Content based Video Retrieval, Classification and Summarization: The State-of-the-Art and the Future

Xiang Ma, Xu Chen, Ashfaq Khokhar and Dan Schonfeld

Abstract This chapter provides an overview of different video content modeling, retrieval and classification techniques employed in existing content-based video indexing and retrieval (CBVIR) systems. Based on the modeling requirements of a CBVIR system, we analyze and categorize existing modeling approaches. Starting with a review of video content modeling and representation techniques, we study view-invariant representation approaches and the corresponding performance analysis. Based on the current status of research in CBVIR systems, we identify the video retrieval approaches from spatial and temporal analysis. Subsequently, we present the video classification approaches from multidimensional distributed Hidden Markov Models. Finally, a summary of future trends and open problems of content-based video modeling retrieval and classification is provided.

1 Video Content Modeling and Representation

In this section, we give a generalization of video content modelling and representation. We first investigate the general problems in video content modelling and representation. With regard to that, we present the state-of-the-art approaches: curvature scale space (CSS) and centroid distance functions (CDF)-based representations. We subsequently propose the null space invariant (NSI) representations for video classification and retrieval due to camera motions. Moreover, we propose the tensor null space invariant (TNSI) representation for high dimensional data. Finally, we give a brief overview of the other approaches in video content modelling and representation and future trends.

1.1 Definition

Motion content-based video collections are growing rapidly in both the professional and consumer environment, and are characterized by a steadily increasing capacity and content variety. Since searching manually through these collections is tedious and time-consuming, classification and retrieval tasks to automated systems becomes crucial for being able to efficiently handle stored video volumes. The development of such systems is based on the algorithms for video content analysis. These algorithms are built around the models bridging the gap between unifying query-by-example based indexing and retrieval systems and high-level semantic query-based activity recognition.

1.2 General Problems

Within the last several years, object motion trajectory-based recognition has gained significant interest in diverse application areas including sign language gesture recognition, Global Positioning System (GPS), Car Navigation System (CNS), animal mobility experiments, sports video trajectory analysis and automatic video surveillance. Psychological studies show that human beings can easily discriminate and recognize an object's motion pattern even from large viewing distances or poor visibility conditions where other features of the object vanish.
The development of accurate activity classification and recognition algorithms in multiple view situations is still an extremely challenging task. Object trajectories captured from different view-points lead to completely different representations, which can be modeled by affine transformation approximately. To get a view independent representation, the trajectory data is represented in an affine invariant feature space. With regard to that, a compact, robust view invariant representation due to camera motions is highly desirable.

1.3 View-Invariant Representation

View-Invariant representation is a very important and sometimes difficult aspect of an intelligent system. The representation is an abstraction of the sensory data, which should reflect a real world situation, be view-invariant and compact, and be reliable for later processing. The view invariant representation includes affine view invariant, scale view invariant and projective view invariant, etc. Once the representation has been defined, various classification or recognition algorithms can be performed. The methods usually involve some kind of distance calculation between a model and an unknown input (query). The model with smallest distance is taken to be the class of motion to which the input belongs. The problem with this is that the system can only recognize a predefined set of behaviors [1] [2] [3].

1.3.1 Curvature Scale Space (CSS) and Centroid Distance Functions (CDF)-based Representation

As described in [4], scale-space is a multi-resolution technique used to represent data of arbitrary dimension without any knowledge of noise level and preferred scale (smoothness). The notion of scale in measured data is handled by representing measured signal at multiple levels of detail, from the finest (original signal) to the coarsest (most-smoothed version). The CSS representation takes curvature data of a parametric curve and represents it by its different evolved versions at increasing levels of smoothness. The curvature $\kappa[k]$ for the trajectory represented as in (1) can be expressed as:

$$\kappa[k] = \frac{x'[k]y''[k] - y'[k]x''[k]}{\{x'[k]^2 + y'[k]^2\}^{3/2}} \tag{1}$$

