Anomaly Detection Based on Trajectory Analysis Using Kernel Density Estimation and Information Bottleneck Techniques

Yuejun Guo, Qing Xu*, Yu Yang, Sheng Liang, Yu Liu, Mateu Sbert

Abstract

In this paper, we propose a new technique to enhance the trajectory shape analysis by explicitly considering the speed attribute of trajectory data, as an effective and efficient way for anomaly detection. An object motion trajectory is mathematically represented by the *Kernel Density Estimation*, taking into account both the shape of the trajectory and the speed of the moving object. An unsupervised clustering algorithm, using the *Information Bottleneck* method, is employed for the trajectory learning to achieve an optimal number of trajectory clusters through maximizing the *Mutual Information* between the clusters and a feature space of the trajectory data. The trajectories are determined as either abnormal (infrequently observed) or normal by a measure based on *Shannon* entropy. Extensive tests on simulated and real data show that the proposed technique behaves very well and outperforms the state-of-the-art methods. To the best of our knowledge, this is the first technique to use the *Information Bottleneck* method on the trajectory data.

Index Terms

Trajectory shape analysis, trajectory clustering, anomaly detection, *Kernel Density Estimation*, *Mutual Information*, *Shannon* entropy.

I. Introduction

IN the present society, the anomaly detection technique [1] has become a compelling topic in many fields, such as video surveillance for the sake of safety and security [2] [3]. In particular, a trajectory, embodying the motion characteristics of an object, has been a critical factor for anomaly detection and actually, the learning and analysis of the motion trajectory data are robust and feasible for detecting abnormal activities, for example, in the setting of traffic and pedestrian monitoring [4].

A lot of methods based on the motion trajectories for detecting anomaly that have been proposed work quite well, and unsupervised techniques are nowadays becoming the most important deployments: here we refer readers to two recent reviews [5] [4]. A typical unsupervised approach to the trajectory-based anomaly detection involves the necessary steps as follows. At first, a motion trajectory is quantitatively depicted by a possible model, for the ease of further processing. Then the trajectory patterns, given the training data, are distinguished with an unsupervised clustering procedure without the need of any prior knowledge. According to a certain number of patterns that have been learned, the testing trajectories are identified as either normal or abnormal, namely, either frequently observed or not.

Among the unsupervised trajectory-based anomaly detection approaches having been proposed so far, those depending upon the shape analysis of the trajectory data play an important role [6]: actually the shape itself, which pays much attention to the global description of a motion trajectory, is very suitable and successful for the abnormal activities detection [7]. Specifically and typically, the shape of a motion trajectory is characterized by a series of the tangential angles at the positions of the corresponding moving object (for the sake of easy description, in this paper, we call these positions as the sample points of the trajectory), and these angles are modeled by using a circular distribution, *von Mises* [6] [7], to represent the trajectory. Despite the fact that the angle attribute is concise and largely helpful for a lot of practical applications, the only use of this attribute cannot discriminate, for instance, whether a person is walking or running in an automated surveillance system. Even when both the angle and speed attributes are considered, a trajectory is defined resorting to some classical distributions, such as an Approximated Wrapped and Linear Gaussian (*AWLG*) [8], relying on the assumption that these distributions fit the real trajectory data. However, in effect it is not clear whether this fitting is true or not.

This paper proposes a novel effective technique to analyze whether a trajectory is either reoccurring or not, namely, either normal or anomalous. The work presented here is an improvement of the classical anomaly detection methods based only on trajectory shape analysis using (semi-)circular statistics [6] [7]. Our main contributions are as follows:

– We explicitly make use of the speed of the moving object, leading to a richer representation than most of the previous ones relying solely upon the angle description.

– We utilize *Kernel Density Estimation* (*KDE*) [9] to more accurately model the trajectory data, rather than fitting some classical parametric probability function to the true distribution of these data.

Y. Guo, Q. Xu, Y. Yang, S. Liang and Y. Liu are with Tianjin University, China. M. Sbert is with University of Girona, Spain. Qing Xu (qingxu@tju.edu.cn) is the corresponding author.

– We propose to make use of the *Information Bottleneck* (*IB*) principle [10] to perform the clustering on the trajectory data, avoiding the use of an explicit distance measure between two trajectories in most of the existing approaches for trajectory analysis. Thus, the difficulty to obtain a good metric for directly measuring the trajectory distance is overcome. Based on the minimization of the loss of *Mutual Information* (*MI*) [11] between the trajectory dataset and a feature space of these trajectories, an optimum number of the trajectory clusters are achieved (for conciseness we call this the *MI* based clustering in our paper).
– Instead of using a "hard" threshold, for the evaluation of the differences between a trajectory being tested and the medoids of the learned clusters, we employ an adaptive decision, by deploying the *Shannon* entropy concept [11], to look at all the differences under consideration as a whole and to make the trajectory analysis more discriminative.

The rest of this paper is organized in this way: Section II covers the related work. An overall look of our proposed technique is described in Section III. *KDE* modeling, *MI* based clustering and *Shannon* entropy based detection are depicted in the following sections. And finally, before the concluding remarks, experiments on simulation and real test data are thoroughly discussed.

II. Related Work

Typically, information and attributes contained in video trajectories are of great value [12]. Furthermore, general motion trajectories, as the important meta-data, have been extensively utilized for anomaly detection in many diverse application fields, such as air traffic [13] and vehicular ad hoc networking [14]. Therefore, numerous trajectory-based analysis methods have been developed so far; for the purpose of brevity, the interested readers are directed to the two example surveys for the detailed review [5] [4]. Roughly speaking, there exist two kinds of trajectory analysis techniques, namely supervised and unsupervised.

In general, supervised approaches require labeled training trajectory data. For example, Hidden Markov Model (*HMM*) is a typical supervised approach for the representation of trajectory data [15] [16] [17]. *HMM* is powerful enough to capture the underlying structure of the trajectories with various lengths; however, the establishment of *HMM* is application specific and correspondingly, the model fitting required is too complicated when the motion trajectories are highly complex, as pointed out in [7] [6]. Actually in essence, the difficult fitting is a general problem for a parametric model like *HMM*. Another classical supervised technique for anomalous detection is based on Support Vector Machine (*SVM*), facilitating the anomaly detection and removal in the training stage [18]. In this paper, we focus the attention on the unsupervised methods, considering the motion trajectories without any labels that are usually encountered in many real scenarios.

As for the unsupervised algorithms, the neural network models such as the Self-Organizing Map (*SOM*) [19] and its fuzzy version [12], and the Learning Vector Quantization (*LVQ*) [20] have been employed for the sake of anomalous detection. However, as indicated in [21], the implementation of the neural network is a big problem.

In fact, as mentioned in Section I, most of the existing unsupervised methods can approximately involve three phases, namely, the modeling of trajectory data, the clustering on the trajectories and the detecting of abnormal trajectories. Due to the noise in the trajectory data during their extraction procedure, it is reasonable to model the motion trajectories in a probabilistic way [22] [7] [8] [6]. The Mixture of Gaussians (*MoG*, also called Gaussian Mixture Model (*GMM*)) [22] and its variant, the Mixture of *von Mises* (*MovM*) [7] [6], are two widely used models for the representation of trajectories. Besides, the Mixture of Approximated Wrapped and Linear Gaussian (*MoAWLG*) [8] is also used for modeling the trajectory data. All these statistical representations are (semi-)parametric: but as a matter of fact, it is not clear whether the (semi-)parametric models used can fit the real trajectory data or not. Considering this, our paper exploits the nonparametric *KDE* for representing the motion trajectories. As for the trajectory attributes, the spatial position, speed, acceleration and size of the moving object, the curvature and shape of the trajectory, and so on [23][24] may be taken into account, depending on the potential applications based on the trajectory analysis. We argue in this paper that the shape and the speed are the two major factors: the former is to globally characterize the motion trajectory and the latter is to locally describe the moving object. The combination of the shape and speed is an important descriptor for the general motion trajectories to have extensive usage in a lot of applications. In fact, the two-dimensional (2D) *KDE* being presented in this paper for the modeling of trajectory data will be generalized to cover more trajectory attributes, as our future work. It is worthwhile to point out that some techniques for dimension reduction, such as the Principal Component Analysis (*PCA*) [17], the Independent Component Analysis (*ICA*) [25], the Discrete Fourier Transformation (*DFT*) [20], the Discrete Wavelet Transformation (*DWT*) [26] and the spline approximation [27], have been used to help model the motion trajectories for relieving the computational burden of the trajectory analysis. We believe that we can combine together the nonparametric *KDE* and some dimension reduction technique to improve the trajectory representation: this will be considered in future work.

For the trajectory clustering, many methods such as the (fuzzy) k-means [28], the k-medoids [6], the spectral clustering [29], the graph-cuts [23], the mean shift [24] and the hierarchical clustering [30], have been demonstrated to work well for this task. All these clustering algorithms either require the explicit measure of distance between trajectories or need a predetermined number of clusters, or even both; but usually, either a good definition of the distance measure, or a predetermination of an optimum number of clusters, is not so easy [4][30]. For the sake of completeness, the often used measures of trajectory distance, such as the classic Euclidian distance and its variants [28][18], the Dynamic Time Warping (*DTW*) [31], the Longest Common

Sub-Sequence (*LCSS*) [32], the (modified) Hausdorff distance [29] and so on, are also referred here. Actually, these different distance measures have been shown to result in a similar performance based on the extensive tests for the trajectory analysis [33][34]. And this exactly implies that it is difficult to obtain a very general and good measure of the trajectory distance. Alternatively, in this paper, we make use of the *IB* to achieve an agglomerative hierarchical clustering, with an optimum number of clusters, on the motion trajectories modeled in a nonparametric manner, avoiding the use of an explicit measure of trajectory distance. Due to its derivation from a very general variational principle in the context of an information-theoretic approach [11], the *IB* technique has strong theoretic soundness, making the clustering procedure more stable [35].

Considering the abnormality detection, generally the key is based on a predefined threshold for deciding whether the trajectory being tested is anomalous or not [5]. Apparently, a good threshold should be data adaptive. In this paper, the *Shannon* entropy is utilized to meet this requirement, as inspired by its successful and popular usage in a large variety of academic areas, such as video processing [36]: as a very recent example, the *Shannon* entropy has been well demonstrated for the video key frame selection and its visualization [37].

Notably, the trajectory-based anomaly detection supporting the incremental training and updating is a very potential approach [28][21], though this approach could still have the problem in the context of the long-term updating training data. In the near future, the underlying ideas presented in this paper will be investigated to devise an incremental-based strategy to enhance the current implementation.

III. The Overall Proposed Technique

Our motivation for doing the trajectory based anomaly detection is on the premise that we do not have any labels on the trajectories in many practical situations. So, naturally, we perform the trajectory analysis, in an unsupervised way, with two components, training and testing. Figure 1 gives a complete procedure of the proposed approach exemplified with a test dataset (Section VII-B).

