An Incremental DPMM-Based Method for Trajectory Clustering, Modeling, and Retrieval

Weiming Hu, Xi Li, Guodong Tian, Stephen Maybank, and Zhongfei Zhang

Abstract—Trajectory analysis is the basis for many applications, such as indexing of motion events in videos, activity recognition, and surveillance. In this paper, the Dirichlet process mixture model (DPMM) is applied to trajectory clustering, modeling, and retrieval. We propose an incremental version of a DPMM-based clustering algorithm and apply it to cluster trajectories. An appropriate number of trajectory clusters is determined automatically. When trajectories belonging to new clusters arrive, the new clusters can be identified online and added to the model without any retraining using the previous data. A time-sensitive Dirichlet process mixture model (tDPMM) is applied to each trajectory cluster for learning the trajectory pattern which represents the time-series characteristics of the trajectories in the cluster. Then, a parameterized index is constructed for each cluster. A novel likelihood estimation algorithm for the tDPMM is proposed, and a trajectory-based video retrieval model is developed. The tDPMM-based probabilistic matching method and the DPMM-based model growing method are combined to make the retrieval model scalable and adaptable. Experimental comparisons with state-of-the-art algorithms demonstrate the effectiveness of our algorithm.

Index Terms—Trajectory clustering and modeling, incremental clustering, Dirichlet process mixture model, time-sensitive Dirichlet process mixture model, video retrieval

1 Introduction

Motion is the essential characteristic distinguishing dynamic videos from still images. Motion information in videos represents the visual content with temporal variation, and plays a very important role in the depiction of the semantic contents in videos. Moving objects in videos can quickly attract the attention of watchers. A motion trajectory is obtained by tracking an object from one frame to the next and then linking its positions in consecutive frames. The resulting trajectories contain rich spatiotemporal semantic information. Trajectory analysis [21], [41] is the basis for many applications, such as indexing of motion events in videos, detection of paths in scenes, understanding of moving object behaviors, detection of anomalous motions, persistent tracking, and persistent labeling of objects [22]. In this paper, we focus on trajectory clustering, modeling, and retrieval.

1.1 Related Work

Key tasks in trajectory analysis include trajectory representation, similarity estimation, clustering, modeling, etc.

1.1.1 Trajectory Representation

The task of trajectory representation is to parameterize each trajectory, and then index the trajectory using the parameters. Trajectory representation methods include polynomial-based curve fitting, the Haar wavelet transform, minimum error-based polygonal approximation, B-spline curves, and discrete Fourier transform (DFT) coefficients. Little and Gu [1] used a polynomial-based curve fitting technique to represent spatiotemporal motion information about an object trajectory. Sahouria and Zakhor [6] applied the Haar wavelet transform to analyze an object trajectory's spatiotemporal information at multiscales. Jung et al. [27] used polynomial curve fitting to extract and encode features of 2D trajectory data and then further constructed a motion model, which was used as an indexing key for accessing individual objects. Pavlidis [28] used minimum error-based polygonal approximation and Cohen et al. [29] used B-spline curves to describe 2D trajectories. Ansari and Delp [30] represented a trajectory using dominant points with high curvatures. Gu [31] proposed a maximum curvature approximation method for reducing the dimensions of the trajectory feature space. Naftel and Khalid [8] learned motion trajectories through self-organizing maps in the DFT coefficient feature space. Yajima et al. [32] encoded trajectories according to the spatiotemporal relations between objects and query the movements of multiple moving objects by expressing each object's trace on a timeline. Palpanas et al. [43] proposed a method for an incrementally updated approximation of streaming time series. They showed that the method is very fast and maintains a high-quality approximation. Compared with point-based trajectory presentation, the parameterized trajectory representations substantially compress the trajectories, and they are even more effective for trajectory similarity estimation and clustering. It was also shown [36] that trajectory representation based on the DFT coefficients

- *W. Hu, X. Li, and G. Tian are with the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, No. 95, Zhongguancun East Road, PO Box 2728, Beijing 100190, P.R. China. E-mail: wmhu@nlpr.ia.ac.cn, lixichinanlpr@gmail.com,guodong_tian@126.com.*
- *S. Maybank is with the Department of Computer Science and Information Systems, Birkbeck College, Malet Street, London WC1E 7HX, United Kingdom. E-mail: sjmaybank@dcs.bbk.ac.uk.*
- *Z. Zhang is with the Department of Computer Science, Watson School of Engineering and Applied Sciences, Binghamton University, Binghamton, NY 13902-6000. E-mail: zhongfei@cs.binghamton.edu.*

Manuscript received 15 Feb. 2011; revised 25 May 2012; accepted 22 Aug. 2012.
Recommended for acceptance by V. Pavlovic.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2011-02-0103.
Digital Object Identifier no. 10.1109/TPAMI.2012.188.

performs more accurately than polynomial-based trajectory representation.

1.1.2 Trajectory Similarity Measure

In order to match or cluster trajectories, trajectory similarity is estimated using trajectory distance measures, such as the euclidean distance, the dynamic time warping (DTW) distance, the principal component analysis (PCA) distance, the edit distance with real penalty (ERP), and the longest common subsequence (LCSS) distance. The euclidean distance and the DTW distance [37], [38] are two basic measures for estimating trajectory similarities [41]. Dagtas and Al-Khatib [26] carried out video retrieval by measuring the minimum euclidean distances between the query trajectory and all the subtrajectories with the same number of points as the query in each candidate trajectory. A PCA-based distance was proposed by Bashir et al. [18] to measure trajectory similarities. Vlachos et al. [7] used the LCSS algorithm to measure the similarity between two object trajectories by analyzing objects' coordinates directly. A two-level PCA operation with coarse-to-fine retrieval was used to identify trajectories close to a query trajectory. Chen and Ng [42] proposed the ERP to measure similarities between time-series data. Hsieh et al. [9] divided trajectories into several small segments and encoded each segment using a semantic symbol. A distance measure which combines the edit distance and the visual distance was exploited to determine similarities between trajectories. Ma et al. [3] proposed a compact representation of multi-object motion trajectories based on 3D tensor subspaces. Multi-trajectory retrieval was carried out using the euclidean distances between the compact representations of multi-trajectories. Su et al. [35] constructed trajectories from motion vectors embedded in MPEG bit streams. The coordinates of each point in a trajectory were replaced with cumulative movement related parameters, which were then differentiated and used to compute a euclidean distance. Wang et al. [45] provided an experimental comparison of representations and distance measures for time-series data. Different trajectory distance measures have their own merits and limitations. For example, the merits of the traditional point-based euclidean distance are that it is simple, nonparametric, very fast, and easy to code. Its limitation is that it is not robust to noise or to distortion along the time axis [39], [40]. The merit of the DTW distance is the ability to tolerate temporal misalignments so as to allow for time warps, such as stretching or shrinking a portion of a sequence along the time axis, and for differences in length between time series. Its limitations are that it requires continuity along the warping path, and it is unable to find trajectories that have similar shapes but with dissimilar gaps in between [40]. The PCA distance obtains more accurate results and requires lower computational cost due to its compact representation. However, it is also sensitive to noise, like the euclidean distance. The ERP can handle the time-series data with local time shifts. However, because distances are measured using the differences of real values, ERP is also sensitive to noise [40]. The LCSS distance is more robust to noise and outliers than the DTW distance because not all points need to be matched. However, it allows gaps of various sizes to exist between similar shapes in the trajectories. It is difficult to set the two parameters for the LCSS distance estimation. The DTW and LCSS distances take more time than the euclidean distance and the PCA distance.

1.1.3 Trajectory Clustering and Modeling

More recent work on trajectory analysis focuses on learning trajectory patterns via trajectory clustering and modeling. Trajectory clustering assigns similar trajectories to the same cluster according to the measured similarities between trajectories. The task of trajectory modeling is to construct a model (especially a parameterized model) to represent and index each cluster of trajectories. Jung et al. [20] proposed a method for event detection based on trajectory clustering. In the training period, captured trajectories were grouped into coherent clusters for which 4D motion histograms were built. In the test period, each new trajectory was compared with the 4D histograms of all the clusters. Saleemi et al. [22] modeled motion patterns of objects in the scene using a multivariate nonparametric probability density function defined on the object locations and the transition times between them. Learning was accomplished by observing the trajectories of objects. Morris and Trivedi [23], [44] learned the normal motions encountered in a scene in an unsupervised way. The spatiotemporal motion characteristics of these paths were encoded by a hidden Markov model. Once these path definitions were established, abnormal trajectories were detected and future intent was predicted. Piotto et al. [24] proposed an algorithm for the syntactic description and matching of object trajectories in videos. Trajectories were segmented and described syntactically. Similarities between trajectories were determined based on inexact or approximate matching. Veeraraghavan and Papanikolopoulos [25] proposed an algorithm for learning sequenced spatiotemporal activities represented by the trajectories of road traffic at intersections. A semi-supervised learning algorithm was used to learn activities modeled by stochastic context-free grammars. Johnson and Hogg [2] used two competing neural networks connected by a leaky neuron layer to model the probability distribution of flow vectors and the trajectories. Alon et al. [5] employed a finite mixture of HMMs to learn motion data using the expectation-maximization (EM) algorithm. Atev et al. [4] presented a method for clustering vehicle trajectories using a Hausdorff distance-based trajectory similarity measure. Although current trajectory clustering and modeling methods solve a variety of specific problems in trajectory analysis, they still have the following limitations:

- An effective criterion is lacking for estimating an appropriate number of trajectory clusters. A manual choice of the number of clusters is subjective. Inappropriate setting of the number of trajectory clusters may result in inaccurate trajectory clustering, especially when the number of trajectories is very large.
- They lack the ability to cluster trajectories incrementally. When new trajectories are obtained, the model has to be retrained using the previous and the new trajectories. This results in a high computational complexity.
- They often index clusters of trajectories using high-dimensional trajectory cluster centroid vectors. It is

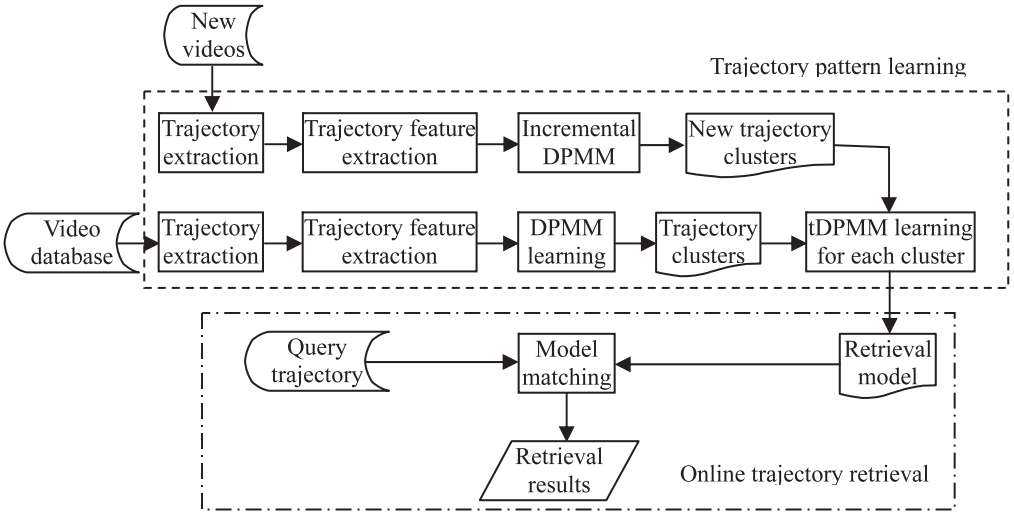


Fig. 1. The architecture of our video retrieval framework.

meaningful, based on the result of trajectory clustering, to construct low-dimensional parameterized indices of trajectory clusters because this saves space.