The curvature of a trajectory has several desirable computational and perceptual characteristics. One such property is that it is invariant under planar rotation and translation of the curve. Curvature is computed from dot and cross products of parametric derivatives and these are purely local quantities, hence independent of rotations and translations. The dot and cross products are based only on the lengths, and angles between, vectors. Hence, these are also independent of rigid transformations of the coordinate system. Given a trajectory as in Eq.(1), the evolved version of the trajectory in terms of scale-space is defined by:

$$r_\sigma[k] = X[k,\sigma], Y[k,\sigma] \ , \tag{2}$$

where

$$X[k;\sigma] = x[k] \otimes g[k;\sigma], \tag{3}$$
$$Y[k;\sigma] = y[k] \otimes g[k;\sigma] \tag{4}$$

with $g[k;\sigma]$ being the symmetric Gaussian kernel used for smoothing. At each level of scale, governed by the increasing standard deviation of Gaussian kernel, curvature of the evolved trajectory is computed. Then the function implicitly defined by

$$\kappa[k;\sigma] = 0 \tag{5}$$

is the curvature scale space image of the trajectory. It is defined as a binary image with a value of 1 assigned to the points of curvature zero crossing at each scale level. Note that each of the arch-shaped contours in the CSS image corresponds to a convexity or concavity on the original trajectory with the size of the arch being proportional to the size of the corresponding feature. The CSS image is a very robust representation under the presence of noise in trajectory data due to small camera motions and minor jitters in tracking. Noise amplifies only the small peaks, with no effect on the location or scale of the feature contour maxima. As evident from the figure, the major peaks of the CSS image remain quite preserved after significant

affine transformation.
The centroid distance function is another invariant representation of the raw shape data used in the affine invariant image retrieval applications. The centroid distance function is expressed by the distance of each point in trajectory from the centroid of the trajectory:

$$c[k] = \sqrt{[x[k]-x_c]^2 + [y[k]-y_c]^2}, k = 0,1,\ldots,N-1 \tag{6}$$

where $x_c = \frac{1}{N}\sum_{t=o}^{N-1} x[k]$,$y_c = \frac{1}{N}\sum_{t=o}^{N-1} y[k]$. Let us denote by uniform affine transformation the set of affine transforms including translation, rotation and uniform scaling. This excludes the shear transformation in general affine transformation. Let us denote the centroid distance function of a trajectory before affine transformation as $C[k]$ and after uniform affine transformation as $C^{'}[k]$. Then it can be easily proved that under uniform affine transformation, the following relation holds between the centroid distance functions computed from original and affined version of the trajectory:

$$C'[k] = \alpha C[k],$$
$$\alpha = 1, for \begin{pmatrix} u[k] \\ v[k] \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x[k] \\ y[k] \end{pmatrix} + \begin{pmatrix} \beta \\ \gamma \end{pmatrix}$$
$$\alpha = \alpha_s, for \begin{pmatrix} u[k] \\ v[k] \end{pmatrix} = \alpha_s \begin{pmatrix} x[k] \\ y[k] \end{pmatrix}. \tag{7}$$

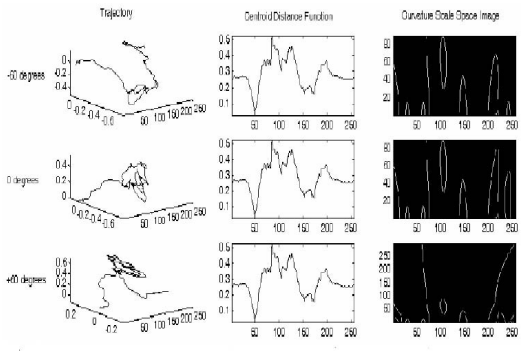

Fig. 1 Trajectory of the hand motion for signing the word 'Alive' in Australian Sign Language with its two rotated versions, and their corresponding representations using centroid distance functions (CDFs) and curvature scale-space (CSS) images.

