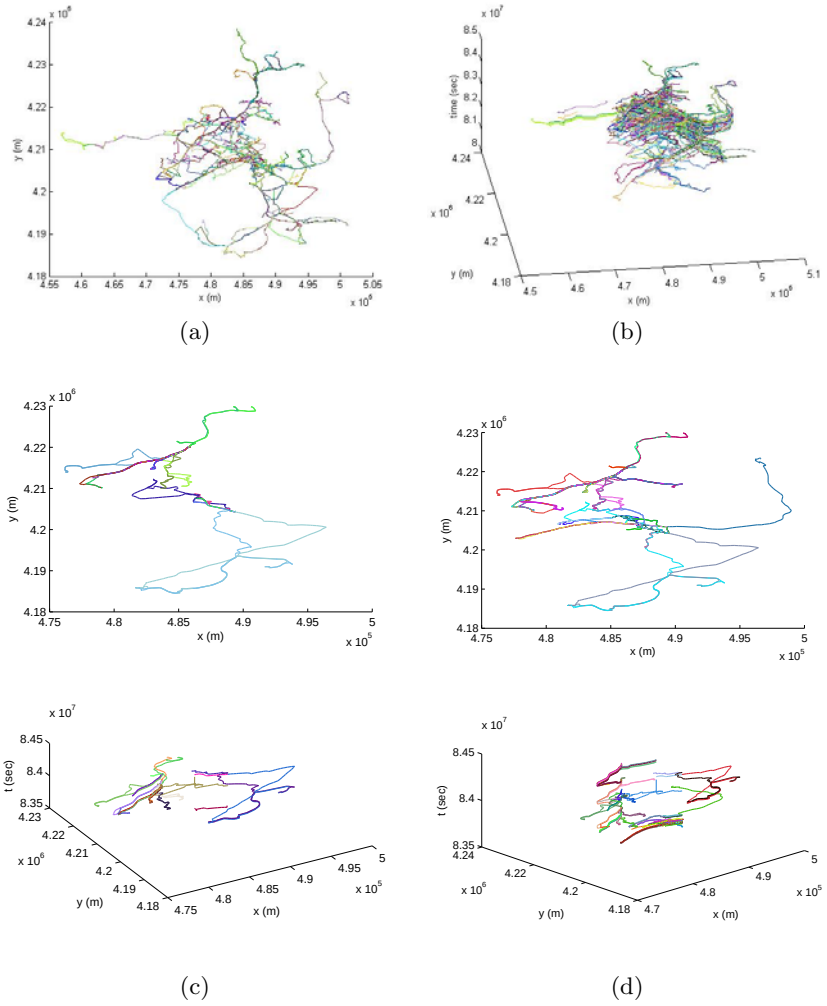
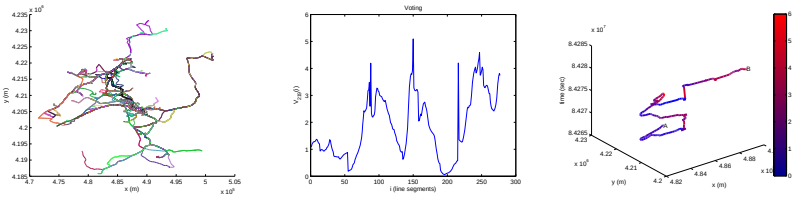


Fig. 2. The trajectories of our dataset (1100 traj.) projected in **(a)** 2-D spatial space ignoring time dimension and **(b)** spatiotemporal 3-D space. **(c), (d)** The 20 and 50 most representative trajectories of the dataset projected in 2-D spatial space (up) and in 3-D spatiotemporal space (down), respectively.

we have depicted a subset (10% or 20%) of the trajectories of our dataset, due to visualization issues. Fig. 2 illustrates the trajectories of our dataset projected in 2-D spatial space ignoring time dimension (Fig. 2(a)) and in spatiotemporal 3-D space (Fig. 2(b)). The provided information of Figs. 2(a) and 2(b) can not be visualized efficiently, due to the large number of projected trajectories in almost the same time and space. In order to solve this problem, we have used the results of classification to sample the dataset. Figs. 2(c) and 2(d) illustrate an efficient sampling/visualazation of the dataset using the 20 and 50 most representative trajectories according to E_k criterion, respectively. According to the proposed method, the estimated representative trajectories have the property to be close to many other trajectories of the dataset and can be used efficiently to visualize them. In our framework we have used the weights $w_1 = 1/1000$, $w_2 = 1/30$ (see Equation 4) and $\sigma = 2.5$ (see Equation 5).