1.2 Our Work

In this paper, we cluster trajectories using the Dirichlet process mixture model (DPMM) [10], [11], [12], [16], [17], [33], and then develop a trajectory-based video retrieval framework whose architecture is shown in Fig. 1. Our framework includes the trajectory pattern learning stage and the online trajectory retrieval stage. The trajectory pattern learning includes batch and incremental trajectory clustering using the DPMM and trajectory cluster modeling, which involves building a statistical model of trajectories in each cluster using the time-sensitive Dirichlet process mixture model (tDPMM).

1. **Clustering trajectories using DPMM:** Object trajectories are extracted using an existing object tracking method. Each object trajectory is represented by a discrete Fourier transform coefficient vector. In the DFT coefficient feature space, the extracted object trajectories are clustered using the DPMM. It is assumed that the trajectories are generated from a mixture of Gaussian distributions. The number of Gaussians is determined automatically, while the DPMM is learned. We further develop an incremental DPMM-based trajectory cluster growing method. When new trajectory data arrive, the incoming trajectories are assigned online to the existing trajectory clusters or to new trajectory clusters which are discovered by the cluster growing method with a low computational complexity. Each cluster of trajectories corresponds to a trajectory pattern.
2. **Modeling trajectory patterns using tDPMM:** In order to characterize the time-varying information about trajectories, each trajectory is segmented into several smaller subtrajectories. A DFT coefficient feature vector is used to represent each subtrajectory, and then a trajectory is represented as a sequence of DFT coefficient vectors. The DFT coefficient vector sequences in each trajectory cluster (i.e., trajectory pattern) are used to train a tDPMM which represents the unique spatiotemporal characteristics of the trajectory pattern. Then, the trajectory patterns are indexed using the parameters of the corresponding tDPMMs. Our incremental DPMM-based trajectory clustering method assigns online new incoming trajectory data to the existing trajectory patterns or new trajectory patterns. Each newly discovered trajectory pattern is modeled using the tDPMM and indexed by the corresponding parameters.
3. **Online trajectory retrieval:** In the online retrieval stage, a sketch-based scheme is used to represent a user-specified query. The query trajectory is matched to the corresponding trajectory pattern based on the posterior probability estimation. The retrieved videos are ranked by the posterior probabilities. As knowledge of the similarities between trajectories is included in the results of trajectory clustering, indexing each cluster of trajectories makes trajectory-based video retrieval more effective than searching the trajectories in the database for trajectories similar to the query, where searching the database can be sped up by using lower bounding measures [45].

The contributions of our work are listed below:

- The DPMM is applied to trajectory clustering. An appropriate number of trajectory clusters is automatically determined.
- We develop an incremental version of the DPMM-based clustering algorithm as an extension to the previous DPMM and apply it to incremental trajectory clustering. The trajectory clusters are adaptively grown after new trajectory samples arrive.
- A tDPMM is applied to model the time-varying information about a trajectory cluster learned from the DPMM. A parameterized index with only two values is built for each trajectory cluster.
- We propose a likelihood estimation method for the tDPMM. The method approximates the likelihood

using a collection of particles generated by Gibbs sampling. This likelihood extends the tDPMM.
- A probabilistic trajectory-based video retrieval model is developed. The probabilistic matching and growing of the retrieval model make the proposed retrieval model scalable and adaptive for online handling of the incoming trajectories.

The remainder of the paper is organized as follows: Section 2 introduces the extraction of trajectory features. Section 3 presents our DPMM-based trajectory clustering algorithm. Section 4 proposes our incremental trajectory cluster growing method. Section 5 describes our tDPMM-based algorithm for trajectory pattern modeling. Section 6 covers our algorithm for trajectory matching for video retrieval. Section 7 demonstrates experimental results. Section 8 summarizes the paper.

2 Trajectory Feature Extraction

In the following, we first describe the DFT-based trajectory feature representation, which is used for trajectory clustering, and then present the segmentation-based trajectory representation, which is used for trajectory pattern modeling.

2.1 DFT-Based Trajectory Feature Representation

Naftel and Khalid [8] proposed the DFT coefficient-based trajectory representation method. They showed that the DFT coefficient-based euclidean distance between trajectories is more robust than the original point-based euclidean distance. So, in our algorithm, DFT coefficients are used to represent object trajectories. Given a trajectory $(x_0, y_0), \ldots, (x_k, y_k), \ldots, (x_{n-1}, y_{n-1})$, where n is the number of points in the trajectory and (x_k, y_k) are the coordinates of the kth point ($0 \leq k \leq n-1$), the DFT coefficients are

$$\{F_x(0), \ldots, F_x(t), \ldots, F_x(T), F_y(0), \ldots, F_y(t), \ldots, F_y(T)\} \quad (1 \leq t \leq T \leq n-1), \tag{1}$$

where

$$F_x(t) = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} x_k \exp(-i2\pi tk/n), \tag{2}$$

$$F_y(t) = \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} y_k \exp(-i2\pi tk/n). \tag{3}$$

As $F_x(0)$ and $F_y(0)$ are real numbers, the dimension of DFT coefficient feature vectors is $2(2T+1)$. As T is set to a constant for all the trajectories, feature vectors extracted for all the trajectories have the same length, even though the number of points in each trajectory may vary. The euclidean measure is used to calculate the distance between any two DFT coefficient vectors.

2.2 Segmentation-Based Trajectory Representation

For characterizing time-varying information about a trajectory, a trajectory is segmented into a series of atomic subtrajectories which are regarded as semantically meaningful sections. Generally, segmentation of a trajectory into subtrajectories can be carried out using the variance of curve curvature [18], i.e., a trajectory is segmented at locations with large curvature. However, local noise due to tracking errors may produce large errors in curvature estimates because estimate of the curvature of a given point is only dependent on a few points near to the given point. Namely, noisy points may produce large curvatures in the locations where the true curvatures are small, and segmentation may mistakenly occur in these locations. Sun et al. [13] proposed a spectral clustering-based trajectory segmentation algorithm which is applied to bidirectional tracking. The position coordinates and the frame number of each point in a trajectory were used as features input to the clustering algorithm. Consecutive points which are close together or in sections of high curvature may be assigned to the same cluster. For example, in a traffic scene, the points in a vehicle trajectory section which is associated with vehicle turning are likely to be assigned into the same cluster to form a semantically meaningful section. As a local noisy point is usually near to points less affected by noise, it is likely that a noisy point and its adjacent points are assigned into the same cluster, i.e., there are fewer segmentation errors caused by noisy points. So, the clustering-based trajectory segmentation algorithm is less sensitive to noise. The spectral clustering method needs specification in advance of the number of subtrajectories as the K-means clustering is used in the last step of the spectral clustering. In order to tackle this problem, we propose an improved spectral clustering-based method which replaces the K-means clustering with the nonparametric adaptive mean shift clustering algorithm [19]. Its optimization goal is to find several modes which have large probability densities, where each mode corresponds to a cluster whose samples are strongly connected. As a result, the improved spectral clustering is capable of identifying an appropriate number of subtrajectories automatically. Fig. 2 shows three examples of segmenting a complex trajectory into several subtrajectories. The first rows in subfigures (a), (b), and (c) are the original trajectories, where the points in the trajectories are explicitly displayed. The second row in each of the subfigures is the segmentation result in which different subtrajectories are represented by different colors. It is clear that the segmentation results are good.

After the trajectory segmentation, a DFT coefficient feature vector is extracted for each subtrajectory. Then, a trajectory is represented by a sequence of DFT coefficient vectors:

$$SO = so_1, so_2, \ldots, so_i, \ldots, so_\Gamma, \tag{4}$$

where so_i is the DFT coefficient vector of the ith subtrajectory and Γ is the number of subtrajectories. This representation for a trajectory is used for trajectory pattern modeling.