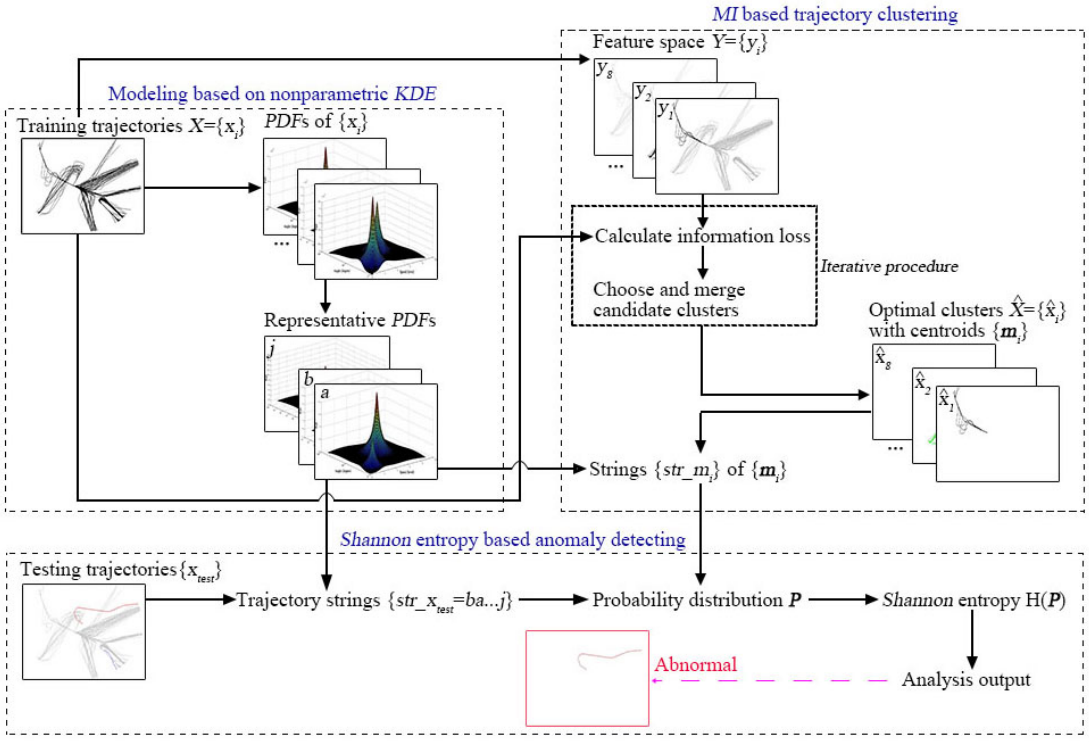


Fig. 1. The overview of our proposed technique.

Given the object motion trajectories being trained, a 2D *KDE* technique [9] [38] [39] is adopted, considering the shapes of trajectories and the speeds of the moving objects, to establish the 2D *Probability Density Functions* (*PDF*s) of these trajectory data. All the *PDF*s are grouped to give a number of representative *PDF*s. With a feature space obtained from the trajectory data, the *MI* based clustering is performed through an iterative procedure to obtain the optimal trajectory clusters to be used for the further anomaly testing. Here, the iterative procedure is used to calculate the information loss of every two clusters and then choose the candidate clusters to merge.

In the testing stage, each trajectory, which is composed of its sample points and is correspondingly defined as a sequence of pairs of speed and angle, is coded into a string based on the representative *PDF*s for the sake of handling the order of

these pairs. In addition, we code the cluster medoids obtained by the *MI* based clustering to build a probability distribution for each testing trajectory by calculating the distances between the string of this trajectory and those of the medoids. Based on the distributions, the *Shannon* entropy for each testing trajectory is obtained to achieve the final analysis output, namely to indicate that a trajectory being tested is either anomalous or not. As usually done, the number of the trajectories in the anomalous cluster is set below a small percent of total number of all the trajectories. Some dynamic change in the trajectory clusters may occur along with the process of the *Shannon* entropy based detection, and we call this the local update of the trajectory clusters. For the practical use of our proposed approach to trajectory analysis based anomaly detection, all the trajectory clusters are globally updated from scratch periodically.

IV. Trajectory modeling based on nonparametric *KDE*

Although the shape attribute is simple but useful for the trajectory based analysis, some other attributes such as the speed of the trajectory are required in many application scenarios. For instance, imagine that for the sake of safety we need to monitor the speeds of vehicles on roads based on the trajectory data analysis, and undoubtedly the use of the trajectory speed is essential for this case. As a result, besides the use of the shape of a trajectory, the speed of the corresponding moving object is additionally and explicitly taken into account, in this paper, for mathematically representing the trajectory.

Considering the fact that the trajectory data are usually uncertain and noisy in real-world contexts, we model the trajectories statistically, as usually done in the field of anomaly detection [1]. That is, we believe that normal trajectory instances happen with high probabilities while anomalies occur with small probabilities. In concrete, generally, a trajectory instance, which can be reasonably deemed to be independently identically distributed, is considered being generated according to a probability distribution function: identifying this function is the key to the trajectory modeling. Rather than assuming that some classical distributions such as *von Mises* [6] [7] and *AWLG* [8] fit the real trajectory data, we utilize the nonparametric *KDE* to estimate the probability distribution function, due to that *KDE* is so powerful to be able to better capture the data without needing a priori knowledge of them [9] [38] [39]. In this paper, the speed and angle of the sample points of a trajectory are both used to establish the probability distribution, or explicitly, the *PDF*, for the model representation of this trajectory. In this way, our model representation describes the trajectory data very well and, we are going to demonstrate the advantage of deploying *KDE* for modeling the trajectories in Section IV-A.

Let $x = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_n\}$ be a trajectory, where $\boldsymbol{s}_i = (v_i, \theta_i)^T$ is a sample point represented by a pair of speed v_i and angle θ_i. Here v_i relies on the distance and time interval between this sample point and its previous adjacent one, and θ_i is determined by the tangential direction at this sample point. The estimated *PDF* for a trajectory is determined according to the multivariate kernel density estimator [39]

$$z(\boldsymbol{s}) = n^{-1} \sum_{i=1}^{n} \{|\mathbf{H}|^{-1/2} K[\mathbf{H}^{-1/2}(\boldsymbol{s} - \boldsymbol{s}_i)]\}, \tag{1}$$

where $\boldsymbol{s} = (v, \theta)^T$ denotes a general sample point of a trajectory. Here $\mathbf{H}$ is a 2×2 diagonal bandwidth matrix, and is set as $\begin{pmatrix} \hat{\sigma_v}^2 n^{-2/5} & 0 \\ 0 & \hat{\sigma_\theta}^2 n^{-2/5} \end{pmatrix}$ where $\hat{\sigma_v}^2$ and $\hat{\sigma_\theta}^2$ are respectively the sample variances for the speed and angle data, as usually suggested for *KDE* [38]. K is a kernel function, our selection for this is simply the very widely used Gaussian density [9]:

$$K(\boldsymbol{s}) = \frac{1}{2\pi} exp(-\frac{1}{2}\boldsymbol{s}^T\boldsymbol{s}). \tag{2}$$

Given all the trajectory data for training, *KDE* is used to model these data into *PDF*s. Then all the trajectory *PDF*s are clustered into a number of groups. We make use of the classical k-medoids algorithm [40] and the Bhattacharyya coefficient [41] respectively for the clustering operation and for the distance measure used in the clustering procedure, inspired by their successful demonstration in the previous trajectory analysis work [6] [7]. The medoid of a clustered group is determined as one of all the trajectories in this group to minimize the sum of distances between it and all the others. The medoids for the clustered groups are taken as the representative *PDF*s for all the sample points of all the training trajectories.

Based on the representative *PDF*s, the testing trajectories and the cluster medoids obtained by *MI* based clustering (Section V) are later transformed into strings for the purpose of dealing with the order of their speed and angle pairs. In our implementation, each representative *PDF* is one-to-one assigned to a single character. For a sample point of a trajectory, its pair of the speed and angle $(v, \theta)^T$ is encoded with a character such that the representative *PDF* corresponding to this character is maximized by $(v, \theta)^T$. For example, suppose we have 10 representative *PDF*s denoted by $\{a, b, \ldots, j\}$, a sample point $\boldsymbol{s} = (v, \theta)^T$ of the trajectory x is encoded by $\underset{u \in \{a,b,\ldots,j\}}{\text{argmax}} \; PDF_u(\boldsymbol{s})$, which is the character corresponding to the representative *PDF* maximized by $\boldsymbol{s} = (v, \theta)^T$. As a result, a trajectory x, with n sample points, turns into a string as

$$str\_x = \bigcup_{i=1}^{n} \underset{u \in \{a,b,\ldots,j\}}{\text{argmax}} \; PDF_u(\boldsymbol{s}_i). \tag{3}$$

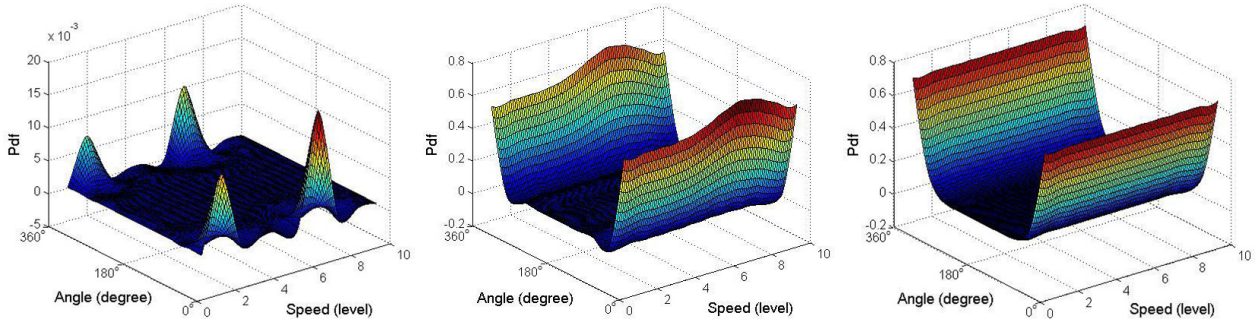
Note str_x is an ordered string, and the order is the same as that of the sample points.

It is worthy to indicate that the trajectory modeling in this paper is meant in essence to statistically represent all the sample points of all the trajectories according to the (representative) *PDF*s. This is different from the way used in [8][6], because we obtain the *PDF*s based on the nonparametric *KDE*, rather than on the parameter fitting for some assumed model distribution.

A. Illustration for the estimated PDFs by the different modeling schemes

In this part, we apply the plot and visualization metaphors to compare the modeling efficiency of *KDE*, *MoAWLG* and *MovM*.