Figure 1 displays one of the trajectories from ASL dataset along with two of its rotated versions, and its representation in terms of the two feature spaces explored

in this presentation. As seen from this figure, the CDF-based representation is absolutely invariant to rotational deformations. The CSS-based representation, on the other hand, results in the curvature zero crossings being consistent but the CSS maxima tend to shift around. In the next paragraph, we outline the procedure for HMM training and classification based on these to invariant feature spaces.
In the case of CSS representation, as outlined previously, the trajectory is represented by the locations of CSS maxima in terms of their temporal ordering and the scale of concavity. In this context, the trajectory data is represented by a time-ordered sequence of two-dimensional feature vectors containing CSS maxima.
We report the results on a range of dataset sizes from the ASL [5] dataset. The

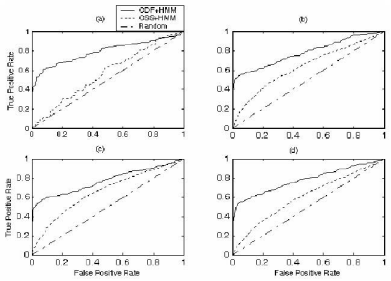

Fig. 2 ROC curves for the rotated trajectories posed for classification.

results are reported in terms of the ROC curves in Figure 2. The ROC curves are two-dimensional depiction of classifier performance.

1.3.2 Null Space View Invariance Representation (NSI)

In this subsection, we introduce a simple but highly efficient view invariant representation based on Null Space Invariant (NSI) matrix. As far as we know, this is the first use of Null space in motion-based classification/retrieval applications.
Indexing and classification of the NSI operator is obtained by extracting features of the null space representation using Principal Components Null Space Analysis (PCNSA), which provides an efficient analysis tool when different classes may have different non-white noise covariance matrices. Dimensionality reduction for indexing of the NSI is achieved by first performing Principal Components Analysis (PCA) as part of PCNSA. Classification is performed in PCNSA by determining the *ith* class M_i-dimensional subspace by choosing the M_i eigenvectors that give the smallest intra-class variance. The M_i-dimensional space is referred to as the Approximate Null Space (ANS). A query is classified into the class if its distance to the class mean in ANS space is lowest among all the other classes. A fundamental set of 2-D

affine invariants for an ordered set of n points in R^2 (not all collinear) is expressed as an n-3 dimensional subspace, H^{n-3}, of R^{n-1}, which yields a point in the 2n-6 dimensional Grassmannian Gr_R(n-3,n-1), which also shows number of invariants is 2n-6 in 2-D.

Null Space Invariant (NSI) of a trajectories matrix (each row in the matrix corresponds to the positions of a single object over time) is introduced as a new and powerful affine invariant space to be used for trajectory representation. This invariant, which is a linear subspace of a particular vector space, is the most natural invariant and is definitely more general and more robust than the familiar numerical invariants. It does not need any assumptions and after invariant calculations it conserves all the information of original raw data.

Let $Q_i = (x_i, y_i)$ be a single 2-D point, $i = 0, 1, \ldots, n-1$, among n ordered non-linear points in R^2, representing a trajectory. Consider the following arrangement of the n 2-D points in a $3 \times n$ matrix M:

$$M = \begin{pmatrix} x_0 & x_1 & \ldots & x_{n-1} \\ y_0 & y_1 & \ldots & y_{n-1} \\ 1 & 1 & \ldots & 1 \end{pmatrix} \tag{8}$$

(n-3)-dimensional linear subspace H^{n-3} can be associated to a 2-D trajectory whose features set is $Q_0, Q_1, \ldots, Q_{n-1}$:

$$H^{n-3} = \{q = (q_0, q_1, \ldots, q_{n-1})^T, i.e. Mq = (0,0,0)^T\} \tag{9}$$

Since at least one determinant of 3×3 minor of M is not zero because of non-linear feature points, H^{n-3} has a dimension of n-3. The attractive property of the linear subspace is that it does not change when it undergoes any of the affine transformations. We use this new invariant of the trajectory matrix M to represent each trajectory. Moreover,

$$H^{n-3} \subset R^{n-1} = \{q = (q_0, q_1, \ldots, q_{n-1})^T \in R^{n-1}, and \sum_{i=0}^{n-1} q_i = 0\} \tag{10}$$