3 DPMM-Based Trajectory Clustering

3.1 DPMM

Let Dir(.) denote the Dirichlet probability density function. Let A be a sample space, and $A_{1:K}$ be K disjoint subsets of A satisfying $A_1 \cup A_2 ... \cup A_k ... \cup A_K = A$. The Dirichlet process [10] with a scale parameter α and a base measure G_0 on A is defined as follows: If a random probability distribution G is sampled from the Dirichlet process, then:

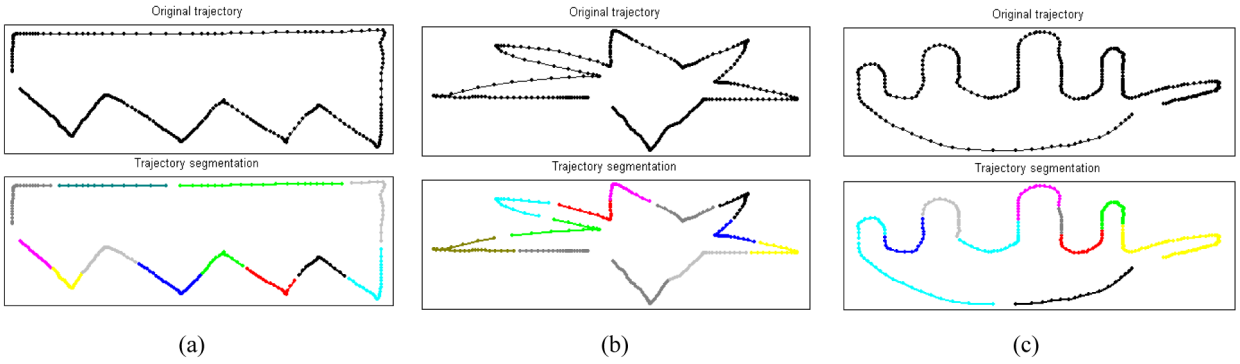


Fig. 2. Three examples of trajectory segmentation: (a) Example 1, (b) Example 2, (c) Example 3.

$$(G(A_1), G(A_2), \ldots, G(A_K)) \sim Dir(\alpha G_0(A_1), \alpha G_0(A_2), \ldots, \alpha G_0(A_K)). \quad (5)$$

The left-hand side of (5) is a random vector for fixed subsets of A because G is random. Let $\eta_{1:N}$ be a sequence of N state variables sampled from A using the distribution G, which is itself a sample from the Dirichlet process. On integrating over G, the joint distribution of $\eta_{1:N}$ under the Dirichlet process (α, G_0) is represented as

$$p(\eta_{1:N}|\alpha, G_0) = \int p(\eta_{1:N}|G)p(G|\alpha, G_0)dG. \quad (6)$$

This joint distribution exhibits a clustering effect [10]. Gibbs sampling is carried out for (6), and then a discrete joint distribution of $\eta_{1:N}$ is obtained. Maximum a posteriori estimation on this discrete joint distribution produces the most probable values of $\eta_{1:N}$ [33]. The obtained values are used to partition $\eta_{1:N}$: The ones with the same value are partitioned into the same cluster, and the ones with different values are partitioned into different clusters. More specifically, the value of η_i conditioned on the previous $i-1$ values of $\eta_{1:i-1}$ is equal to either one of the values of $\eta_{1:i-1}$ or a new value from the base measure G_0, where the new value, which means that a new cluster is generated, causes an appropriate number of clusters to be determinable automatically. The conditional probability distribution $p(\eta_i|\eta_{1:i-1})$ required for the Gibbs sampling [34] can be found using the Polya Urn scheme [11] and represented as

$$p(\eta_i|\eta_{1:i-1}) \propto \alpha G_0(\eta_i) + \sum_{j=1}^{i-1} \delta(\eta_i - \eta_j), \quad (7)$$

where $\delta(\varphi)$ is the Dirac delta function:

$$\delta(\varphi) = \begin{cases} 1 & if \varphi = 0 \\ 0 & otherwise. \end{cases} \quad (8)$$

As a result, the $\eta_{1:N}$ are randomly partitioned into clusters where the state variables with the same value are assigned into the same cluster. Let $\nu_{1:L}$ denote L different values that $\eta_{1:i-1}$ take (i.e., the L clusters to which the $\eta_{1:i-1}$ are assigned), and let m_l be the number of occurrences of ν_l in $\eta_{1:i-1}$ for $1 \le l \le L$ (i.e., the number of state variables assigned into cluster l). Then, (7) is transformed to

$$\eta_i = \begin{cases} \nu_l & \text{with probability } \frac{m_l}{i-1+\alpha} \\ & \text{for each distinct } l \\ \eta_{new}(\eta_{new} \sim G_0) & \text{with probability } \frac{\alpha}{i-1+\alpha}, \end{cases} \quad (9)$$

where η_{new} is a new state value which corresponds to a new cluster.

If the random variables $\eta_{1:N}$ are exchangeable, the marginal distribution $p(\eta_i|\eta_{-i})$ is formulated as

$$p(\eta_i|\eta_{-i}) \propto \alpha G_0(\eta_i) + \sum_{j \in -i} \delta(\eta_i - \eta_j), \quad (10)$$

where $-i$ denotes the indices in $1{:}N$ except for i.

If the Dirichlet process is used as a nonparametric prior in a hierarchical Bayesian model, the following DPMM [10], [11] [12], [16], [17] is obtained:

$$y_i \sim p(\cdot|\eta_i); \eta_i \sim G; G \sim (\alpha, G_0), \quad (11)$$

where y_i is the ith observation. Each observation y_i has its own state variable η_i. The value of η_i corresponds to the cluster label of y_i. Fig. 3 shows a standard graphical model for the DPMM, illustrating the way in which the DPMM generates the observations $\{y_i\}$. The process of learning, the reverse of the generative process, is to determine the values of the state variables $\{\eta_i\}$ given the observations $\{y_i\}$, using a strategy such as Gibbs sampling.

3.2 DPMM-Based Trajectory Clustering

We apply the Bayesian posterior probability estimate of a DPMM to cluster trajectories which correspond to observations in the DPMM. Object trajectories are specified by points in the DFT coefficient feature space represented by (1) and these points are clustered using the DPMM. Given N trajectories $y_{1:N}$ which are, of course, exchangeable, we estimate the posterior probability $p(\eta_{1:N}|y_{1:N})$ where each element η_i in $\eta_{1:N}$ denotes the state variable associated with y_i in $y_{1:N}$. Trajectories with the same state variable value are assigned to the same cluster. The probability $p(\eta_{1:N}|y_{1:N})$ is

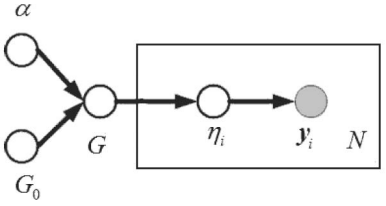


Fig. 3. Graphical model for the DPMM.

approximated by sampling from $p(\eta_i|\eta_{-i}, y_{1:N})$ $(1 \leq i \leq N)$ iteratively using the Gibbs sampler. It can be assumed that the current trajectory is only dependent on its own state, i.e., it is independent of the states of other trajectories. Then, according to Bayes' rule, the probability $p(\eta_i|\eta_{-i}, y_{1:N})$ is represented as

$$p(\eta_i|\eta_{-i}, y_{1:N}) \propto f(y_i|\eta_i)p(\eta_i|\eta_{-i}), \tag{12}$$

where $f(y_i|\eta_i)$ denotes the likelihood of the ith trajectory given the value of the state variable η_i. The likelihood $f(y_i|\eta_i)$ is assumed to be the Gaussian distribution. Its parameters, the mean, and the covariance matrix are estimated using a subset of the trajectories. We find the subset $\Upsilon = \{y_j|\eta_j = \eta_i, y_j \neq y_i\}$, which includes the trajectories with equal state values to η_i but excludes the trajectory y_i. The mean and covariance of the DFT coefficient vectors of the trajectories in Υ are used as the mean and covariance for $f(y_i|\eta_i)$. Substitution of (10) into (12) yields the following formula:

$$p(\eta_i|\eta_{-i}, y_{1:N}) \propto \alpha G_0(\eta_i)f(y_i|\eta_i) + \sum_{j \in -i} f(y_i|\eta_i)\delta(\eta_i - \eta_j). \tag{13}$$

In this paper, the base measure G_0 is assumed to be the Gaussian distribution $N(0, 1)$. Gibbs sampling, which is one type of Markov chain Monte Carlo (MCMC) sampling, is applied to (13) to obtain a maximum a posteriori estimate of the state variables $\eta_{1:N}$. The value of each state variable η_i is sampled using the conditional distribution $p(\eta_i|\eta_{-i}, y_{1:N})$ in which the other variables η_{-i} take the values of their most recent estimates. After the sampling for $\eta_{1:N}$ is repeated for sufficient times, the posterior $p(\eta_{1:N}|y_{1:N})$ is approximated. Then, the latent state labels associated with the trajectories $y_{1:N}$ are obtained. The trajectories associated with the same state value drawn from G are grouped into a single cluster.

4 Incremental Trajectory Cluster Growing

Incremental trajectory clustering is required in applications, such as visual surveillance. When new trajectories are added into the database of clustered trajectories, it is necessary to decide whether they belong to any of the existing clusters. If so, then the new trajectories are assigned to the corresponding clusters; otherwise new trajectory clusters are produced. This is called the incremental trajectory cluster growing process. We design the conditional distribution used for the Gibbs sampling in the incremental clustering by fixing the states of the previous samples and updating the states of new samples. It is assumed that there are, in the database, N trajectories $y_{1:N}$ whose cluster labels are known, and there are ϕ newly added trajectories which are represented by $y_{N+1:N+\phi}$. Let $W_{1:N}$ denote the known states (cluster labels) associated with $y_{1:N}$, and let η_{-i}^{new} $(N + 1 \leq i \leq N + \phi)$ represent all the state variables in $\eta_{N+1:N+\phi}$ except for η_i. Bayes' rule yields the following formula:

$$\begin{aligned} &p(\eta_i|\eta_{-i}, y_{1:N+\phi}) \\ &= p(\eta_i|\eta_{1:N} = W_{1:N}, \eta_{-i}^{new}, y_{1:N+\phi}) \quad (N + 1 \leq i \leq N + \phi) \\ &\propto p(y_i|\eta_i, \eta_{1:N} = W_{1:N}, \eta_{-i}^{new}, y_{-i}) \\ &\quad p(\eta_i|\eta_{1:N} = W_{1:N}, \eta_{-i}^{new}, y_{-i}). \end{aligned} \tag{14}$$

Due to the assumption that the current trajectory is dependent on its own state and but independent of the states of other trajectories, the following formulas are obtained:

$$\begin{aligned} &p\big(y_i|\eta_i, \eta_{1:N} = W_{1:N}, \eta_{-i}^{new}, y_{-i}\big) \\ &= p(y_i|\eta_i) \end{aligned} \quad (N + 1 \leq i \leq N + \phi), \tag{15}$$

$$\begin{aligned} &p\big(\eta_i|\eta_{1:N} = W_{1:N}, \eta_{-i}^{new}, y_{-i}\big) \\ &= p\big(\eta_i|\eta_{1:N} = W_{1:N}, \eta_{-i}^{new}\big) \end{aligned} \quad (N + 1 \leq i \leq N + \phi). \tag{16}$$

Then, we define the incremental cluster growing process as the following formula:

$$\begin{aligned} &p(\eta_i|\eta_{-i}, y_{1:N+\phi}) \\ &\propto p(y_i|\eta_i)p\big(\eta_i|\eta_{1:N} = W_{1:N}, \eta_{-i}^{new}\big) \end{aligned} (N + 1 \leq i \leq N + \phi), \tag{17}$$

where $p(y_i|\eta_i)$ is the data likelihood, which is assumed to be the Gaussian distribution as in (12), and $p(\eta_i|\eta_{1:N} = W_{1:N}, \eta_{-i}^{new})$ is estimated using the Polya Urn scheme as in (7). In this way, we only carry out Gibbs sampling on $\eta_{N+1:N+\phi}$, rather than on $\eta_{1:N+\phi}$, to determine the values of $\eta_{N+1:N+\phi}$. The data $y_{1:N}$ and their labels $W_{1:N}$ are only used to estimate $p(y_i|\eta_i)$. The computational complexity is greatly reduced compared with clustering the whole dataset $y_{1:N+\phi}$ using the original version of the DPMM.