1) Plots of the PDF surfaces: We design a simulation object trajectory to exemplify the different capabilities of *KDE*, *MoAWLG* and *MovM* for trajectory modeling. This simulation trajectory involves 79000 sample points. All the angles at the trajectory sample points are between 0° and 360°, and most of them are close to either 0° or 360°. The speeds of the moving object vary around either level 2 or level 7 (the speed levels in all are $0 \sim 9$ here). Figure 2(1) and Figure 2(2) respectively illustrate the *PDF*s estimated by *KDE* and by *MoAWLG*. Obviously, the *PDF* obtained by *KDE* closely reflects the distribution characteristics of the speeds and angles of the trajectory sample points. By contrast, the *PDF* evaluated by *MoAWLG* cannot faithfully represent the speed characteristic of the designed trajectory, though it can express the angle distribution of the trajectory sample points. We believe this is due to that *KDE* takes the nonparametric statistical way to correctly model a motion trajectory, avoiding the resort to the assumption that the parametric *AWLG* fits the true *PDF*. In practice, this means that *KDE* does not need to deal with the necessary but difficult model fitting when a parametric function is used, resulting in the better behavior.



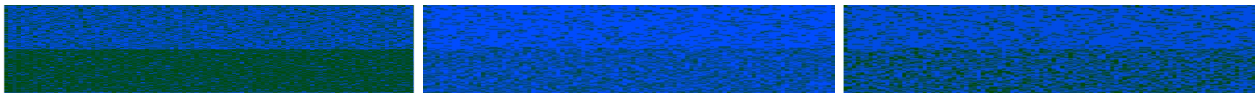
(1) KDE (2) $MoAWLG$ (3) $MovM$

Fig. 2. The plots of the *PDF*s by the three modeling schemes.

As for *MovM*, it just focuses on a single univariate distribution, namely the angle distribution, of the trajectory sample points. Based on using *MovM*, we can basically obtain a 2D plot of the angle distribution for the trajectory sample points with a given single speed value. Then, for comparison, a 3D plot, which integrates all the 2D plots based on all the possible speed values, is shown in Figure 2(3).

2) Visualizations of the coded trajectory strings: Here, we specially devise 300 simulation trajectories and each trajectory has 100 sample points. All the trajectories are with the very similar angles around 0°. Considering that the proposed modeling by *KDE* explicitly utilizes the speed attribute of the motion trajectory, the sample points for the first 150 simulated trajectories are with large speeds, while those for the others have small speed values.

Coded strings, as mentioned above, which are the products of the trajectory modeling, are visualized with colors to visually show the advantage of our *KDE* based modeling technique. For each coded character of a trajectory sample point, we use a small color rectangle for its visualization. All the color rectangles are arranged in sequence to form a line with colors to represent a motion trajectory, as shown in Figure 3.



(1) KDE (2) $MoAWLG$ (3) $MovM$

Fig. 3. The visualizations for the three modeling schemes. The modeling by *KDE* (1) clearly separates two trajectory groups respectively with the two largely different speed attributes. The modelings by *MoAWLG* (2) and *MovM* (3) are barely satisfactory.

We use two RGB colors, blue and green, to do the visualization for the coded characters. In this experimentation, there exist ten different coded characters that correspond to the ten representative *PDF*s obtained in the procedure of the *KDE* based modeling. The ten coded characters are descendingly sorted according to the speed values of their corresponding representative *PDF*s. The speed for a representative *PDF* refers to that for the trajectory corresponding to this *PDF* (here for simplicity, an average speed of all the sample points for a trajectory is taken as the motion speed for the whole trajectory). The first five sorted coded characters are respectively visualized employing pure blue colors in five different brightness, $(0, 0, 1)$, $(0, 0, 0.8)$,

$(0, 0, 0.6)$, $(0, 0, 0.4)$ and $(0, 0, 0.2)$. The other five sorted coded characters are respectively visualized using five pure green colors in five different brightness, $(0, 1, 0)$, $(0, 0.8, 0)$, $(0, 0.6, 0)$, $(0, 0.4, 0)$ and $(0, 0.2, 0)$. Analogously, two distinctly different pure colors, blue and green, and for each five small different ones respectively with different brightness levels are applied to visualize the trajectory sample points modeled by *MoAWLG* and *MovM*.

From the visualization comparison demonstrated in Figure 3, it is obvious that the modeling by *KDE* clearly discriminates the two categories of the trajectories respectively with fast and slow speeds. This shows the merit of explicitly using the speed attribute for a motion trajectory, and also of exploiting the nonparametric approach, the *KDE*. The modelings by *MoAWLG* and *MovM* are inferior to that by *KDE* but acceptable, because our designed simulation trajectories are in fact very difficult for these two parametric schemes. Note that in effect the modeling by *MovM* can implicitly consider the speed factor of a trajectory to some degree, as has been indicated by the inventors of this approach [7].

V. *MI* BASED TRAJECTORY CLUSTERING

For the task of trajectory learning, there usually exist two important and difficult problems. Undoubtedly the number of clusters is critical for the clustering performance, however, finding an optimal number of the learned clusters is a challenging issue [30]. Another challenge is that it is difficult to define an explicit distance measure between two trajectories to be used in the clustering process [33][34]. Fortunately, the *IB* technique [10] can be taken advantage of for coping with the two problems.

A. The feature space used by IB

According to the *IB* framework, an appropriate clustering is to minimize the loss of *MI* between the input data and the so-called feature space of these data [10]. That is, the *IB* based clustering maintains the relevant "features" in the input data with respect to the feature space. Or, equally speaking, the clustering on the input data based on *IB* is "regularized" by the feature space of these data. As a matter of fact, the feature space of the input data is composed of the disjoint sets (note, sets are called elements for easy explanation in the following) of these data and, the elements here are usually obtained according to the data features under consideration.

First of all, we point out the necessity and importance of obtaining a feature space of the trajectory data, in order to use *IB* for clustering these data. Fundamentally, the attainment of the feature space should be unsophisticated and actually, some direct and simple trajectory attributes such as the positions of trajectories can be used for this purpose. Figure 4 gives an example of obtaining the feature space Y for a simulation dataset with 150 trajectories. Obviously, all the trajectories here are naturally partitioned according to their positions.

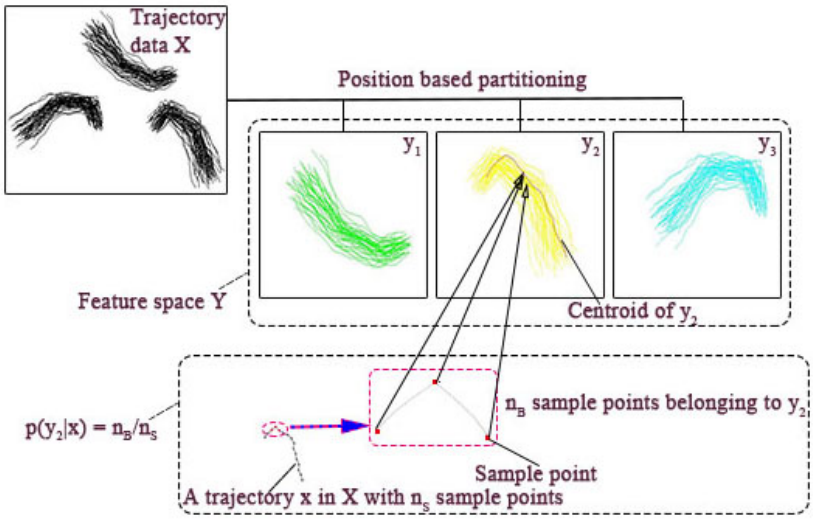


Fig. 4. The classical k-means with the Euclidian distance between trajectories is used to establish the position based partition of the trajectory dataset X (this dataset is contributed by Piciarelli *et al.* [18]), obtaining the feature space $Y = \{y\} = \{y_1, y_2, y_3\}$. For each trajectory $x \in X$, the conditional probability $p(y|x)$ is calculated for the *IB* procedure.

Notice that, besides the simple application of the natural attribute based partitioning on the trajectory data to obtain a feature space of them, it is, furthermore, better to make use of some a priori knowledge, such as some physical meaning, behind the trajectory data for the obtainment of the feature space. Generally, it is a common observation that the trajectories in a practical scenario usually have their special physical meaning. For instance, the aircraft trajectory data, as shown in Figure 9 (see details on this test data in Section VII-B), approaching from the specified directions and landing on the predefined runways, present some similar patterns for landing. As a matter of fact, the aircraft trajectories, corresponding to a terminal area, reflect their

own flight operations. Here the operation is either landing or taking off. As a result, for the input aircraft trajectory data, the distribution of these trajectories according to their flight operations, which constitutes a feature space of the data, can be obtained. And clearly, this feature space correlates to the input data.

B. The trajectory clustering based on IB

Without loss of generality, given a set of trajectory data $X = \{x\}$, we can always establish a feature space $Y = \{y\}$ of X and, X has correlation with Y. Here X and Y are both considered as random variables. Thus in this way, we develop an information channel between X and Y. The three basic components of the information channel are as follows:
– Input probability $p(x)$. This is the probability of a single trajectory. We simply use $p(x) = 1/n_T$ to assign the uniform "importance" for all the trajectories under consideration, here $n_T = |X|$.
– Conditional probability $p(y|x)$. Assuming that a given y involves k trajectories $\{x_1, x_2, \ldots, x_k\}$, we obtain $p(y|x)$ based on the centroid trajectory of y. The centroid trajectory is achieved according to its *PDF*. As stated in Section IV, here each x_i ($i \in \{1, 2, \ldots, k\}$) can be represented by a *PDF* z_i (1). The *PDF* of the centroid trajectory of y, $PDF_y^{centroid}(\cdot)$, is taken as the one in $\{z_1, z_2, \ldots, z_k\}$ minimizing the summed distances between it and all the others

$$PDF_y^{centroid}(\cdot) = z_{\underset{i\in\{1,2,\ldots,k\}}{\operatorname{argmin}}\ [\sum_{j=1,j\neq i}^{k} d(z_i,z_j)]}, \tag{4}$$

where $d(z_i, z_j)$ is the distance, computed by Bhattacharyya coefficient [41], between z_i and z_j. Naturally, the centroid trajectory of y, $y_{centroid}$, is the trajectory corresponding to this $PDF_y^{centroid}(\cdot)$. Then for a sample point $\boldsymbol{s} = (v, \theta)^T$ of x, we speak that the sample point $\boldsymbol{s}$ belongs to the given y when

$$PDF_y^{centroid}(\cdot) = \underset{w\in Y}{\operatorname{argmax}}\, PDF_w^{centroid}(\boldsymbol{s}), \tag{5}$$

meaning that in this case (the centroid trajectory of) y gives the largest *PDF* value for this sample point $\boldsymbol{s}$. Finally, for all the n_S sample points of x, we can obtain

$$p(y|x) = \frac{n_B}{n_S}, \tag{6}$$

where n_B is the number of sample points of x belonging to y. Notice Figure 4 illustrates an example of how to obtain the conditional probability.
– Output probability $p(y)$. This probability is obtained by

$$p(y) = \sum_{x\in X} p(x)p(y|x). \tag{7}$$

Importantly, the *MI* $I(X;Y)$ of the information channel gives the quantitative correlation between X and Y [11]. Similarly, in the procedure of the *IB* based clustering, the correlation between the set of the trajectory clusters, $\hat{X}$, and Y is defined by $I(\hat{X};Y)$. An optimum number of the trajectory clusters can be obtained, according to the *IB* principle [10], through

$$max\{I(\hat{X};Y) - \alpha I(\hat{X};X)\}, \tag{8}$$

where α is the Lagrange multiplier. In practice, the solution is achieved with a greedy algorithm based on a bottom-up merging strategy by minimizing the loss of *MI*, $I(X;Y) - I(\hat{X};Y)$ [10].