which produces (n-3)-planes in (n-1)-space, $Gr_R(n-3, n-1)$. $Gr_R(n-3, n-1)$ is a well understood manifold of dimension 2n-6, which is the number of invariants associated to the matrix M. H^{n-3} is spanned by the vectors $v_i = (q_0^i, q_1^i, \ldots, q_{n-1}^i)^T, i = 3, 4, \ldots, n-1$, where

$$q_0^i = -det\begin{pmatrix} x_1 & x_2 & x_i \\ y_1 & y_2 & y_i \\ 1 & 1 & 1 \end{pmatrix} / det\begin{pmatrix} x_0 & x_1 & x_2 \\ y_0 & y_1 & y_2 \\ 1 & 1 & 1 \end{pmatrix} \tag{11}$$

$$q_1^i = det\begin{pmatrix} x_0 & x_2 & x_i \\ y_0 & y_2 & y_i \\ 1 & 1 & 1 \end{pmatrix} / det\begin{pmatrix} x_0 & x_1 & x_2 \\ y_0 & y_1 & y_2 \\ 1 & 1 & 1 \end{pmatrix} \tag{12}$$

$$q_2^i = -det\begin{pmatrix} x_0 & x_1 & x_i \\ y_0 & y_1 & y_i \\ 1 & 1 & 1 \end{pmatrix} / det\begin{pmatrix} x_0 & x_1 & x_2 \\ y_0 & y_1 & y_2 \\ 1 & 1 & 1 \end{pmatrix} \tag{13}$$

$$q_i^i = -1 and q_i^j = 0 for j = 3,4,\ldots,i-1,i+1,\ldots,n-1 \tag{14}$$

This is an elegant and simple method to find the null space of a trajectory matrix M, which is also the most promising invariant of a trajectory. It is important to note that not every basis representation of the null space can be used to form an invariant matrix for a trajectory. For example, using the singular value decomposition (SVD) to form an orthogonal basis of the null space would not guarantee an affine invariant representation. Now each trajectory is represented now with a matrix $NSI_{n\times(n-3)}$, which is composed of v_i columns, from the trajectory matrix $M_{3\times n}$. While is an affine invariant matrix, this representation increases the dimensionality significantly, from 2n to $n\times(n-3)$. We reduce the dimensionality to a comparably small but effective value such as n by using PCA.

Once we have generated the affine invariant representation provided by the null space operator, $NSI_{n\times(n-3)}$, we can rely on numerous methods for indexing and classification. We choose a method for dimensionality-reduction and classification based on PCNSA [6]. Perturbation analysis is an important mathematical method which is widely used for analyzing the sensitivity of linear systems by adding a small term to the mathematical description of the exactly solvable problem. As described in [7], based on the perturbed null operator, it is desirable to know the ratio of the input error and the output error where the input error is referred to the error of the trajectory matrix and the output error is referred to the error of the null operator. Therefore, the ratio of the output error and input error is:

$$\tau = \frac{E\|Q-\tilde{Q}\|_F^2}{E\|Z\|_F^2} = \frac{1}{N}[(N-3)(r_{01}^2+r_{02}^2+r_{12}^2)+\sum_{i=3}^{N-1}(r_{0i}^2+r_{1i}^2+r_{2i}^2)]\,. \tag{15}$$

It can be seen that the ratio only relies on the trajectory itself while independent of the noise.

Besides the error ratio, SNR is widely used to evaluate robustness of the system. Let us derive the expression for the output SNR. Defining the power of the output signal as $\|Q\|_F^2$, SNR can be computed by $\Delta_{SNR} = \frac{\|Q\|_F^2}{E\|Q-\tilde{Q}\|_F^2}$ as:

$$\Delta_{SNR} = \{A\sum_{i=3}^{N-1} y_i^2 + B\sum_{i=3}^{N-1} x_i^2 + C\sum_{i=3}^{N-1} x_i y_i + D\sum_{i=3}^{N-1} x_i + E\sum_{i=3}^{N-1} y_i + F\}/\{2\delta^2[(N-3)\sum_{j,k=0}^{2} r_{jk}^2 + \sum_{i=3}^{N-1}(r_{0i}^2 + r_{1i}^2 + r_{2i}^2)]\}\ , \tag{16}$$