In our incremental clustering method, the ϕ new state variables $\{\eta_i\}_{i=N+1,\ldots,N+\phi}$ are sampled from the joint distribution of all the states $\{\eta_j\}_{j=1,\ldots,N+\phi}$, where the first N states are fixed. The conditional distribution in (17) is derived from the global joint distribution of all the states $\{\eta_j\}_{j=1,\ldots,N+\phi}$. It is assumed that trajectories are sampled from Gaussian distributions in the coefficient feature space. A Gaussian distribution corresponds to a cluster of trajectories. The N previous trajectories are modeled by a mixture of Gaussians. Each of the new trajectories either belongs to one of the existing Gaussians or belongs to a new Gaussian. The parameters in the likelihood, the mean, and the covariance matrix are estimated using the trajectory subset $\Upsilon = \{y_j|\eta_j = \eta_i, y_j \neq y_i\}$, whether y_j belongs to the previous trajectories or is a new trajectory. The conditional distribution in (17) is designed using a standard mathematical inference. The above elements ensure that fixing the states of the N previous trajectories and sampling the states of the new trajectories in the joint distributions of the states of all the trajectories yield a valid joint distribution and do not influence the convergence of the sampling using (17).

5 Trajectory Pattern Modeling Based on tDPMM

Each cluster of trajectories is organized as a trajectory pattern which is modeled using the tDPMM based on the segmentation-based trajectory representation.

5.1 The tDPMM

The time-sensitive Dirichlet process mixture model [14] is a time-weighted version of the DPMM. It describes the sequential relations between state variables which are not exchangeable. A time series of observations is represented

by: $(o_1, s_1, t_1), \ldots, (o_N, s_N, t_N)$, where o_i is the observation at time t_i ($1 \leq i \leq N$ and $t_1 < \cdots < t_i < \cdots < t_N$), and s_i is the state variable associated with o_i. The tDPMM introduces a weight function $\omega(t, c)$ to characterize the influence of the history of the state variable sequence $s_{1:i}$ ($t_i < t$) on the state value c at time t:

$$\omega(t, c) = \sum_{\{m|t_m<t, s_m=c\}} k(t - t_m), \tag{18}$$

where

$$k(t) = \begin{cases} e^{-\lambda t} & if\ t > 0 \\ 0 & otherwise, \end{cases} \tag{19}$$

where the parameter λ is a positive constant. According to [14], the state transition distribution $p(s_i|s_{-i})$ in the tDPMM has the following form:

$$p(s_i|s_{-i}) \propto p(s_i|s_{1:i-1})\left(\prod_{n=i+1}^{N} p(s_n|s_{1:n-1})\right), \tag{20}$$

where s_{-i} denotes the remainder of $s_{1:N}$ after removing s_i. The probability $p(s_i|s_{1:i-1})$ is defined as

$$p(s_i|s_{1:i-1}) = \begin{cases} \dfrac{\omega(t_i, s_i)}{\alpha + \sum_{v=1}^{V} \omega(t_i, s_v^*)} & if\ s_i\ in\ s_{1:i-1} \\ \dfrac{\alpha}{\alpha + \sum_{v=1}^{V} \omega(t_i, s_v^*)} & otherwise, \end{cases} \tag{21}$$

where V is the number of unique values that the state variables in $s_{1:i-1}$ take, and s_v^* is the vth unique value in the set of the V unique values. The probability $p(s_n|s_{1:n-1})$ in (20) is estimated by replacing "i" in (21) with "n." It is obvious that the state transition distribution $p(s_i|s_{-i})$ in (20) is parameterized by $\Theta = \{\alpha, \lambda\}$, i.e., $p(s_i|s_{-i}) = p(s_i|s_{-i}, \Theta)$.

5.2 Trajectory Cluster Modeling

We model each cluster of trajectories using a tDPMM associated with a number of state variables where representation of trajectories by sequences of DFT coefficient vectors is used. In a trajectory cluster, DFT-based feature vectors of all the subtrajectories which correspond to a state variable are used to estimate the distribution of the state variable. Given a cluster of Ω trajectories $\{SO_j\}_{j=1,\ldots,\Omega}$, the number N of state variables in the tDPMM is determined by

$$N = \frac{1}{\Omega}\sum_{j=1}^{\Omega} \Gamma_j, \tag{22}$$

where Γ_j is the number of subtrajectories obtained by segmenting the jth trajectory using the method in Section 2.2. The ith observation o_i for the tDPMM is represented by the set of the DFT coefficient vectors of the subtrajectories $\{so_{\lfloor i*\Gamma_j/N \rfloor}\}_{j=1,\ldots,\Omega}$ in this cluster of trajectories.

According to [14], $p(s_i|s_{-i}, o_{1:N})$ is formulated as

$$p(s_i|s_{-i}, o_{1:N}) \propto p(s_i|s_{-i}) p(o_i|o_{-i:s_{-i}=s_i}), \tag{23}$$

where $o_{-i:s_{-i}=s_i}$ denotes the set of the subtrajectories in this cluster whose state values are equal to the value of s_i, except for the subtrajectories corresponding to o_i, and $p(s_i|s_{-i})$ is defined in (20). The distribution $p(o_i|o_{-i:s_{-i}=s_i})$ is assumed to be Gaussian, and the parameters are estimated using the subtrajectories corresponding to $o_{-i:s_{-i}=s_i}$. Thus, $p(s_{1:N}|o_{1:N})$ is approximated by sampling from $p(s_i|s_{-i}, o_{1:N})$ in (23) iteratively using the Gibbs sampler [14].

The tDPMM is capable of modeling long-range interaction dependencies of the latent state variables corresponding to the observations. Its state transition distribution (20) is governed by the kernel-weighted Dirichlet process (21). Due to the intrinsic properties of the Dirichlet process, tDPMM is very suitable for modeling time-series data with a countably infinite number of states. By maximum a posteriori estimation of the state variables from $p(s_{1:N}|o_{1:N})$, the most probable latent state label sequence associated with the observation sequence can be obtained.

5.3 Parameter Estimation

The model parameters $\Theta = \{\alpha, \lambda\}$ of the tDPMM for each trajectory cluster are learned using the stochastic EM algorithm [14]. Let $Q(\Theta_0, \Theta)$ be the complete data log likelihood function, where Θ_0 is the currently estimated vector of the model parameters and Θ is the vector of the parameters in the next estimation step. A cycle of sampling states $S_1, S_2, \ldots, S_N$ from (23) using the Gibbs sampler forms a particle. Let $\{S^{(m)} = S_1^{(m)} S_2^{(m)}, \ldots, S_N^{(m)}\}_{m=1}^{M}$ be M particles yielded by the Gibbs sampling. The parameters are updated using the first order gradients of Q on Θ:

$$\frac{\partial Q}{\partial \alpha} = \frac{1}{M}\sum_{m=1}^{M}\sum_{i=1}^{N} \frac{\partial}{\partial \alpha} \log p\big(s_i^{(m)}|s_{1:i-1}^{(m)}, T, \Theta\big), \tag{24}$$

$$\frac{\partial Q}{\partial \lambda} = \frac{1}{M}\sum_{m=1}^{M}\sum_{i=1}^{N} \frac{\partial}{\partial \lambda} \log p\big(s_i^{(m)}|s_{1:i-1}^{(m)}, T, \Theta\big), \tag{25}$$

where $T = t_{1:N}$ and the $p(s_i^{(m)}|s_{1:i-1}^{(m)}, T, \Theta)$ is defined in (21). The gradients are

$$\frac{\partial}{\partial \alpha} \log p\big(s_i^{(m)}|s_{1:i-1}^{(m)}, T, \Theta\big) = \begin{cases} \dfrac{-1}{\alpha + \sum_{v=1}^{V} w\big(t_i, s_v^*\big)} & if\ s_i\ in\ s_{1:i-1} \\ \dfrac{1}{\alpha} - \dfrac{1}{\alpha + \sum_{v=1}^{V} w\big(t_i, s_v^*\big)} & otherwise, \end{cases} \tag{26}$$

$$\frac{\partial}{\partial \lambda} \log p(s_i^{(m)}|s_{1:i-1}^{(m)}, T, \Theta) = \begin{cases} \dfrac{\frac{\partial}{\partial \lambda} w(t_i, s_i)}{w(t_i, s_i)} - \dfrac{\sum_{v=1}^{V} \frac{\partial}{\partial \lambda} w(t_i, s_v^*)}{\alpha + \sum_{v=1}^{V} w(t_i, s_v^*)} & if\ s_i\ in\ s_{1:i-1} \\ -\dfrac{\sum_{v=1}^{V} \frac{\partial}{\partial \lambda} w(t_i, s_v^*)}{\alpha + \sum_{v=1}^{V} w(t_i, s_v^*)} & otherwise, \end{cases} \tag{27}$$

where

$$\frac{\partial}{\partial \lambda} \omega(t, c) = \sum_{\{j|t_j<t, s_j=c\}} -(t - t_j) e^{-\lambda(t-t_j)}. \tag{28}$$

5.4 Likelihood Estimation for tDPMM

The original version of the tDPMM [14] did not give the inference mechanism for likelihood estimation which is required for trajectory retrieval, which is the application we are concerned with in this paper. In the following, we

propose a Gibbs sampling-based observation likelihood estimation algorithm for the tDPMM as an extension to the tDPMM.