The greedy algorithm is performed iteratively. In the beginning, each trajectory corresponds to a single cluster. In each iteration, two clusters are merged based on the minimal loss of information induced by this merging. Let $\hat{x}_1$ and $\hat{x}_2$ be the two candidate clusters to be merged, $\hat{x}_1, \hat{x}_2 \in \hat{X}$. The information loss caused by merging $\hat{x}_1$ and $\hat{x}_2$ is

$$\begin{aligned}\Delta I(\hat{x}_1,\hat{x}_2) &= I(\hat{X}_{before};Y) - I(\hat{X}_{after};Y)\\ &= \sum_{y,i=1,2} p(\hat{x}_i,y)\log\tfrac{p(\hat{x}_i,y)}{p(\hat{x}_i)p(y)} - \sum_{y} p(\hat{x}_{new},y)\log\tfrac{p(\hat{x}_{new},y)}{p(\hat{x}_{new})p(y)}\\ &= \sum_{y,i=1,2} p(\hat{x}_i)KL(p(y|\hat{x}_i)||p(y|\hat{x}_{new}))\end{aligned} \tag{9}$$

where $I(\hat{X}_{before};Y)$ and $I(\hat{X}_{after};Y)$ respectively denote the mutual information before and after the merging, $y \in Y$, and KL indicates the Kullback-Leibler divergence [42]. Here $\hat{x}_{new}$ is a new cluster obtained by merging $\hat{x}_1$ and $\hat{x}_2$. When the merging is done, the conditional probability is updated by

$$p(y|\hat{x}_{new}) = \frac{p(\hat{x}_1)}{p(\hat{x}_{new})}p(y|\hat{x}_1) + \frac{p(\hat{x}_2)}{p(\hat{x}_{new})}p(y|\hat{x}_2) \tag{10}$$

where $p(\hat{x}_{new}) = p(\hat{x}_1) + p(\hat{x}_2)$. Obviously, $p(\hat{x}_1) = 1/n_T$ and $p(\hat{x}_2) = 1/n_T$ when to start the clustering algorithm because $\hat{x}_1$ and $\hat{x}_2$ correspond to two single trajectories in X.

Through the *IB* algorithm, we obtain an optimal number of trajectory clusters, without the need of an explicit distance measure between trajectories. For the sake of trajectory anomaly detection (Section VI), we obtain the centroid trajectory for each cluster and achieve this totally analogous to the way we obtain the $y_{centroid}$ discussed above. The process of the bottom-up *IB* algorithm is summarized in Algorithm 1.

Algorithm 1: Bottom-up IB algorithm

Input:
Dataset X, feature space Y;
Conditional probability distribution: $p(y|x)$, $x \in X$, $y \in Y$.
Output:
$K-$ optimal clusters: $\hat{x}_i$, $i = 1, 2, \ldots, K$.

1) Initialize $\hat{X} = X$, $min\Delta I = 0, m, n$
2) **while** $\min \Delta I < Threshold$ **do**
3) $min\Delta I = \min_{\hat{x}_i, \hat{x}_j \in \hat{X}, i \neq j} \{\Delta I(\hat{x}_i, \hat{x}_j) | m = i, n = j\}$
4) $\hat{x}_{new}$ = Merge $(\hat{x}_m, \hat{x}_n)$
5) Update $\hat{X}$ = $\{\hat{X} - \{\hat{x}_m, \hat{x}_n\}\} \cup \{\hat{x}_{new}\}$
6) **end while**

Notably, the *Threshold* employed in Algorithm 1 is an adaptively configured threshold to terminate the iterative process of the clustering. In the *IB* clustering, merging two appropriate clusters causes the loss of mutual information. And essentially, the *Threshold* corresponds to an abrupt change of the information loss. As an example, Figure 5 shows a plot of the information loss versus the number of iterations, for the iterative *IB* processing on a real dataset with 320 aircraft trajectories (Section VII-B). The iteration number begins at 0 and ends at 319, and as a matter of fact this number corresponds to the number of clusters obtained in each iteration (the sum of these two numbers is 320, which is obviously the total number of the trajectories). The adaptive threshold, used as the termination condition for the *IB* algorithm, corresponds to the vertical axis value of the "Break point". Here the "Break point" is an abrupt change of the loss of mutual information. From this point on, the cluster merging results in a very large information loss and in a poor clustering, suggesting a terminating condition for the *IB* and an optimal number of clusters obtained.

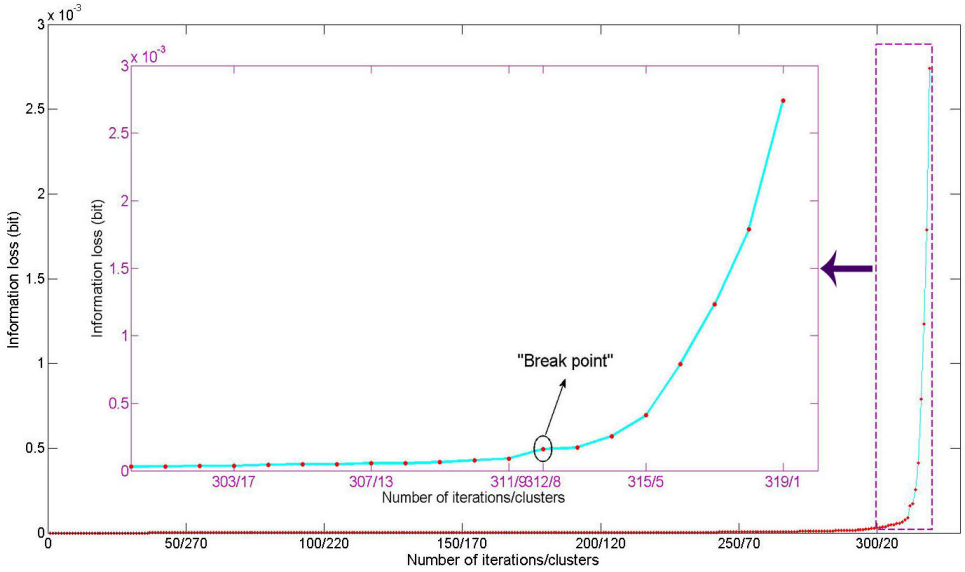


Fig. 5. The information loss happens in the *IB* process for a real trajectory dataset. Here the "Break point" indicates that the optimum number of clusters obtained is 8.

VI. *Shannon* ENTROPY BASED TRAJECTORY ANOMALY DETECTION

Classically, if the differences between a test trajectory and the cluster medoids are higher than a "hard" threshold, then this trajectory is classified as abnormal; otherwise the test trajectory belongs to a labeled group, which is possibly an anomalous cluster [6] [7]. However, it is infeasible to set the direct and "hard" threshold to do the classification for the possible quite large ranges of the trajectory differences in many and varied practical situations. As a matter of fact, the key observation for

an abnormal trajectory is that, among all the differences between this trajectory and all the cluster medoids, there is no one being significantly larger than the others. That is, the differences between the abnormal trajectory and the cluster medoids can be regarded as "approximately equal". In this paper, a probability distribution is constructed based on the differences between the trajectory to be tested and the cluster medoids and, the *Shannon* entropy [11] of this probability distribution is exploited to fulfill the task of anomaly detection. We make use of the *Shannon* entropy to estimate the homogeneity of all the differences between the considered trajectory and the medoids of the clusters learned. In this case, we make clear an adaptive situation for all the differences under consideration: this is opposed to the approach proposed before [6] [7], in which these differences themselves are just separately evaluated as whether large or small, by using some straightforward threshold. Consequently, our anomaly detection is more discriminating.

Suppose that we have a testing trajectory x_{test} and a set of the learned clusters represented with the trajectory medoids $\{\boldsymbol{m}_i\}, (i = 1, 2, \ldots, nc)$, here $\boldsymbol{m}_i$ is the medoid of the *i*-th cluster. First, we transform x_{test} and $\{\boldsymbol{m}_i\}$ into strings as str_x_{test} and $\{str\_m_i\}$ respectively. Then the distances, $\{\boldsymbol{D}_j\}, (j = 1, 2, \ldots, nc)$, between x_{test} and $\{\boldsymbol{m}_i\}$ are defined as the distances between str_x_{test} and $\{str\_m_i\}$ (see the details in the following on the trajectory distance employed). Further, a probability distribution can be obtained as $\boldsymbol{P} = \{\boldsymbol{p}_i = \frac{\boldsymbol{D}_i}{\sum_{j=1}^{nc} \boldsymbol{D}_j}(i = 1, \ldots, nc)\}$, and the *Shannon* entropy of this distribution is

$$H(\boldsymbol{P}) = -\sum_{i=1}^{nc} \boldsymbol{p}_i \log \boldsymbol{p}_i. \tag{11}$$

If $H(\boldsymbol{P}) > \lambda \cdot \log(nc)$ then x_{test} is identified as abnormal; otherwise x_{test} belongs to the cluster with the medoid $\boldsymbol{m}_{id}$ meeting $\boldsymbol{D}_{id} = \min\{\boldsymbol{D}_j\}$: notice that this cluster here could be an anomaly having been labeled. The λ has a pre-determined value, being set as 0.98 by experimentation. Obviously, $\log(nc)$ is the maximum *Shannon* entropy of $\boldsymbol{P}$, which is achieved when all the $\boldsymbol{p}_i$ are equal. In consequence, an anomaly is reported if there is no specific distance $\boldsymbol{D}_i (i \in \{1, \ldots, nc\})$ that is apparently bigger than the other $\boldsymbol{D}_j (j \neq i, j \in \{1, \ldots, nc\})$.