where

$$A = \sum_{j,k=0}^{2}(x_j - x_k)^2, B = \sum_{j,k=0}^{2}(y_j - y_k)^2\ , C = -2\sum_{j,k=0}^{2}(x_j - x_k)(y_j - y_k)\ , \tag{17}$$

$$D = 2\sum_{j,k=0}^{2}(y_j - y_k)(x_j y_k - x_k y_j)\ , E = 2\sum_{j,k=0}^{2}(x_j - x_k)(x_j y_k - x_k y_j)\ , \tag{18}$$

$$F = (N-3)[\sum_{j,k=0}^{2}(x_j y_k - x_k y_j)^2 + (x_1 y_2 - x_2 y_1 + x_0 y_1 - x_1 y_0 - x_0 y_2 + x_2 y_0)^2]\ , \tag{19}$$

where $j \neq k$. It can be seen that the critical points of SNR are trajectory-dependent. Using basic perturbation theorem, we obtain:

$$\frac{\|\tilde{Q} - Q\|}{\|Q\|} \leq \|M^{-1}Z\|\ , \tag{20}$$

where $\|.\|$ denotes a matrix norm and a consistent vector norm, referred as normwise bounds. If we weaken the bound, we can relate it with the condition number of M.

$$\frac{\|\tilde{Q} - Q\|}{\|Q\|} \leq \|M^{-1}\|\|Z\| = \kappa(M)\frac{\|Z\|}{\|M\|}\ , \tag{21}$$

where the condition number $\kappa(M) = \|M\|\|M^{-1}\|$. In addition, if $\|M^{-1}Z\| < 1$, it is easy to obtain:

$$\frac{\|\tilde{Q} - Q\|}{\|Q\|} \leq \frac{\|M^{-1}Z\|}{1 - \|M^{-1}Z\|}\ . \tag{22}$$

If we are willing to compute the inverse of M:

$$|\tilde{Q} - Q| \leq |M^{-1}\|Z\|\tilde{Q}|\ . \tag{23}$$

Moreover, if for some consistent matrix norm $\||M^{-1}||Z|\| < 1$, then $(I - |M^{-1}||Z|)^{-1}$ is nonnegative and

$$|\tilde{Q} - Q| \leq (I - |M^{-1}||Z|)^{-1}|M^{-1}||Z||Q|\ . \tag{24}$$

Given the perturbation, designing optimal sampling strategy is very important for the robustness of the system. Expanding arbitrary trajectories in x and y directions in Maclaurin series, we obtain:

$$x = f(t) = f(0) + f^{'}(0)t + \frac{f^{''}(0)}{2!}t^2 + ... + \frac{f^{(n)}(0)}{n!}t^n \ . \tag{25}$$

$$y = g(t) = g(0) + g^{'}(0)t + \frac{g^{''}(0)}{2!}t^2 + ... + \frac{g^{(n)}(0)}{n!}t^n \ . \tag{26}$$

The distance between two arbitrary samples can be computed as:

$$\begin{aligned} r_{kj}^2 &= (x_k - x_j)^2 + (y_k - y_j)^2 \\ &= [f^{'}(0)(t_j - t_k) + ... + \frac{f^{(n)}(0)}{n!}(t_j^n - t_k^n)]^2 + \\ &\quad [g^{'}(0)(t_j - t_k) + ... + \frac{g^{(n)}(0)}{n!}(t_j^n - t_k^n)]^2 \ . \end{aligned} \tag{27}$$