For an observation sequence $O = o_{1:N}$, the likelihood $p(O|\Theta)$ conditioned on the learned tDPMM parameterized by Θ ($\Theta = \{\alpha, \lambda\}$) satisfies

$$p(O|\Theta) = \int p(O|S)p(S|\Theta)dS, \tag{29}$$

where $S(S = s_{1:N})$ is the latent state sequence corresponding to $O(O = o_{1:N})$. The computational cost of (29) is high because of the integration over all the values of S. To simplify the computation, we approximate $p(O|\Theta)$ by the summation weighted by the particles sampled from the conditional distribution $p(s_i|s_{-i}) = p(s_i|s_{-i}, \Theta)$ in (20) using the Gibbs sampler given the estimated model parameters. It can be assumed that $O = o_{1:N}$ are mutually independent given $S = s_{1:N}$, i.e., $p(O|S) = \prod_{n=1}^{N} p(O_n|S_n)$. Let $S^{(m)}$ ($S^{(m)} = s_{1:N}^{(m)}$) denote the particle sampled at the mth Gibbs sampling iterative step. The likelihood $p(O|\Theta)$ is computed by:

$$\begin{aligned} p(O|\Theta) &\approx \frac{1}{M}\sum_{m=1}^{M} p\left(O|S^{(m)}\right) \\ &= \frac{1}{M}\sum_{m=1}^{M} p\left(o_{1:N}|s_{1:N}^{(m)}\right) \\ &= \frac{1}{M}\sum_{m=1}^{M}\left(\prod_{n=1}^{N} p(o_n|s_n^{(m)})\right), \end{aligned} \tag{30}$$

where M denotes the number of particles. The $p(o_n|s_n^{(m)})$ is assumed to be the Gaussian distribution whose parameters are estimated using the DFT coefficient vectors of the subtrajectories in the trajectory pattern whose state values are equal to the value of $s_n^{(m)}$.

6 Matching and Growing for Video Retrieval

We consider the video retrieval in which each trajectory is associated with a video clip and the trajectories are indexed by the parameters α and λ of the learned trajectory patterns. The query is represented by a trajectory which is transformed to a DFT coefficient vector sequence which is described using (4). Given the query trajectory R, the posterior probability that R belongs to the jth trajectory pattern is estimated by

$$p(\Theta_j|R) \propto p(R|\Theta_j)p(\Theta_j), \tag{31}$$

where Θ_j ($\Theta_j = \{\alpha_j, \lambda_j\}$) is the tDPMM's parameter vector for the jth trajectory pattern, and $p(R|\Theta_j)$ is the likelihood, which is estimated using (30). Each trajectory pattern is assumed to have the same prior probability. Then, the matches between R and the trajectory patterns are ranked according to the magnitude of the posterior probabilities in (31).

When new trajectories are added into the indexed trajectory database, any new trajectory patterns can be found online using the incremental DPMM-based clustering in Section 4. Subsequently, for each new trajectory cluster, the parameters α and λ of a tDPMM are learned and used as the index of the trajectory pattern. The tDPMMs of the newly generated trajectory clusters are then added to the indexed database. This incremental growing capability results in scalability and adaptability properties for our algorithm.

7 Experiments

The performance of the proposed trajectory clustering, modeling, and retrieval algorithm was evaluated using the following three trajectory datasets:

- The synthetic trajectory dataset:[1] This dataset contains 2,500 trajectories from 50 clusters. Each cluster consists of 50 trajectories with complex shapes.
- The hand sign dataset:[2] This dataset is from the Australian sign language collection. There are, in total, 35 clusters of trajectories of hand sign words. Each cluster consists of 20 trajectories of hand movements of different signers. As in the previous work [7], [8] on trajectory analysis, we only used the 2D (x, y)-coordinate features from this dataset in which the original data are high dimensional, as the (x, y) coordinates are the fundamental features of trajectories.
- The vehicle motion trajectory dataset: In this dataset, there are 1,500 trajectories which were collected by tracking vehicles in a real traffic scene and labeled manually to produce 15 clusters.

All the datasets used in the experiments are available at http://159.226.19.33/dataset/trajectory/dataInfo.htm.

In the experiments, the dimension of the DFT coefficient feature space was set to 18. The scale parameter α in the DPMM was initialized as 0.2. The number M of particles in (30) was set to 200.

To demonstrate the claimed contributions of the proposed algorithm, we evaluated, in succession, the accuracy of trajectory pattern finding, the trajectory learning accuracy, the ability to deal with noise in trajectories, the incremental cluster growing capability, and the trajectory retrieval accuracy.

7.1 Trajectory Pattern Finding

We quantitatively compared our algorithm with the mean shift clustering for finding the number of trajectory patterns on the synthetic trajectory dataset and the hand sign dataset. Trajectories with known cluster labels were used to test the effectiveness of the strategies for determining an appropriate number of clusters of trajectories. We randomly selected 30 subsets from each of the two datasets. Each cluster in a subset includes more than half the trajectories in the cluster in the corresponding dataset. For our algorithm, half of the trajectories in each subset were clustered using the DPMM-based algorithm and, based on the found clusters, the incremental DPMM-based algorithm was used to cluster the remaining trajectories. The mean shift clustering algorithm clustered all the trajectories in each subset simultaneously. The test process was carried out 30 times for each of the two datasets, i.e., one time for each subset. For each algorithm, the number of learned clusters

1. http://mmplab.eed.yzu.edu.tw/trajectory/trajectory.rar.
2. http://kdd.ics.uci.edu/databases/auslan2/auslan.data.html.

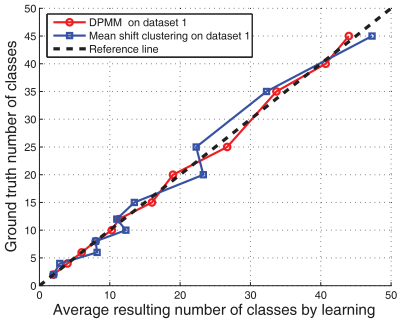


Fig. 4. Numbers of clusters learned using our algorithm and the mean shift clustering algorithm tested on the synthetic trajectory dataset.

was averaged over the subsets with the same ground-truth cluster number.

Figs. 4 and 5 show the curves of the average numbers of the trajectory clusters learned using our algorithm and the mean shift clustering algorithm, for subsets of the synthetic trajectory dataset and subsets of the hand sign dataset, respectively. The curves are obtained by linking the points whose x-coordinates are the average numbers of the learned clusters and y-coordinates are the ground truths of the numbers of the clusters, sorted in the ascending order of the y-coordinates. In the figures, the dashed curves are the ground-truth curves, the thin red solid curves correspond to our algorithm, and the blue solid curves correspond to the mean shift clustering algorithm. The closer to the ground-truth curves the learned curves are, the more accurate the numbers of the trajectory clusters learned by the algorithm. From Figs. 4 and 5, it is seen that the fluctuations of the curve for our algorithm about the benchmark of the ground-truth curve are much less than those for the mean shift clustering algorithm.

We also calculated the average of the absolute values of deviations of the estimated numbers of clusters obtained using these two algorithms from the ground-truth cluster numbers. Tested on the synthetic trajectory dataset, the corresponding averages for our algorithm and the mean shift clustering algorithm are 0.56 and 0.61, respectively. Tested on the hand sign dataset, the corresponding averages for these two algorithms are 1.74 and 2.08, respectively. It is seen that our algorithm is more accurate at learning the number of trajectory clusters than the mean shift clustering algorithm.

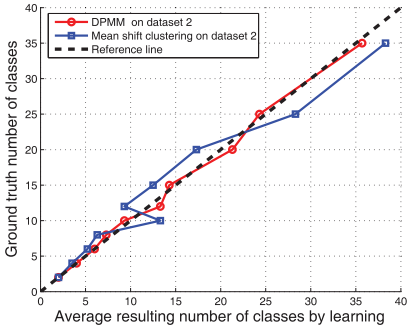


Fig. 5. Numbers of clusters learned using our algorithm and the mean shift clustering algorithm tested on the hand sign dataset.

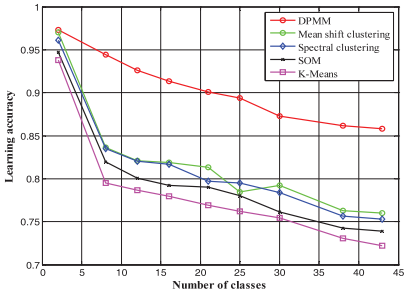


Fig. 6. Trajectory pattern learning accuracies of our algorithm, the mean shift clustering, the spectral clustering, the self-organizing mapping, and the K-means clustering tested on the synthetic trajectory dataset.

7.2 Learning Accuracy

The trajectory pattern learning accuracy $\Re$ is defined as:

$$\Re = \frac{1}{\mathbb{N}} \sum_{i=1}^{\mathbb{N}} \frac{b_i}{B_i}, \tag{32}$$

where $\mathbb{N}$ is the number of the learned clusters, B_i is the number of the trajectories in the ith learned cluster, and b_i is the number of the trajectories which have a fixed ground-truth cluster label and have the highest proportion in the ith learned cluster. With respect to trajectory pattern learning accuracy, we first compared our algorithm with four classic algorithms and then with the algorithms based on the euclidean distance and the DTW distance.

7.2.1 *Comparison with Four Classic Algorithms*

We compared the trajectory pattern learning accuracy of our algorithm with those of four classic unsupervised learning algorithms: mean shift clustering [15], spectral clustering, self-organizing mapping [2], and K-means clustering, tested on the synthetic dataset. The parameters of the four competing learning algorithms were chosen during the course of the experiments to make the clustering results as accurate as possible.