It is noteworthy that the measure of the trajectory distance used in the process of anomaly detection is important. Considering in practice that the performances of the different distance measures are close to each other [33] [34], for the sake of efficiency we use the popular measures that can be computed fast. As for the popular measures, *DTW* and *LCSS* are the two good choices [31][32]. *DTW* has advantages over *LCSS*: because *DTW* can handle the nonlinear difference between trajectories and, it does not need the parameter that is usually determined by experimentation [43][44]. Actually, *DTW* has been proved better than *LCSS* in a recent work for trajectory analysis [29]. As a result, we select *DTW* to obtain the trajectory distance in the procedure of abnormal detection. In the calculation of *DTW*, the distance between trajectory sample points is computed using Bhattacharyya coefficient [41] for the corresponding representative *PDF*s to these points, inspired by the work of Calderara *et al.* [6].

VII. Experiments on simulation and real trajectory data

Based on simulation and real data with different speed, angle and noise attributes, extensive tests have been done to evaluate the performance of our proposed technique. Further details on trajectory data, along with the experimental results, are presented in the following sections.

A. Experiments on simulation trajectory data

Motivated by evaluating the performance of our proposed approach comprehensively, we design a trajectory generator to create the simulation trajectory data. The simulation data particularly consider the speed attribute of the trajectories, compared with those data mainly focusing on the angle attribute used in [6]. Additionally we add random noise to these simulation trajectories to vividly simulate the cases in reality. In all, our devised dataset consists of 20 classes (Figure 6 and Table I) which contain 5130 trajectories with the duration and sample points varying from 10 to 20 seconds and 50 to 100 points, respectively. The number of the trajectories in the "abnormal" cluster is kept below 5% of total number of the simulation trajectories.

Class	*C1*	*C2*	*C3*	*C4*	*C5*	*C6*	*C7*	*C8*	*C9*	*C10*
Speed level	3	3	3	3	0	5	0,7	3,4	9	3
Main angle	0°	180°	99°	45°	90°	270°	$126^\circ, 54^\circ$	$36^\circ, 288^\circ$	0°	90°

Class	*C11*	*C12*	*C13*	*C14*	*C15*	*C16*	*C17*	*C18*	*C19*	*C20*
Speed level	3,4,3	3,4,3	4,3,4	3	6	6	7,8	4,3,3	2,1	0,9
Main angle	$45^\circ, 0^\circ, 81^\circ$	$27^\circ, 0^\circ, 306^\circ$	$306^\circ, 0^\circ, 27^\circ$	135°	63°	60°	$81^\circ, 27^\circ$	$225^\circ, 90^\circ, 225^\circ$	$81^\circ, 27^\circ$	63°,63°

TABLE I

The speed levels and main angles of the simulation trajectories (classes *C1* $\sim$ *C20*). The speed has 10 levels $0 \sim 9$. For the class with more than one sub-class, the corresponding speed and angle values to these sub-classes are presented one by one.

We conduct 10 tests *Test_n* $(n = 1, 2, \ldots, 10)$, with the details listed in Table II. For each test in this Table, from left to right, the columns respectively denote the name and the brief description of the test, the expectation and standard deviation

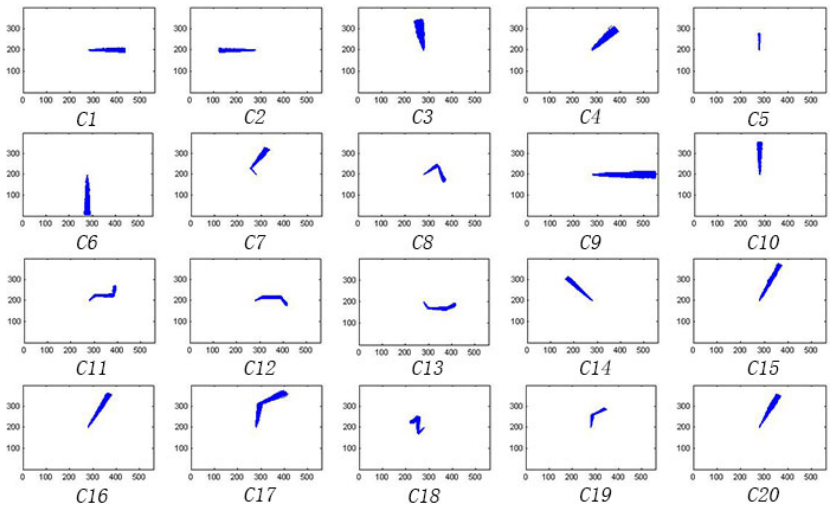


Fig. 6. The simulation trajectories are devised to have 20 classes, with the varied speed, angle and noise attributes. The denotation Cn indicates the class n, $n = 1, 2, \ldots, 20$. For each Cn, the trajectories are designed/shown starting from the origin $(320, 210)$. Some class involves more than one sub-class, and these sub-classes are linked in sequence. The information on speed levels and main angles is in Table I.

Type of test	μ	σ	Training classes		Testing classes	
			Classes	Total	Classes	Total
Test_1 (periodicity)	0.1	0.5	*C1*(150) *C2*(150)	300	*C1*(150) *C2*(150)	300
Test_2 (noise)	0.2	0.5	*C3*(200) *C4**(100)	300	*C4*(150) *C5**(10)	160
Test_3 (mono-modality)	0.1	0.5	*C1*(150) *C2*(150) *C5*(100) *C6**(20)	420	*C1*(150) *C2*(150) *C5**(50) *C6**(10)	360
Test_4 (multi-modality)	0.1	0.5	*C2*(150) *C7**(150) *C8**(10) *C20*(100)	410	*C2*(100) *C7**(100) *C8**(10) *C20*(100)	310
Test_5 (similar angle)	0.1	0.4	*C1*(150) *C9*(240) *C17*(200) *C19*(200)	790	*C1**(100) *C9**(60) *C17**(120) *C19**(50)	330
Test_6 (similar speed)	0.1	0.4	*C1*(150) *C2**(20) *C10*(240)	410	*C1**(100) *C2**(10) *C10**(60)	170
Test_7 (robustness)	0.1	0.5	*C11*(200) *C12*(240) *C13*(200)	640	*C11**(120) *C12**(120) *C13**(60) *C14**(20)	320
Test_8 (similar speed and angle)	0.1	0.3	*C15*(100) *C16**(5)	105	*C15**(50) *C16**(15)	65
Test_9 (cluster updating)	0.1	0.5	*C2*(150) *C8**(20) *C17*(200)	370	*C2*(150) *C8*(150) *C17**(120)	420
Test_10 (complex)	0.1	0.5	*C2*(150) *C6*(150) *C7**(150) *C12*(240) *C15*(100) *C18**(10)	800	*C2*(150) *C6*(100) *C7**(120) *C12**(120) *C15**(50) *C18**(10)	550

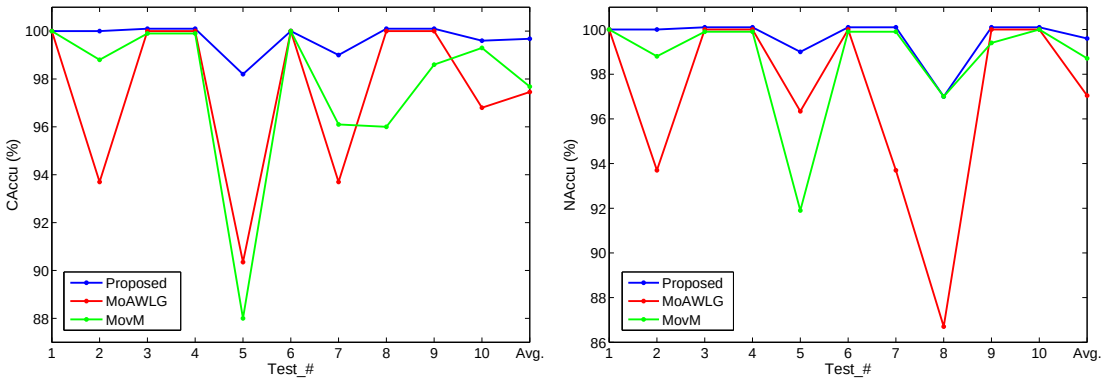
TABLE II
THE DESIGNED SIMULATION DATA. HERE "*" INDICATES A SUBSET OF THE ORIGINAL IS CHOSEN.

of the added noise for the test data, the classes and the total number of the trajectories for training and testing stages. The 10 tests are depicted as follows.

1) *Test_1* is to evaluate the modeling performance for the angle *periodicity* of the trajectory data. Basically the angles of the trajectories in *C1* and *C2* vary around $0°$ and $180°$ respectively, and in fact the medium-high level of random noise makes the trajectory angles fluctuate around these two certain values.
2) *Test_2* is to consider the robustness to high-level *noise*. The employed trajectories contained in *C3*, *C4* and *C5* have pretty high level of noise, which makes the test task rather harder.
3) *Test_3* and *Test_4* are respectively to study the applicabilities to *mono-modality* and *multi-modality* of the trajectories. In concrete, a mono-modality trajectory set is based on the data with single average speed and angle attributes. In contrast, multi-modality set contains multiple speed and angle properties.
4) *Test_5* and *Test_6* are to analyze respectively the abilities to discriminate the *similar angle* attributes and *similar speed* attributes of trajectory data. In *Test_5*, trajectories in *C1* and *C9* have apparent different speed but very similar angle attributes, and this also applies to *C17* and *C19*. As for *Test_6*, the trajectories are designed to behave similar speed but different angle properties.
5) *Test_7* is to account for the *robustness* to that the trajectories in *C11*, *C12* and *C13* have sample points with similar angles and speeds but in different organization order. In addition, the angle values of the trajectories in *C14** used as testing data are rather different from those of the training data.
6) *Test_8* is to consider the special case where both the *similar speed and angle* attributes of the trajectory data happen.
7) *Test_9* is to study the skill of the *cluster updating*: here an abnormal cluster *C8** becomes the normal *C8*, with the increasing number of the trajectories in this cluster (see here the different numbers of the trajectories, 20 and 150, respectively contained in *C8** and *C8*).
8) *Test_10* is to give a *complex* case where the trajectories are based on both single and multiple average speed and angle attributes.

We compare our proposed approach with two state-of-the-art methods, *MoAWLG* [8] and *MovM* [6] based techniques for trajectory shape analysis. The parameters of the two methods for comparison are experimentally selected to obtain their best possible results. Note that, especially for the sake of fair comparison, our approach just uses the k-medoids clustering for

the simulation data in the stage of trajectory training, considering that the *MoAWLG* [8] and *MovM* [6] based methods both use k-medoids for this stage. And this is also feasible, because we know in advance the optimal number of clusters for our designed trajectories. Here the trajectory distance used in k-medoids is obtained by using DTW, as discussed in Section VI. Two metrics, namely the Classification Accuracy (*CAccu*, the capacity to classify a new trajectory to a cluster correctly) and the Normal Accuracy (*NAccu*, the capacity to accurately label a trajectory as either normal or abnormal), are used to quantitatively evaluate the three methods. The higher accuracy corresponds to the better results. The accuracy results by the new approach, *MoAWLG* and *MovM* are shown in Figure 7, and obviously, our proposed method achieves the best performance: details are discussed in the following.