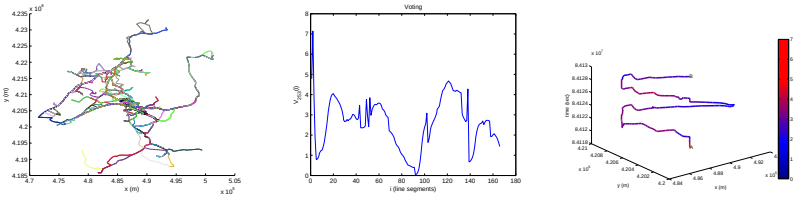


(a) (b) (c)

Fig. 3. Results of GVA for 219th trajectory of the dataset. **(a)** The 219th trajectory (bold black color) and some trajectories of our dataset projected in 2-D spatial space. **(b)** The voting descriptor V_{219}. **(c)** The 219th trajectory in 3-D space. The used colors correspond to the values of V_{219}, (red color for high values, blue color for low values).

Figs. 3 and 4 show the results of GVA for the trajectories 219 and 253 of the dataset, respectively. Figs. 3(a) and 4(a) show with bold black color the trajectories 219 and 253 and some of the trajectories of the dataset projected in 2-D spatial space. The estimated voting descriptors V_{219} and V_{253} are illustrated in Figs. 3(b) and 4(b). As it was mentioned before, the estimated voting descriptors change continuously over the trajectory segments. Figs. 3(c) and 4(c) illustrate the trajectories 219 and 253 in 3-D using a blue-to-red color map according to the corresponding to segments voting values (red color for high values, blue color for low values). By the analysis of these figures, it can be observed that the most representative paths of the trajectory 219 are found at the middle and at the end of the trajectory, while the most representative path of the trajectory 253 is found at the start of the trajectory. Moreover, the maximum values of V_{219} and V_{253} descriptors shows how many trajectories are close to the most representative paths of 219 and 253 trajectories, respectively.

Fig. 5(a) illustrates the classification results for 220 trajectories of our dataset projected in 2-D spatial space using E_k descriptor. The used colors correspond



(a) (b) (c)

Fig. 4. Results of GVA for 253th trajectory of the dataset. **(a)** The 253th trajectory (bold black color) and some trajectories of our dataset projected in 2-D spatial space. **(b)** The voting descriptor V_{253}. **(c)** The 253th trajectory in 3-D space. The used colors correspond to the values of V_{253}, (red color for high values, blue color for low values).

to the class of the trajectory (red color for representative trajectories). The most representative trajectory of the dataset is illustrated with red bold line. Similar results are obtained using M_k descriptor (see Fig. 5(b)). The most representative trajectories of the dataset are detected close to the center of the dataset, where most of the trajectories are crossed. Fig. 5(c) illustrates the classification results for trajectories line segments of 110 trajectories of the dataset projected in 2-D spatial space. The line segments with $V_k(i)$ greater than 10 are illustrated with red colors. The voting descriptor of the most representative line segment of the dataset has the value of 67.4. These figures are very useful for traffic monitoring, since they efficiently the trajectories and the segments, where the traffic is high.

5 Conclusions

In this paper, we have discussed the trajectory voting and classification problems in real spatiotemporal MOD. We have proposed an algorithm for trajectory voting and classification based on local trajectory similarity. Finally, a local trajectory descriptor per trajectory segment is estimated, that changes continuously over the trajectory segments. By the analysis of this descriptor the representative paths of the trajectory can be detected, that followed by many objects at almost the same time and at the same place. These results have been used to visualize a MOD. We have tested the proposed method under real databases and the experimental results shows that the method provides an efficient local (per segment) and global (per trajectory) classification of the dataset.

As future work, we plan to apply the voting results for trajectory segmentation, sampling, searching and retrieval. Segmentation and clustering algorithms can be applied on trajectory descriptor V_k providing a trajectory segmentation and a clustering of the dataset. Moreover, we plan to associate the voting results with an error function in order to measure the performance of the proposed sampling and to make comparisons with other works.