In the experiments, 30 subsets were randomly selected from the dataset to test the trajectory pattern accuracies of our algorithm and the four competing algorithms. For our algorithm, half of the trajectories in each subset were clustered using the DPMM-based algorithm, and the incremental DPMM-based algorithm clustered the remaining trajectories. The four competing algorithms clustered all the trajectories in each subset simultaneously. We measured the average trajectory pattern accuracy for the subsets with the same ground-truth cluster number. The five curves of the average trajectory pattern accuracies for these five algorithms are shown in Fig. 6, where each curve is obtained by linking the points whose x-coordinates are the ground-truth numbers of the clusters and y-coordinates are the learning accuracies, sorted in ascending order of the number of clusters. From this figure, it is seen that the learning accuracy of our algorithm is always higher than those of the four competing methods. The learning accuracy of our algorithm tends to be much more stable than those of the four competing methods as the number of trajectory clusters increases. The ability of our algorithm to predict the number of clusters and the stability of the trajectory patterns learned

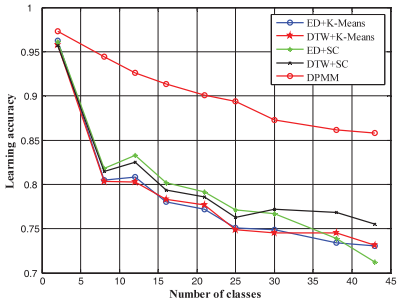


Fig. 7. Comparison between our algorithm and the four algorithms which use the euclidean distance or the DTW distance as the trajectory similarity measure and use spectral clustering or K-means as the clustering algorithm, tested on the synthetic trajectory dataset.

using our algorithm effectively facilitate the higher level trajectory-based video retrieval.

7.2.2 Comparison with the euclidean and DTW Distances

We compared our DPMM-based trajectory clustering algorithm with the four algorithms which use the euclidean distance (ED) or the DTW distance as the trajectory similarity measure and use spectral clustering or K-means as the clustering algorithm.

Fig. 7 shows the curves of the average trajectory pattern accuracies for these five algorithms, tested on the synthetic trajectory dataset, where the test setting is the same as in Fig. 6. It is seen that our algorithm obtains more accurate results than the four algorithms based on the euclidean distance and the DTW distance. The reasons for this are that our DPMM-based clustering algorithm obtains more accurate results than the spectral clustering algorithm and the K-means clustering algorithm, and the DFT-based distance is more effective than the point-based distances for measuring the similarity between trajectories [8], [36].

The hand sign dataset was used to compare our algorithm with the four algorithms based on the euclidean distance or the DTW distance. We randomly selected 30 subsets from the dataset and for each algorithm we measured the average learning accuracy for the subsets. The results are shown in Table 1. It is seen that our algorithm obtains more accurate results than the four competing algorithms. The DTW distance-based algorithms work more accurately than the euclidean distance-based algorithms, as the DTW distance is more appropriate than the euclidean distance for measuring the shape similarity between trajectories and the trajectories in the hand sign collection have complex shapes.

TABLE 1
Learning Accuracies Tested on the Hand Sign Dataset

Algorithms	Learning accuracy
ED + K-Means	67.2%
ED + Spectral Clustering	69.9%
DTW + K-Means	70.1%
DTW + Spectral Clustering	73.2%
Our algorithm	74.2%

TABLE 2
Learning Accuracies Tested on the Traffic Dataset

Algorithms	Learning accuracy
ED + K-Means	82.6%
ED + Spectral clustering	85.0%
DTW + K-Means	83.2%
DTW + Spectral clustering	85.3%
Our algorithm	86.7%

The traffic vehicle trajectory dataset was also used to compare our algorithm with the four algorithms based on the euclidean distance or the DTW distance. The learning accuracy results obtained from the whole dataset are shown in Table 2. It is seen that our algorithm obtains more accurate results than the four competing algorithms. The results of the euclidean distance-based algorithms are almost similar to the results of the DTW distance-based algorithms as shapes of the traffic vehicle trajectories are too simple to show the benefits of the DTW. The high computational cost puts the DTW distance at a disadvantage.

7.3 Learning with Noise

We tested the ability of our algorithm to deal with broken trajectories and noises caused by tracking errors, etc.

Broken trajectories can be caused by occlusion or loss of tracking. To simulate broken trajectories, we omitted the initial or last G points in two of 10 trajectories in each cluster in the traffic dataset. The labels of the trajectories were kept unchanged. Four datasets were produced, corresponding to values of 10, 20, 30, and 40 percent for G. The results from these four test sets are shown in Table 3 for our algorithm and the two competing algorithms which measure trajectory similarity using the euclidean distance and the DTW distance, respectively, and which both use spectral clustering as the clustering algorithm. It is seen that the learning accuracies of all the three algorithms decrease when the percentage of omitted points increases. Our algorithm obtains the best results among the three algorithms.

To simulate noises due to tracking errors, we added Gaussian noise to all the points in each trajectory. Three different levels of added noises were used to produce three datasets of trajectories with noise. Table 4 shows the results obtained from the three datasets for our algorithm and the two competing algorithms which measure trajectory similarity using the euclidean distance and the DTW distance, respectively, and which both cluster trajectories using the spectral clustering algorithm. It is seen that the learning accuracies of all the three algorithms decrease when the added noise increases (from Level 1 to Level 3). Our algorithm obtains the best results among the three algorithms for all three datasets. For the dataset with the largest noise level, the euclidean distance-based algorithm fails, while a good result is still maintained for our algorithm.

TABLE 3
Learning Accuracies When Some Trajectories Are Broken

Datasets with Point omission	Learning accuracy		
	ED + Spectral clustering	DTW + Spectral clustering	Our algorithm
10%	84.4%	85.1%	86.1%
20%	82.7%	83.5%	85.7%
30%	79.3%	81.1%	81.8%
40%	76.1%	77.5%	78.1%

TABLE 4
Learning Accuracies with Gaussian Noise

Datasets with noise	Learning accuracy		
	ED + Spectral clustering	DTW + Spectral clustering	Our algorithm
Level 1	83.3%	83.5%	84.3%
Level 2	81.2%	81.9%	83.3%
Level 3	51.2%	74.6%	81.5%

One reason why our algorithm is less sensitive to tracking errors and broken trajectories than the two competing algorithms is that the DFT-based distance is more robust than the point-based distances [8], [36].

7.4 Incremental Growing Capability

The results in Sections 7.1, 7.2, and 7.3 partly show the incremental growing capability of our algorithm. In this section, this capability of our algorithm is further quantitatively illustrated using the synthetic dataset. In the experiments, after some trajectory patterns are learned, new trajectories belonging to new clusters are added. The number of new trajectory clusters is found using the incremental DPMM-based clustering algorithm. We chose five different numbers of newly added clusters: 1, 3, 5, 7, and 9. For each of these five numbers, we randomly selected 10 subsets of trajectories from the dataset such that each of the subsets had the given number of newly added clusters. The number of new clusters in each of the subsets is found by our algorithm. The average of the numbers of new found clusters for the 10 subsets is set as the estimated number of new clusters. Fig. 8 shows the estimated numbers of new clusters for all five numbers, 1, 3, 5, 7, and 9, versus the ground-truth numbers of new clusters, where the x- and y-coordinates are the ground truth and estimated numbers, respectively; the solid curve, which is sorted in the ascending order of the ground-truth numbers, corresponds to the learned results, and the dashed line is the reference that indicates the perfect matching. It is seen that the fluctuations in the learned curve away from the perfect matching line are very small, i.e., our algorithm identifies new clusters with a very small error.

We made a comparison of learning accuracy between the following cases with different proportions of data used for batch and incremental learning:

- all the trajectories in the test set were clustered in batch mode by the DPMM;
- half the trajectories in the test set were clustered using the batch DPMM and the remainder were clustered using the incremental DPMM;
- all the trajectory clusters were learned by the incremental DPMM.

Table 5 shows the results for the three cases tested using the synthetic trajectory dataset, the sign dataset, and the traffic dataset, respectively, where the accuracies for the synthetic trajectory dataset and the hand sign dataset are the averages for 30 subsets, as in Fig. 7 and Table 1, and the accuracy for the traffic dataset is estimated from the whole dataset. It is seen that the fewer the trajectories there are in batch mode, the less accurate the clustering results are. This is inevitable for an incremental clustering algorithm. But our incremental trajectory clustering still maintains good accuracy compared with the batch trajectory clustering algorithms shown in Tables 1 and 2.

In the experiments, our incremental DPMM sampling converged in every case. Fig. 9 shows the convergence of the incremental DPMM-based clustering when half the trajectories in the synthetic dataset are incrementally clustered. It is seen that as the number of the Gibbs

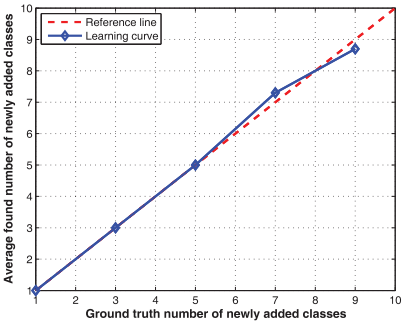


Fig. 8. The learning results of incremental cluster growing using the synthetic dataset.

TABLE 5
Learning Accuracies with Different Proportions of Data for Batch and Incremental Learning Tested Using the Synthetic Trajectory Dataset, the Hand Sign Dataset, and the Traffic Dataset, Respectively

Different proportions of data	Learning accuracy		
	The synthetic dataset	The hand sign dataset	The traffic dataset
All trajectories are clustered in batch mode	86.5%	75.3%	90.0%
Half-batch and half-incremental	81.8%	74.2%	86.7%
All-incremental	80.2%	74.1%	85.1%

sampling rounds increases, the learning accuracy is increased until it reaches a stable value.

7.5 Retrieval Accuracy

The following experiments were implemented to evaluate the performance of the proposed trajectory-based video retrieval algorithm. We compared our video retrieval algorithm, using the synthetic trajectory dataset, with the following algorithms:

- Two typical state-of-the-art trajectory-based video retrieval algorithms in the recent literature: the PCA-based algorithm in [18] and the algorithm based on string matching and polynomial fitting in [9].
- The euclidean distance-based algorithm and the DTW distance-based algorithm, both of which search all the trajectories in the dataset for the desired trajectories.

The recall rate and the precision rate were used to quantitatively measure the performance of these five trajectory-based video retrieval algorithms. The recall rate and the precision rate are defined as

$$recall = \frac{B_r}{B_r + B_o}, \quad precision = \frac{B_r}{B_r + B_p}, \tag{33}$$

where B_r is the number of the retrieved trajectories which satisfy the query requirement, B_o is the number of the unretrieved samples which satisfy the query requirement, and B_p is the number of the retrieved samples which do not satisfy the query requirement.

The number of the retrieved samples which are ranked by the posterior probabilities controls the recall. The higher this number is, the higher the recall. In the experiments, for 30 different queries we calculated the average precision with a fixed recall. Fig. 10 shows the average recall-precision curves of our algorithm, the PCA-based algorithm, the algorithm based on string matching and polynomial fitting, the euclidean distance-based algorithm, and the DTW distance-based algorithm, where x-coordinates represent recalls and y-coordinates represent precisions, sorted in ascending order of the recalls. It is seen that the curve of our algorithm is above the curves of the two state-of-the-art algorithms and the two exhaustive search-based algorithms. Trajectory retrieval by indexing trajectory clusters is more effective than trajectory retrieval by matching the query with all the trajectories in the dataset.