(1) *CAccu* (2) *NAccu*

Fig. 7. The accuracy statistics on the simulation trajectories. Our method achieves the best performance in most cases.

Importantly, the results of *Test_5* indicate the superiority of the proposed approach over the other two methods for the trajectory data with large different speeds but very similar angles. We believe this is because the *KDE* used by our approach can explicitly account for, besides the shape, the speed attribute of a trajectory, modeling the trajectory data very well. By contrast, in this case, *MoAWLG* cannot dependably represent the trajectories with varied speeds while with very similar angles, due to that it resorts to the difficult model fitting in the procedure of the trajectory modeling. And, *MovM* does not explicitly take into account the speed attribute of a trajectory. For the case of *Test_8* where the trajectory speeds and angles are both similar, the trajectory modeling is difficult. Our proposed method also outperforms its competitors. The results based on *Test_2* and *Test_7* show that the trajectory modeling in a statistical way is insensitive to data noise and to the order of the trajectories organized. Apparently, the proposed technique is better than the other two. Additionally, our scheme has much more robustness for the training and testing data. Notice that, in *Test_7*, the angle values of the testing trajectories in *C14** and those of the training data are largely different. *Test_1* and *Test_6* are used to analyze the modeling performances of different methods for the trajectory data emphasizing their angle attribute. Here all the three techniques work very well. From the results of *Test_3*, *Test_4* and *Test_10*, we can clearly see that the three methods perform satisfactorily on the trajectory data with both single and multiple average speed and angle characteristics. The results of *Test_9* show that all the three techniques have the ability to do the cluster updating. All in all, we can conclude that our proposed method is the best among the three methods for comparing.

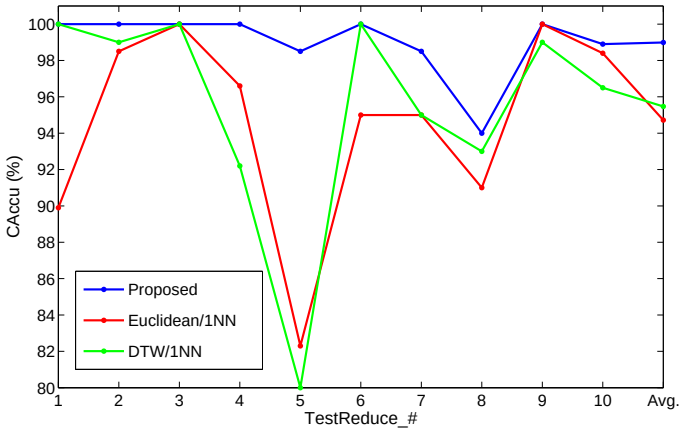


Fig. 8. The comparison with the classical Euclidean/1NN and *DTW*/1NN. Our method behaves the best.

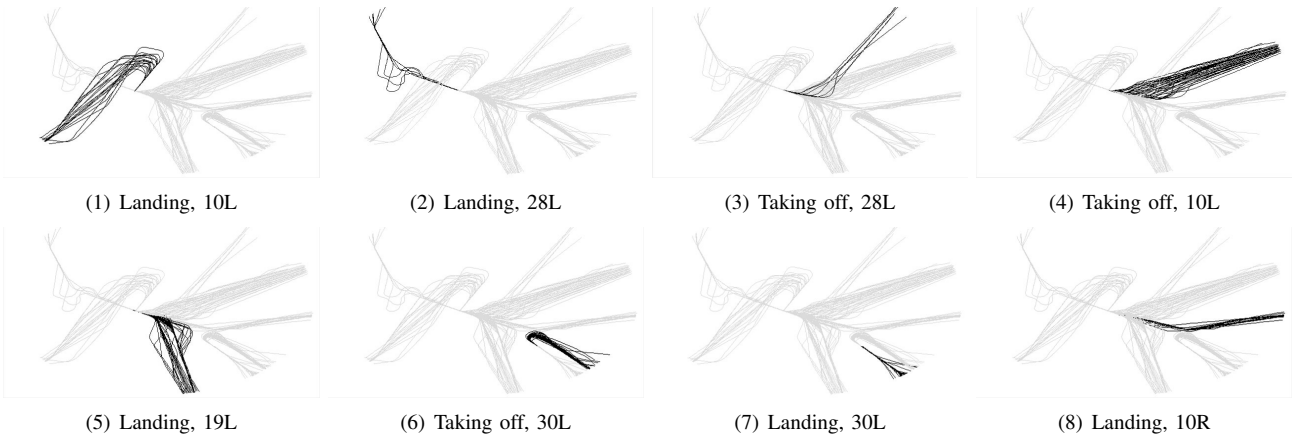
In addition, we compare our method with the classic data mining approaches based on two widely used distance measures,

Euclidian [28][18] and *DTW* distances [31], between two sample points without modeling the trajectories in a compact and representative form in advance. Here, the classical supervised *one-Nearest-Neighbor* framework (1NN) [33], which can run very fast and robustly, is taken for our purpose. Concretely, the two approaches, Euclidean/1NN and *DTW*/1NN, are employed to do the comparison. Considering the fair comparison, now the test trajectories come from those in *Test_n* ($n = 1, 2, \ldots, 10$), without the ones that are employed for testing but not for training. For convenience, we use *TestReduce_n* to stand for the *n*-th test of comparison with classical approaches. The comparison results can be seen in Figure 8. Overall, our proposed method achieves the best. The average accuracy by us is 98.99%, in contrast with 94.72% by Euclidean/1NN and with 95.47% by *DTW*/1NN. *TestReduce_5* shows that our proposed method is significantly better than the other two when the trajectory data have different speeds but quite similar angles, which is due to that the *KDE* used by us can simultaneously consider speed and angle attributes, but *MoAWLG* and *MovM* are not able to achieve that level. As the complexity of the trajectory data becomes bigger, for example, in *TestReduce_10*, the proposed solution reaches the accuracy 98.9%, higher than 98.4% by Euclidean/1NN and than 96.5% by *DTW*/1NN, adequately demonstrating the advantage of the statistical modeling utilized by us.

B. Experiments on real trajectory data

In order to compare the performances of the three approaches, we make use of a real aircraft trajectory dataset from [45] contributed by Gariel *et al.* [46]. The dataset consists of 320 aircraft trajectories. The trajectories have different numbers of sample points varying from 70 to 210 and include the taking off and landing operations on runways 10L, 10R, 19L, 28L and 30L.

As mentioned in Section V, obtaining a feature space Y of the trajectory data set X is very necessary and important for *MI* based clustering. In this experimentation, the feature space Y is established according to the physical meaning of the trajectory data. Apparently, an aircraft trajectory, which includes either a taking off or a landing operation, is usually based on a certain runway. Actually, a trajectory happening to a certain runway, either for taking off or for landing operations, corresponds either to the speeding up or to the slowing down of an aircraft. As a result, a trajectory x in the trajectory dataset X ($x \in X$) is grouped into an element y of the feature space Y ($y \in Y$) based both on to which runway x happens and on whether x is speeding up or slowing down. The 8 elements of this feature space are highlighted in Figure 9(1) - Figure 9(8) respectively.



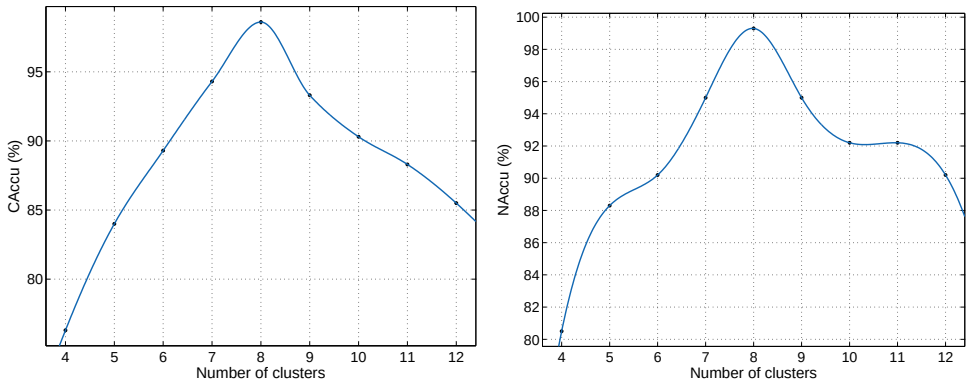
(1) Landing, 10L (2) Landing, 28L (3) Taking off, 28L (4) Taking off, 10L

(5) Landing, 19L (6) Taking off, 30L (7) Landing, 30L (8) Landing, 10R

Fig. 9. A feature space of the real aircraft trajectory data.

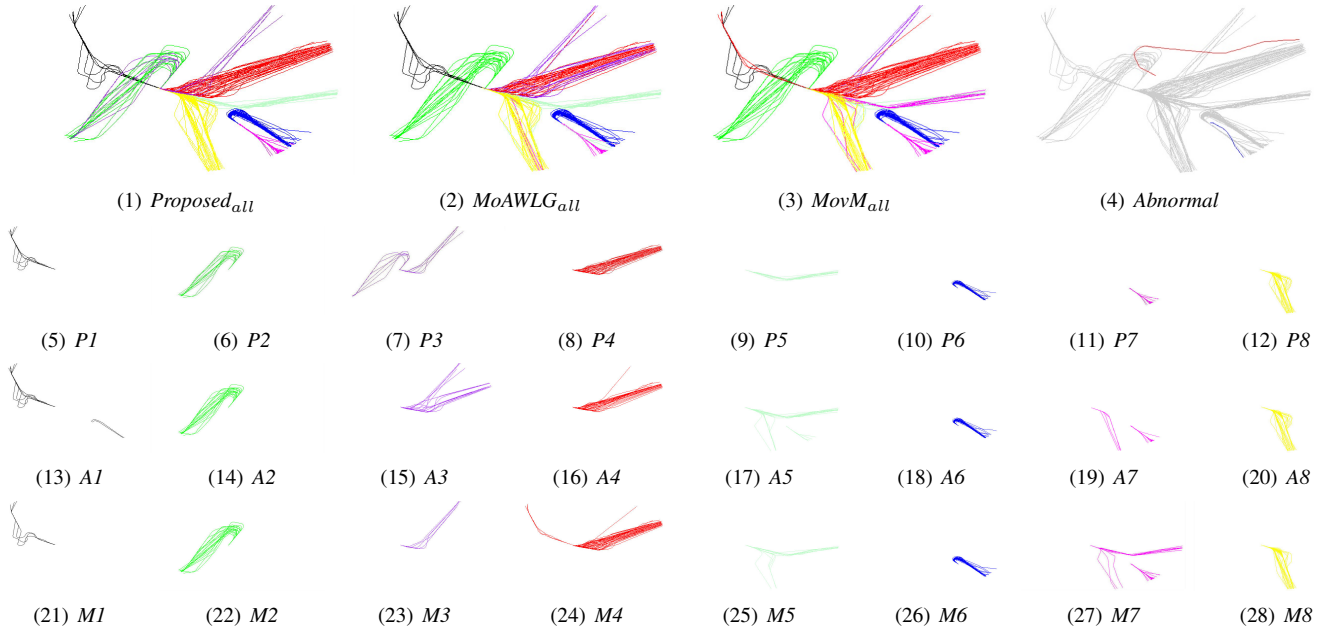
Figure 5 presents the loss of mutual information by the cluster merging in the procedure of the *IB* based trajectory clustering for this real trajectory dataset. Our approach obtains 8 as the optimum number of clusters through the use of the *IB* technique. Specifically, based on the modeling representations resulted from using *KDE*, we use the k-medoids to do the clustering to experimentally find a good number of clusters, given a range of candidate numbers of clusters. With each candidate, we run the k-medoids times to obtain the best possible clustering results, and then we use our proposed technique for anomaly detection discussed in Section VI to obtain the classification results. Figure 10 gives the plots for the *CAccu* and *NAccu*, versus different candidate numbers of clusters. It is clearly verified that 8 is the optimal number of clusters for this real trajectory dataset. It is also worthy to point out that, by extensive experimentation, *MoAWLG* and *MovM* obtain the best cluster number as 8. This further exemplifies the effectiveness of our method to achieve an optimal number of clusters.