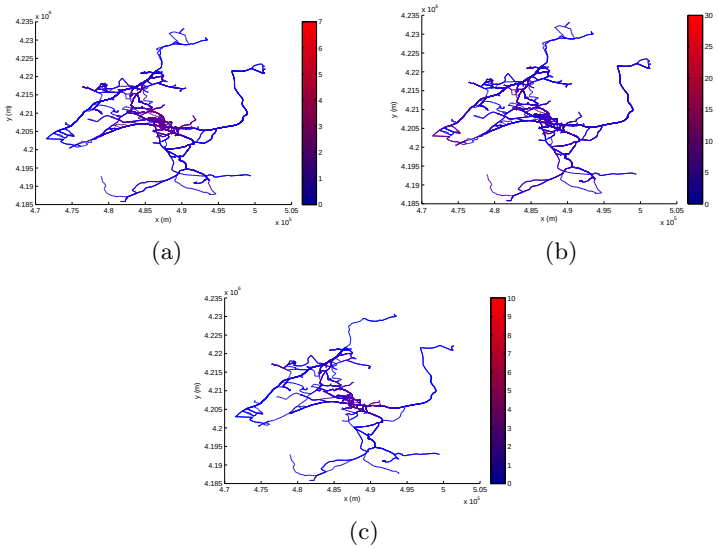


Fig. 5. Classification results for 220 trajectories of our dataset projected in 2-D spatial space using **(a)** E_k descriptor **(b)** using M_k descriptor, respectively. The used colors correspond to the class of the trajectory (red color for representative trajectories). **(c)** Classification results for trajectories line segments.

References

1. Giannotti, F., Nanni, M., Pinelli, F., Pedreschi, D.: Trajectory pattern mining. In: KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining. (2007) 330–339
2. Wang, L., Hu, W., Tan, T.: Recent developments in human motion analysis. Pattern Recognition **36**(3) (2003) 585–601
3. Giannotti, F., Pedreschi, D.: Geography, mobility, and privacy: a knowledge discovery vision. Springer (2007)
4. Vlachos, M., Gunopulos, D., Das, G.: Rotation invariant distance measures for trajectories. In: KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. (2004) 707–712
5. Chen, L., Özsu, M.T., Oria, V.: Robust and fast similarity search for moving object trajectories. In: SIGMOD '05: Proc. of the 2005 ACM SIGMOD int. conf. on Management of data. (2005) 491–502
6. Lee, J.G., Han, J., Whang, K.Y.: Trajectory clustering: a partition-and-group framework. In: SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data. (2007) 593–604
7. Lee, J.G., Han, J., Li, X., Gonzalez, H.: Traclass: trajectory classification using hierarchical region-based and trajectory-based clustering. PVLDB
8. Panagiotakis, C., Ramasso, E., Tziritas, G., Rombaut, M., Pellerin, D.: Shape-based individual/group detection for sport videos categorization. IJPRAI **22**(6) (2008) 1187–1213

9. Panagiotakis, C., Ramasso, E., Tziritas, G., Rombaut, M., Pellerin, D.: Shape-motion based athlete tracking for multilevel action recognition. In: Proc. of AMDO 2006. (2006) 385–394
10. Giannotti, F., Nanni, M., Pedreschi, D.: Efficient mining of sequences with temporal annotations. In: n Proc. SIAM Conference on Data Mining. (2006) 346–357
11. Anagnostopoulos, A., Vlachos, M., Hadjieleftheriou, M., Keogh, E., Yu, P.S.: Global distance-based segmentation of trajectories. In: KDD '06: Proc. of the 12th ACM SIGKDD int. conf. on Knowledge discovery and data mining. (2006) 34–43
12. Pelekis, N., Kopanakis, I., Marketos, G., Ntoutsi, I., Andrienko, G., Theodoridis, Y.: Similarity search in trajectory databases. In: TIME '07: Proc. of the 14th Int. Symposium on Temporal Representation and Reasoning. (2007) 129–140
13. Sacharidis, D., Patroumpas, K., Terrovitis, M., Kantere, V., Potamias, M., Mouratidis, K., Sellis, T.: On-line discovery of hot motion paths. In: EDBT '08: Proc. of the 11th int. conf. on Extending database technology. (2008) 392–403
14. Andrienko, G., Andrienko, N., Wrobel, S.: Visual analytics tools for analysis of movement data. SIGKDD Explor. Newsl. **9**(2) (2007) 38–46
15. Rubner, Y., Tomasi, C., Guibas, L.J.: A metric for distributions with applications to image databases. In: ICCV '98: Proceedings of the Sixth International Conference on Computer Vision. (1998)
16. Shishibori, M., Tsuge, S., Le, Z., Sasaki, M., Uemura, Y., Kita, K.: A fast retrieval algorithm for the earth mover's distance using emd lower bounds. In: IRI. (2008) 445–450
17. Lumelsky, V.J.: On fast computation of distance between line segments. **21** (1985) 55–61
18. Patterson, D.: Artificial Neural Networks. Prentice Hall (1996)
19. Yuan, J., Bo, L., Wang, K., Yu, T.: Adaptive spherical gaussian kernel in sparse bayesian learning framework for nonlinear regression. Expert Syst. Appl. **36**(2) (2009) 3982–3989
20. : URL: http://infolab.cs.unipi.gr/pubs/tkde2009/.