The recall-precision area, which is defined as the area underneath the recall-precision curve, is a criterion for quantitatively evaluating the quality of the recall and precision. The larger the recall-precision area is, the more accurate the retrieval result. It is shown that the recall-precision area of our algorithm is larger than those of the two state-of-the-art algorithms by more than 11.2 and 4.9 percent, respectively, and larger than those of the euclidean distance-based algorithm and the DTW distance-based algorithm, by more than 20.5 and 20.1 percent, respectively. This indicates that our algorithm has higher retrieval accuracy than the four competing algorithms.

Fig. 11 illustrates a retrieval example in which the query is a full trajectory, tested using the synthetic trajectory dataset. In Fig. 11a, a pentacle-like full trajectory was drawn manually as a query. In Fig. 11b, 12 matching clusters of trajectories are shown, where each of the 12 trajectories represents a cluster of trajectories and the displayed trajectory is selected from the cluster and has the maximum likelihood for its cluster. These 12 trajectory clusters are ranked from left to right and from up to down, according to

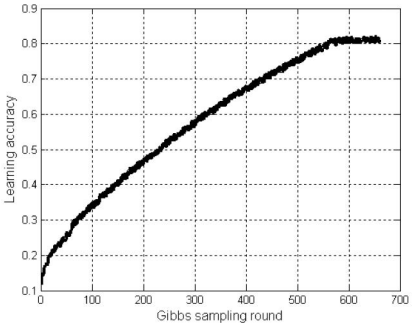


Fig. 9. Convergence of the proposed incremental DPMM-based clustering.

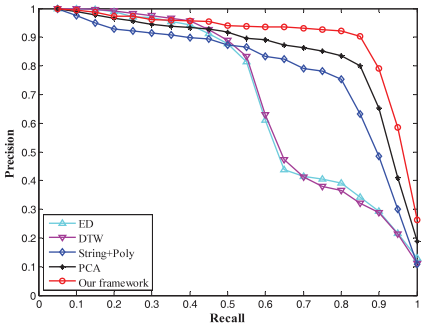


Fig. 10. Recall-precision curves of our algorithm, the PCA-based algorithm, the algorithm based on string matching and polynomial fitting, the ED-based algorithm, and the DTW distance-based algorithm.

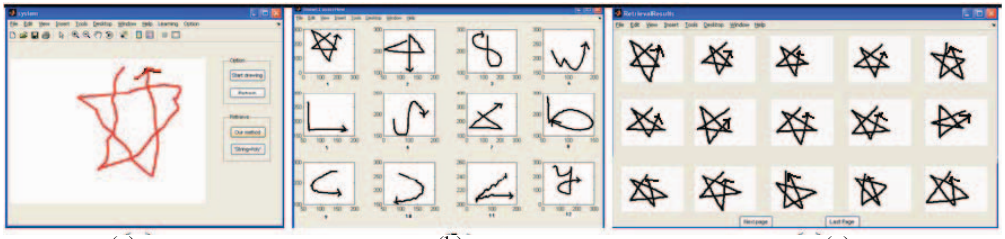


Fig. 11. A retrieval example in which the query is a full trajectory: (a) the query trajectory, (b) the retrieved trajectory patterns ranked from left to right and from top to bottom, (c) trajectory samples corresponding to the pattern most similar to the query trajectory.

Fig. 12. A retrieval example in which the query is a partial trajectory: (a) the query trajectory, (b) the retrieved trajectory patterns ranked from left to right and from top to bottom, (c) trajectory samples corresponding to the pattern most similar to the query trajectory.

the probabilities of the clusters given the query. The 15 trajectories which are shown in Fig. 11c are obtained from the cluster which is the most similar to the query, i.e., corresponds to the top-left subgraph in Fig. 11b. It is seen that the ranking of the matching trajectory patterns shown in Fig. 11b is consistent with their similarities to the query trajectory, and the retrieved trajectories shown in Fig. 11c are similar to the query trajectory.

Fig. 12 illustrates a retrieval example in which the query is a partial trajectory, tested on the synthetic trajectory dataset. A manually drawn "α"-like partial trajectory shown in Fig. 12a was input as a query. By matching using the posterior probability in (31), the 12 candidate trajectory clusters shown in Fig. 12b are retrieved and ranked from left to right and from up to down. The 15 trajectories in the cluster shown in the top-left subgraph in Fig. 12b are shown in Fig. 12c. This example indicates that our retrieval algorithm has the capability for partial trajectory matching in response to a partial trajectory query in video retrieval. This is because the local spatiotemporal property of a trajectory is effectively captured in the tDPMM.

Fig. 13 shows three retrieval examples, tested on the vehicle trajectory dataset. In Fig. 13a, a trajectory drawn manually represents a user query to retrieve video clips containing vehicles moving down and then turning right on the road. In Fig. 13b, the four retrieved video clips which match the query in Fig. 13a are displayed. In Fig. 13c, a query for video clips containing vehicles that make the "U" turn is given. The corresponding retrieved results are shown in Fig. 13d. In Fig. 13e, a query for video clips containing vehicles moving from left to right in the image and then turning left on the road is input, and in Fig. 13f, the corresponding retrieved results are shown. All the results shown in Figs. 13b, d, and f are correct.

From the above five retrieval examples, it is seen that our algorithm can perform retrieval for both full and partial trajectory queries. The trajectory pattern-based index enables an efficient search of the database.

7.6 Discussion

While our method focuses on dealing with 2D coordinate trajectories, our method can be extended to multivariate trajectories by adding DFT coefficients for each of the newly added point feature vector components to (1). The challenge in applying our method to high-dimensional multivariate trajectories is to automatically determine each point vector component's weights, which depend on the attributes and significance of each point feature vector component.

The trajectories in the vehicle dataset are acquired from the same viewpoint. As the vehicle dataset is captured from an outdoor surveillance scene far from the camera, small changes in viewpoints tend not to bring about large changes in trajectories features. Our approach is adaptable to such small changes. For uncalibrated trajectories acquired from

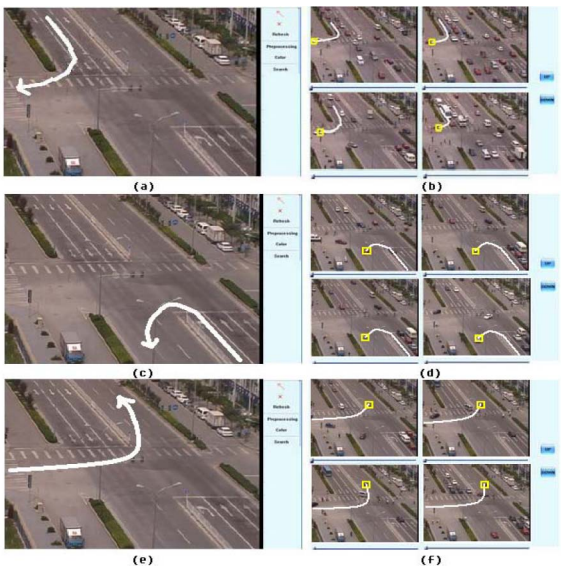


Fig. 13. Three retrieval examples tested on the traffic dataset: (b) shows the results for the query in (a), (d) shows the results for the query in (c), and (f) shows the results for the query in (e).

different viewpoints, the DFT features of trajectories are less sensitive to changes in viewpoints than the trajectory features in the original geometrical space. Our method can be extended to the view-invariant trajectory analysis. We can manually label some of the uncalibrated trajectories which are acquired from different viewpoints and belong to the same cluster in the physical space. A semi-supervised DPMM can be constructed to cluster all the trajectories from different viewpoints in order to automatically assign the trajectories belonging to the same cluster in the physical space to the same cluster. In our future work, we will investigate this issue.

8 Conclusion

In this paper, the DPMM has been applied to trajectory clustering, modeling, and retrieval. We have proposed an incremental DPMM-based clustering algorithm and applied it to trajectory clustering. An appropriate number of trajectory clusters can be automatically and incrementally determined. The tDPMM has been applied to learn the trajectory pattern which captures the time-series characteristics of the trajectories in a learned cluster. A parameterized index has been constructed for each pattern. In particular, we have proposed a likelihood estimation algorithm for the tDPMM. Then, a trajectory-based video retrieval model has been developed. The probabilistic model matching and incremental growing ensure that the model is scalable and adaptable: When new incoming data with new clusters arrive, the model automatically identifies the new clusters without retraining. Experimental results have demonstrated the superiority of the proposed incremental trajectory clustering algorithm and the proposed trajectory-based video retrieval model to the peer methods in the recent literature.

Acknowledgments

The authors thank Drs. Xiaoqin Zhang and Guan Luo for their valuable suggestions on the work. This work is partly supported by the Natural Science Foundation of China (Grant No. 60825204, 60935002), the National 863 High-Tech R&D Program of China (Grant No. 2012AA012504), the Natural Science Foundation of Beijing (Grant No. 4121003), US National Science Foundation (NSF) (IIS-0812114, CCF-1017828), National Basic Research Program of China (2012CB316400), and Alibaba Financial, Zhejiang University Joint Research Lab.

References

[1] J.J. Little and Z. Gu, "Video Retrieval by Spatial and Temporal Structure of Trajectories," *Proc. SPIE Storage and Retrieval for Media Databases,* vol. 4315, pp. 545-552, 2001.

[2] N. Johnson and D. Hogg, "Learning the Distribution of Object Trajectories for Event Recognition," *Image and Vision Computing,* vol. 14, no. 8, pp. 609-615, 1996.

[3] X. Ma, F. Bashir, A.A. Khokhar, and D. Schonfeld, "Event Analysis Based on Multiple Interactive Motion Trajectories," *IEEE Trans. Circuits and Systems for Video Technology,* vol. 19, no. 3, pp. 397-406, Mar. 2009.

[4] S. Atev, G. Miller, and N.P. Papanikolopoulos, "Clustering of Vehicle Trajectories," *IEEE Trans. Intelligent Transportation Systems,* vol. 11, no. 3, pp. 647-657, Sept. 2010.