We compare the new proposed method with the state-of-the-art *MoAWLG* [8] and *MovM* [6], and here all the parameters employed in the two competitive approaches are finely tuned to obtain their best possible outputs. The visual results by the three methods for comparison are shown in Figure 11. The 8 clusters *P1* - *P8* achieved by our proposed approach are presented in Figure 11(5) - Figure 11(12) respectively and, the clusters *A1* - *A8* (Figure 11(13) - Figure 11(20)) and *M1* - *M8* (Figure 11(21) - Figure 11(28)) are respectively produced by *MoAWLG* and *MovM*.



(1) *CAccu* (2) *NAccu*

Fig. 10. The clustering and classification accuracies due to different numbers of clusters.



(1) $Proposed_{all}$ (2) $MoAWLG_{all}$ (3) $MovM_{all}$ (4) *Abnormal*

(5) *P1* (6) *P2* (7) *P3* (8) *P4* (9) *P5* (10) *P6* (11) *P7* (12) *P8*

(13) *A1* (14) *A2* (15) *A3* (16) *A4* (17) *A5* (18) *A6* (19) *A7* (20) *A8*

(21) *M1* (22) *M2* (23) *M3* (24) *M4* (25) *M5* (26) *M6* (27) *M7* (28) *M8*

Fig. 11. $Proposed_{all}$, $MoAWLG_{all}$ and $MovM_{all}$ respectively represent the clustering results by three approaches being compared, and *P1* - *P8*, *A1* - *A8* and *M1* - *M8* come form $Proposed_{all}$, $MoAWLG_{all}$ and $MovM_{all}$, respectively. In *Abnormal*, the anomalous trajectory, upper and red, can be recognized by the three methods for comparison; but, the abnormal trajectory, bottom and blue, can be identified by the proposed technique only.

The clustering conveyed by *P1* - *P8* gives a kind of "optimal" division of the trajectory data, based on their speed and angle attributes, and meanwhile on the regularization by the feature space of the data through the *IB*. Actually the clustered result achieved by us is in accordance with the physical rationality of the real data in the practical civil aviation scenario, considering that, given a trajectory happening to a certain runway with a certain operation (either taking off or landing), both speed and angle attributes of it are the important factors for the vital issues, such as the runway safety [47]. The median speeds for *P1* - *P8* are 106, 151, 179, 198, 152, 171, 107 and 161 respectively. Notice, the median speed for a trajectory is a median of the speeds for all the sample points of this trajectory. Here the speed for a sample point is a normalized value based on the largest speed for all the sample points of all the trajectories. The median speed for a trajectory cluster is taken as an average of the median speeds for all the trajectories of this cluster. The main directions for *P1* - *P8* are NW-SE, SW-NE, SW-NE, SW-NE, E-W, NW-SE, SE-NW and SE-NW respectively. In concrete, the median angles for *P1* - *P8* are 281°, 67°, 66°, 30°, 179°, 305°, 126° and 109° respectively, here the median is completely analogous to that for the speed value mentioned just now. Notice that there exist two relatively separate parts of *P3* respectively belonging to the two different elements of the feature space of the trajectory data (Figure 9(2) and Figure 9(3)). Actually, the respective trajectories in these two parts have very similar speed and angle attributes, thus our method groups them into a single cluster. This is rational, for example, from the perspective of the safety control on the speed of aircrafts, for the practical use of these trajectories.

Remarkably it is not so easy to correctly differentiate the trajectories respectively belonging to the two clusters *P2* and *P3*, because the trajectories respectively in these two clusters have very similar angle attribute and in speed they only have some

difference that is not big. In this case, *MoAWLG* unfortunately combines the trajectories in the left part of *P3* with those of *P2* to give an unsatisfactory *A2*, and very similarly *MovM* provides an inferior *M2*. Actually, this is because that the *MoAWLG* is based on a bi-variable Gaussian like function and, it is difficult to do a good model fitting for the case that the value of a certain variable (here, in this example, the value of the trajectory angle) is approximately constant. And as for the *MovM*, this model does not explicitly take into account the speed attribute. Moreover, some problem happens to *A3* and *A4* by *MoAWLG*, also to the right part of *M4* by *MovM* (and here so that *M3* is not correct either). That is, *A3*, *A4* and the right part of *M4* all unsatisfyingly mix the trajectories respectively in the two separate clusters *P3* and *P4*, of which the respective trajectories have small speed differences but large angle distinctions. The reason for the current issue of *A3* and *A4* is due to the use of the bi-variable Gaussian like function in *MoAWLG*, as discussed above. The outlier for the right part of *M4* results from the trajectory clustering method used in *MovM*. Note that the cluster *P1* achieved by our method is composed of the "straight" trajectories and the curved ones, and this is because of considering the integration of both speed and angle attributes of these trajectories. For the same reason, *MoAWLG* also gives a similar result, *A1*. However, this *A1* unsatisfactorily includes two trajectories belonging to the cluster *P6* and, based on the analysis of the trajectory clustering in *MoAWLG*, we have found that this shortage is due to the measure of trajectory distance, *LCSS*, used by *MoAWLG*. But notably, the clustering performance by our method in this case is better, due to deploying the *IB* to avoid the explicit application of the difficult trajectory distance. The cluster *M1* by *MovM* solely involves the curved trajectories in *P1*, because *MovM* considers the angle attribute only. Besides, *MovM* introduces some "straight" trajectories in *P1* to form the unsatisfactory left part of *M4*, without reasonably discriminating the enough speed and angle differences between the trajectories respectively in the left and right parts of *M4*: this is also as a result of the metric of trajectory distance, *LCSS*, deployed in the clustering procedure of *MovM*.

The three clusters *P5*, *P7* and *P8* are achieved very well by our method. However, *MoAWLG* and *MovM* cannot obtain this. As a matter of fact, some "mess" is caused by *MoAWLG* and *MovM* for their clusters involving the trajectories that come from *P5*, *P7* and *P8*. In concrete, when the three clusters *A5*, *A7* and *A8* provided by *MoAWLG* are put together, we can easily find that the corresponding "mess" presents in the trajectories with the cyan, rose and yellow colors in Figure 11(2). The similar situation applies to the outputs of *MovM*, *M5*, *M7*, *M8*, and the cyan, rose and yellow trajectories in Figure 11(3). Here the reason is somewhat complicated. The clusters *P7* and *P8* contain the respective trajectories with (very) similar angle while clear difference in speed. In the case of *MoAWLG*, the problem by the model fitting leads to *A7* and, to the partial *A5* including the trajectories from both *P7* and *P8*. Also, the difficulty of model fitting makes *MovM* confuse the trajectories in *M7* that respectively come from *P7* and *P8*. The other "mess" problems are due to the clustering used by *MoAWLG* and *MovM*, concretely resulting from the measure of the distance, *LCSS*, between the trajectories in *P5* and those either in *P7* or in *P8*.

Accuracy	*Proposed*	*MoAWLG*	*MovM*
CAccu	95.60%	90.30%	87.20%
NAccu	96.50%	92.20%	90.43%

TABLE III
THE ACCURACY STATISTICS ON REAL TEST TRAJECTORY SET.

Figure 11(4) presents two trajectories, an upper (red) and a bottom (blue) ones. The median speed and angle for the upper trajectory are 217 and 179° respectively, and those for the bottom one are 151 and 270° respectively. In essence the two trajectories here are both anomalous in the sense that, either their speeds or angles, or even both, are different enough from the corresponding values of the trajectories in all the clusters learned. Obviously and undoubtedly, these two themselves do have the clear difference in their speed and angle attributes. And it is important to note, *MoAWLG* and *MovM* fail to detect the bottom trajectory as abnormal, because here some single "hard" threshold employed is not capable of identifying this anomaly really as abnormal. On the contrary, both the upper and bottom trajectories can be recognized as anomalous by our *Shannon* entropy based anomaly detection, due to an adaptive differentiation between normal and abnormal trajectories.

Additionally, we quantitatively evaluate the three approaches for the real test dataset, and Table III lists the *CAccu* and *NAccu* statistics. Apparently Table III indicates that our method performs the best, which is compatible with the visual results demonstrated in Figure 11.

Last but also important, the processing time for the anomaly detection only is briefly reported here, considering that the trajectory learning is done in an off-line way. The runtime results are obtained on a Windows PC with Intel Core 2 Duo CPU P8700 (2.53 GHz) and 2 GB RAM. For a trajectory with an average of 75 sample points, the proposed technique costs 15 ms, while *MoAWLG* and *MovM* use 409 ms and 432 ms respectively. Notice that, the model fitting in the procedure of anomaly detection, which is necessary for *MoAWLG* and *MovM*, but not for our proposed approach, results in a big difference of the runtime here.

VIII. Conclusion and Future Work

In this paper, for the purpose of anomaly detection, we have established a new effective technique in which the speed attribute is explicitly utilized to improve the shape analysis of trajectory data. A motion trajectory is statistically represented

based on the nonparametric *KDE*, without the need of the difficult model fitting for some assumed parametric function that is not feasible in practice. In this way, the trajectory data can be depicted correctly and very well, which has also been demonstrated on the statistical models obtained, both by the 3D surface plot rendering and by the visualization metaphor. As for the trajectory learning, we have exploited the *MI* and *IB* to do the clustering. And as a result, both the trajectory distance and the optimal number of trajectory clusters, which are very important but difficult to handle, have been given solutions sound. To our knowledge this is the first deployment of the powerful *IB* technique for trajectory processing. Also importantly, the *Shannon* entropy has been adopted, in a finely discriminative way, for adaptively identifying whether a test trajectory behaves as an anomaly or not. A lot of experiments on both simulation and real trajectory data have revealed clearly that the proposed technique significantly outperforms the state-of-the-art and classic methods, in terms of quantitative and qualitative evaluations.