[5] J. Alon, S. Sclaroff, G. Kollios, and V. Pavlovic, "Discovering Clusters in Motion Time-Series Data," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 1, pp: I-375-I-381, 2003.

[6] E. Sahouria and A. Zakhor, "Motion Indexing of Video," *Proc. Int'l Conf. Image Processing,* vol. 2, pp. 526-529, Oct. 1997.

[7] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering Similar Multidimensional Trajectories," *Proc. Int'l Conf. Data Eng.,* pp. 673-684, 2002.

[8] A. Naftel and S. Khalid, "Motion Trajectory Learning in the DFT-Coefficient Feature Space," *Proc. IEEE Int'l Conf. Computer Vision Systems,* pp. 47-47, Jan. 2006.

[9] J. Hsieh, S. Yu, and Y. Chen, "Motion-Based Video Retrieval by Trajectory Matching," *IEEE Trans. Circuits and Systems for Video Technology,* vol. 16, no. 3, pp. 396-409, Mar. 2006.

[10] T. Ferguson, "A Bayesian Analysis of Some Non-Parametric Problems," *Ann. of Statistics,* vol. 1, no. 2, pp. 209-230, Mar. 1973.

[11] D. Blackwell and J.B. MacQueen, "Ferguson Distribution via Polya Urn Schemes," *Ann. of Statistics,* vol. 1, no. 2, pp. 353-355, 1973.

[12] D.M. Blei and M.I. Jordan, "Variational Methods for the Dirichlet Process," *Proc. Int'l Conf. Machine Learning,* pp. 121-144, 2004.

[13] J. Sun, W. Zhang, X. Tang, and H. Shum, "Bidirectional Tracking Using Trajectory Segment Analysis," *Proc. IEEE Int'l Conf. Computer Vision,* vol. 1, pp. 717-724, 2005.

[14] X. Zhu, Z. Ghahramani, and J. Lafferty, "Time-Sensitive Dirichlet Process Mixture Models," Technical Report CMUCALD-05-104, School of Computer Science, Carnegie Mellon Univ., 2005.

[15] D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach toward Feature Space Analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 5, pp. 603-619, May 2002.

[16] R. Neal, "Markov Chain Sampling Methods for Dirichlet Process Mixture Models," *J. Computational and Graphical Statistics,* vol. 9, no. 2, pp. 249-265, June 2000.

[17] C. Zhang, S. Zu, and Y. Gong, "Trend Analysis for Large Document Streams," *Proc. Int'l Conf. Machine Learning and Applications,* pp. 285-295, Dec. 2006.

[18] F.I. Bashir, A.A. Khokhar, and D. Schonfeld, "Real-Time Motion Trajectory-Based Indexing and Retrieval of Video Sequences," *IEEE Trans. Multimedia,* vol. 9, no. 1, pp. 58-65, Jan. 2007.

[19] B. Georgescu, I. Shimshoni, and P. Meer, "Mean Shift Based Clustering in High Dimensions: A Texture Classification Example," *Proc. IEEE Int'l Conf. Computer Vision,* vol. 1, pp. 456-463, 2003.

[20] C.R. Jung, L. Hennemann, and S.R. Musse, "Event Detection Using Trajectory Clustering and 4-D Histograms," *IEEE Trans. Circuits and Systems for Video Technology,* vol. 18, no. 11, pp. 1565-1575, Nov. 2008.

[21] B.T. Morris and M.M. Trivedi, "A Survey of Vision-Based Trajectory Learning and Analysis for Surveillance," *IEEE Trans. Circuits and Systems for Video Technology,* vol. 18, no. 8, pp. 1114-1127, Aug. 2008.

[22] I. Saleemi, K. Shafique, and M. Shah, "Probabilistic Modeling of Scene Dynamics for Applications in Visual Surveillance," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 31, no. 8, pp. 1472-1485, Aug. 2009.

[23] B.T. Morris and M.M. Trivedi, "Learning, Modeling, and Classification of Vehicle Track Patterns from Live Video," *IEEE Trans. Intelligent Transportation Systems,* vol. 9, no. 3, pp. 425-437, Sept. 2008.

[24] N. Piotto, N. Conci, and F.G.B. De Natale, "Syntactic Matching of Trajectories for Ambient Intelligence Applications," *IEEE Trans. Multimedia,* vol. 11, no. 7, pp. 1266-1275, Nov. 2009.

[25] H. Veeraraghavan and N.P. Papanikolopoulos, "Learning to Recognize Video-Based Spatiotemporal Events," *IEEE Trans. Intelligent Transportation Systems,* vol. 10, no. 4, pp. 628-638, Dec. 2009.

[26] S. Dagtas and W. Al-Khatib, "Models for Motion-Based Video Indexing and Retrieval," *IEEE Trans. Image Processing,* vol. 9, no. 1, pp. 88-101, Jan. 2000.

[27] Y.K. Jung, K.W. Lee, and Y.S. Ho, "Content-Based Event Retrieval Using Semantic Scene Interpretation for Automated Traffic Surveillance," *IEEE Trans. Intelligent Transportation Systems,* vol. 2, no. 3, pp. 151-163, Sept. 2001.

[28] T. Pavlidis, "Polygonal Approximation by Newton's Method," *IEEE Trans. Computers,* vol. 26, no. 8, pp. 800-807, Aug. 1977.

[29] F.S. Cohen, Z. Huang, and Z. Yang, "Invariant Matching and Identification of Curves Using B-Splines Curve Representation," *IEEE Trans. Image Processing,* vol. 4, no. 1, pp. 1-10, Jan. 1995.

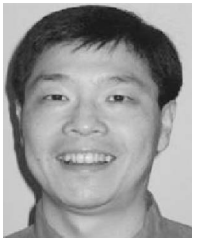
[30] N. Ansari and E.J. Delp, "On Detecting Dominant Points," *Pattern Recognition,* vol. 24, no. 5, pp. 441-451, 1991.
[31] Z. Gu, "Video Database Retrieval Based on Trajectory Analysis," master's thesis, Dept. of Computer Science, Univ. of British Columbia, 1997.
[32] C. Yajima, Y. Nakanishi, and K. Tanaka, "Querying Video Data by Spatio-Temporal Relationships of Moving Object Traces," *Proc. IFIP 2.6 Working Conf. Visual Database Systems,* pp. 357-371, May 2002.
[33] Y.W. Teh, "Dirichlet Process," *Encyclopedia of Machine Learning,* pp. 280-287, Springer, 2010.
[34] S. Chib, *Markov Chain Monte Carlo Technology,* pp. 89-92. Springer, 2004.
[35] C.-W. Su, H.-Y.M. Liao, H.-R. Tyan, C.-W. Lin, D.-Y. Chen, and K.-C. Fan, "Motion Flow-Based Video Retrieval," *IEEE Trans. Multimedia,* vol. 9, no. 6, pp. 1193-1201, Oct. 2007.
[36] A. Naftel and S. Khalid, "Classification and Prediction of Motion Trajectories Using Spatiotemporal Approximation," *Proc. Int'l Workshop Human Activity Recognition and Modeling,* pp. 17-26, 2005.
[37] E. Keogh, "Exact Indexing of Dynamic Time Warping," *Proc. Int'l Conf. Very Large Data Bases,* pp. 406-417, 2002.
[38] E.J. Keogh and M.J. Pazzani, "Scaling Up Dynamic Time Warping for Datamining Applications," *Proc. Int'l Conf. Knowledge Discovery and Data Mining,* pp. 285-289, 2000.
[39] E. Keogh, T. Palpanas, V.B. Zordan, D. Gunopulos, and M. Cardle, "Indexing Large Human-Motion Databases," *Proc. Int'l Conf. Very Large Data Bases,* pp. 780-791, 2004.
[40] L. Chen, M.T. Ozsu, and V. Oria, "Robust and Fast Similarity Search for Moving Object Trajectories," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 491-502, 2005.
[41] B.T. Morris and M.M. Trivedi, "Learning Trajectory Patterns by Clustering: Experimental Studies and Comparative Evaluation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 312-319, 2009.
[42] L. Chen and R. Ng, "On the Marriage of LP-Norms and Edit Distance," *Proc. Int'l Conf. Very Large Data Bases,* pp. 792-803, 2004.
[43] T. Palpanas, M. Vlachos, E. Keogh, D. Gunopulos, and W. Truppel, "Online Amnesic Approximation of Streaming Time Series," *Proc. IEEE Int'l Conf. Data Eng.,* pp. 338-349, Mar. 2004.
[44] B.T. Morris and M.M. Trivedi, "Trajectory Learning for Activity Understanding: Unsupervised, Multilevel, and Long-Term Adaptive Approach," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 33, no. 11, pp. 2287- 2301, Nov. 2011.
[45] X. Wang, H. Ding, G. Trajcevski, P. Scheuermann, and E.J. Keogh, "Experimental Comparison of Representation Methods and Distance Measures for Time Series Data," *Data Mining and Knowledge Discovery,* vol. 26, no. 2, pp. 275-309, Mar. 2013.

Weiming Hu received the PhD degree from the Department of Computer Science and Engineering, Zhejiang University, in 1998. From April 1998 to March 2000, he was a postdoctoral research fellow with the Institute of Computer Science and Technology, Peking University. Currently, he is a professor in the Institute of Automation, Chinese Academy of Sciences. His research interests include visual surveillance, and filtering of Internet objectionable information.

Xi Li received the doctoral degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2009. He is currently a senior research associate at the University of Adelaide, Australia. From 2009 to 2010, he worked as a postdoctoral researcher at CNRS Telecomd ParisTech, France.

Guodong Tian received the BS and MS degrees in information and communication engineering, from Beijing University of Posts and Telecommunications, Beijing, China, in 2007 and 2010, respectively. He is working toward the PhD degree in the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include activity recognition and motion trajectory learning.

Stephen Maybank received the BA degree in mathematics from King's College Cambridge in 1976 and the PhD degree in computer science from Birkbeck College, University of London, in 1988. Currently, he is a professor in the School of Computer Science and Information Systems, Birkbeck College. His research interests include the geometry of multiple images, camera calibration, visual surveillance, etc.



Zhongfei Zhang received the BS degree in electronics engineering, the MS degree in information science, both from Zhejiang University, China, and the PhD in computer science from the University of Massachusetts at Amherst. He is a professor of computer science at the State University of New York at Binghamton. His research interests include computer vision and multimedia processing, etc.