Several improvements for trajectory modeling and clustering will be performed in our future work, and some of them have been indicated above. Some typical transformations, such as *DFT* and *DWT*, will be used for the speeding up of the trajectory computing. Some more trajectory attributes, such as spatial position, acceleration and etc., will be considered for the comprehensive modeling of trajectory data. In order to obtain the multivariate statistical modeling, *Maximum Entropy Principle* [48] would be exploited and, hierarchical spatio-temporal like processing [28] [49] could be employed for the benefit of computing efficiency. As for the *IB*, some means such as stochastic optimization would serve for the goal of its global optimization [50] [42]. For the incremental updating of the clusters having been learned, some mathematical representation such as the *Dirichlet* process mixture model [51] could be exploited to achieve this. In addition, the general and mighty *visual analytics* technique [52] would be developed to propose attractive visualizations and effective user interactions to further enhance all the steps of the trajectory analysis.

ACKNOWLEDGEMENT

This work has been funded by Natural Science Foundation of China (61471261, 61179067, U1333110), Spanish Government grant (TIN2013-47276-C6-1-R), and Catalan Government grant (2014-SGR-1232).

REFERENCES

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys (CSUR)*, vol. 41, no. 3, p. 15, July 2009.
[2] H. M. Dee and S. A. Velastin, "How close are we to solving the problem of automated visual surveillance?" *Machine Vision and Applications*, vol. 19, no. 5-6, pp. 329–343, October 2008.
[3] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 34, no. 3, pp. 334–352, August 2004.
[4] B. T. Morris and M. M. Trivedi, "Understanding vehicular traffic behavior from video: a survey of unsupervised approaches," *Journal of Electronic Imaging*, vol. 22, no. 4, pp. 041 113–041 113, September 2013.
[5] ——, "A survey of vision-based trajectory learning and analysis for surveillance," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 8, pp. 1114–1127, June 2008.
[6] S. Calderara, A. Prati, and R. Cucchiara, "Mixtures of von mises distributions for people trajectory shape analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 4, pp. 457–471, March 2011.
[7] A. Prati, S. Calderara, and R. Cucchiara, "Using circular statistics for trajectory shape analysis," in *IEEE Conference on Computer Vision and Pattern Recognition, 2008 (CVPR 2008)*. IEEE, June 2008, pp. 1–8.
[8] S. Calderara, A. Prati, and R. Cucchiara, "Learning people trajectories using semi-directional statistics," in *Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance, 2009 (AVSS'09)*. IEEE, September 2009, pp. 213–218.
[9] B. W. Silverman, *Density estimation for statistics and data analysis*. London: Chapman and Hall, 1986.
[10] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method," in *Proceedings of the 37th annual Allerton Conference on Communication, Control, and Computing*. Citeseer, September 1999, pp. 368–377.
[11] T. M. Cover and J. A. Thomas, *Elements of Information Theory, second ed.* Wiley-Interscience, San Francisco, July 2006.
[12] W. Hu, D. Xie, T. Tan, and S. Maybank, "Learning activity patterns using fuzzy self-organizing neural network," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 3, pp. 1618–1626, June 2004.
[13] E. Smart and D. Brown, "A two-phase method of detecting abnormalities in aircraft flight data and ranking their impact on individual flights," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 3, pp. 1253–1265, March 2012.
[14] M. Kakkasageri and S. Manvi, "Information management in vehicular ad hoc networks: A review," *Journal of Network and Computer Applications*, vol. 39, pp. 334–350, March 2014.
[15] J. Alon, S. Sclaroff, G. Kollios, and V. Pavlovic, "Discovering clusters in motion time-series data," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE, June 2003, pp. I–375–I–381.
[16] F. Porikli, "Trajectory distance metric using hidden markov model based representation," in *IEEE European Conference on Computer Vision, PETS Workshop*, vol. 3. Citeseer, May 2004.
[17] F. I. Bashir, A. A. Khokhar, and D. Schonfeld, "Object trajectory-based activity classification and recognition using hidden markov models," *IEEE Transactions on Image Processing*, vol. 16, no. 7, pp. 1912–1919, July 2007.
[18] C. Piciarelli, C. Micheloni, and G. L. Foresti, "Trajectory-based anomalous event detection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1544–1554, November 2008.
[19] J. Owens and A. Hunter, "Application of the self-organising map to trajectory classification," in *Proceedings of the Third IEEE International Workshop on Visual Surveillance*. IEEE, 2000, pp. 77–83.
[20] S. Khalida and S. Razzaq, "Frameworks for multivariate m-mediods based modeling and classification in euclidean and general feature spaces," *Pattern Recognition*, vol. 45, no. 3, pp. 1092–1103, March 2012.
[21] R. Laxhammar and G. Falkman, "Online learning and sequential anomaly detection in trajectories," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 6, pp. 1158–1173, September 2014.
[22] A. Basharat, A. Gritai, and M. Shah, "Learning object motion patterns for anomaly detection and improved object detection," in *IEEE Conference on Computer Vision and Pattern Recognition, 2008 (CVPR 2008)*. Anchorage, AK: IEEE, June 2008, pp. 1–8.
[23] I. N. Junejo, O. Javed, and M. Shah, "Multi feature path modeling for video surveillance," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004 (ICPR 2004)*, vol. 2. Cambridge, England, UK: IEEE, August 2004, pp. 716–719.

[24] N. Anjum and A. Cavallaro, "Multifeature object trajectory clustering for video analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1555–1564, September 2008.

[25] G. Antonini and J. P. Thiran, "Counting pedestrians in video sequences using trajectory clustering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 8, pp. 1008–1020, August 2006.

[26] W. Chen and S.-F. Chang, "Motion trajectory matching of video objects," in *Proceedings of the SPIE 3972, Storage and Retrieval for Media Databases 2000*. San Jose, CA, USA: International Society for Optics and Photonics, December 1999, pp. 544–553.

[27] R. R. Sillito and R. B. Fisher, "Semi-supervised learning for anomalous trajectory detection," in *Proceedings of the British Machine Vision Conference, 2008 (BMVC 2008)*, Leeds, UK, September 2008, pp. 1–10.

[28] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank, "A system for learning statistical motion patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1450–1464, September 2006.

[29] S. Atev, G. Miller, and N. P. Papanikolopoulos, "Clustering of vehicle trajectories," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 647–657, Month 2010.

[30] W. Ge, R. T. Collins, and R. B. Ruback, "Vision-based analysis of small groups in pedestrian crowds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 1003–1016, May 2012.

[31] C. A. Ratanamahatana and E. Keogh, "Three myths about dynamic time warping data mining," in *Proceedings of the 2005 SIAM International Conference on Data Mining*. SIAM, 2005, pp. 506–510.

[32] M. Vlachos, D. Gunopulos, and G. Das, "Rotation invariant distance measures for trajectories," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining 2004*. ACM, 2004, pp. 707–712.

[33] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: experimental comparison of representations and distance measures," *Proceedings of the VLDB Endowment*, vol. 1, no. 2, pp. 1542–1552, August 2008.

[34] B. Morris and M. Trivedi, "Learning trajectory patterns by clustering: Experimental studies and comparative evaluation," in *IEEE Conference on Computer Vision and Pattern Recognition, 2009 (CVPR 2009)*. IEEE, June 2009, pp. 312–319.

[35] K. P. Burnham and D. R. Anderson, "Multimodel inference understanding aic and bic in model selection," *Sociological methods & research*, vol. 33, no. 2, pp. 261–304, November 2004.

[36] M. Feixas, A. Bardera, J. Rigau, Q. Xu, and M. Sbert, *Information Theory Tools for Image Processing*. San Rafael, CA, USA: Morgan & Claypool, 2014.

[37] Q. Xu, Y. Liu, X. Li, Z. Yang, J. Wang, M. Sbert, and R. Scopigno, "Browsing and exploration of video sequences: A new scheme for key frame extraction and 3d visualization using entropy based jensen divergence," *Information Sciences*, vol. 278, pp. 736–756, September 2014.

[38] D. W. Scott, *Multivariate density estimation: theory, practice, and visualization*. New York: Wiley, 1992.

[39] M. P. Wand and M. C. Jones, *Kernel smoothing, Monographs on Statistics and Applied Probability*. CRC Press, 1995, vol. 60.

[40] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Third Edition*. Orlando, FL, USA: Academic Press, Inc., 2006.

[41] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 2. Hilton Head Island, SC: IEEE, June 2000, pp. 142–149.

[42] N. Slonim, "The information bottleneck: Theory and applications," *Ph.D Thesis, Tel-Aviv University*, 2002.

[43] L. Chen, M. T. Özsu, and V. Oria, "Robust and fast similarity search for moving object trajectories," in *Proceedings of ACM SIGMOD*. Baltimore, Maryland, USA: ACM, June 2005, pp. 491–502.

[44] N. Pelekis, I. Kopanakis, G. Marketos, I. Ntoutsi, G. Andrienko, and Y. Theodoridis, "Similarity search in trajectory databases," in *14th International Symposium on Temporal Representation and Reasoning*. Alicante: IEEE, June 2007, pp. 129–140.

[45] NASA, "https://c3.nasa.gov/dashlink/resources/132/," 2011.

[46] M. Gariel, A. N. Srivastava, and E. Feron, "Trajectory clustering and an application to airspace monitoring," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1511–1524, December 2011.

[47] Wiki, "http://en.wikipedia.org/wiki/runway_safety_area," 2014.

[48] J. E. Shore and R. W. Johnson, "Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy," *IEEE Transactions on Information Theory*, vol. 26, no. 1, pp. 26–37, January 1980.

[49] Q. Xu, H. Jiang, R. Scopigno, and M. Sbert, "A novel approach for enhancing very dark image sequences," *Signal Processing*, vol. 103, pp. 309–330, October 2014.

[50] S. Andradóttir, "Stochastic optimization with applications to discrete event systems," *Ph.D Thesis, Stanford University*, 1990.

[51] W. Hu, X. Li, G. Tian, S. Maybank, and Z. Zhang, "An incremental dpmm-based method for trajectory clustering, modeling, and retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 5, pp. 1051–1065, May 2013.

[52] R. May, P. Hanrahan, D. A. Keim, B. Shneiderman, and S. Card, "The state of visual analytics: views on what visual analytics is and where it is going," in *IEEE Symposium on Visual Analytics Science and Technology (VAST)*. Salt Lake City, UT: IEEE, October 2010, pp. 257–259.