Therefore, τ is equal to $O(N^{2n})$ which increases dramatically with the number of the samples. With regard to SNR, we have the following property:
Property 1: With uniform sampling $t_k = kT$,

$$\lim_{N\to\infty} \Delta_{SNR} = \frac{A(g^{(n)}(0))^2 + B(f^{(n)}(0))^2}{6\delta^2[(g^{(n)}(0))^2 + (f^{(n)}(0))^2]} + \frac{Cf^{(n)}(0)g^{(n)}(0)}{6\delta^2[(g^{(n)}(0))^2 + (f^{(n)}(0))^2]} \tag{28}$$

where A, B and C are defined in the expression of SNR.
Property 2: $\lambda = O(N)$ should be chosen for Poisson sampling to guarantee the convergence of τ, where N is the total number of samples.
Property 3: τ converges in mean sense given $\lambda = O(N)$. Specifically, for $\lambda = \frac{N}{T}$,

$$\lim_{N\to\infty} E(\tau) = 3\sum_{k=1}^{2n} \frac{(a_k + b_k)T^k}{k+1} \ , \tag{29}$$

where k is the index for Taylor series and for a_k and b_k, if k is odd,

$$a_k = \sum_{i=1}^{\frac{k-1}{2}} \frac{2g^{(i)}(0)g^{(k-i)}(0)}{i!(k-i)!} \ , \tag{30}$$

$$b_k = \sum_{i=1}^{\frac{k-1}{2}} \frac{2f^{(i)}(0)f^{(k-i)}(0)}{i!(k-i)!} \ , \tag{31}$$

if k is even,

$$a_k = \sum_{i=1}^{\frac{k-2}{2}} \frac{2g^{(i)}(0)g^{(k-i)}(0)}{i!(k-i)!} + \frac{(g^{(\frac{k}{2})}(0))^2}{(\frac{k}{2}!)^2} \ , \tag{32}$$

$$b_k = \sum_{i=1}^{\frac{k-2}{2}} \frac{2f^{(i)}(0)f^{(k-i)}(0)}{i!(k-i)!} + \frac{(f^{(\frac{k}{2})}(0))^2}{(\frac{k}{2}!)^2} \ . \tag{33}$$

Property 4: If the trajectories are sampled with $\lambda = O(N)$, the variance of the error ratio converges to zero, namely,

$$\lim_{N\to\infty} Var(\tau) = 0\ . \tag{34}$$

Remark: Property 3 can be proved by showing that

$$\lim_{N\to+\infty} \frac{1}{N}\sum_{j=3}^{N-1} E(t_j^k) = \frac{1}{k+1}\ . \tag{35}$$

In our framework, the density λ corresponds to the average number of samples per unit-length; i.e. $\lambda = \frac{N}{T}$.

Since in real life trajectories in a class may have different lengths, we normalize

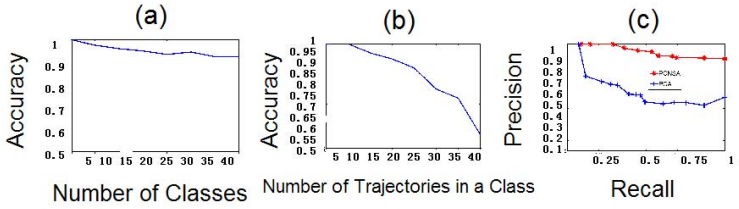

Fig. 3 (a) Accuracy values for the classification problem on increasing number of classes, (b) Accuracy values for the classification problem on increasing number of trajectories in a class. (c) The comparison with Precision-Recall Curve for retrieval with 20 classes for perfect trajectories.

the length by taking the Fourier Transform and choosing the biggest n=32 coefficients and then taking the Inverse Fourier Transform so that all the trajectories are of size 32 before invariant matrix calculations. The invariant matrix, for each trajectory is certainly robust to affine transformation such that it always preserves its values against rotation or translation operations. For all the simulations $\delta_1 = 10^7$, $\delta_2 = 10^{-4}$ as thresholds and $L = 32$ in PCNSA. Although we did not show here, we note that increasing L gives better results.

Figure.3(a) depicts accuracy of the proposed classification system versus number of classes. There are $K = 20$ trajectories in each class word. Simulation results show that our system preserves its efficiency even for higher number of different classes.

Figure.3(b) depicts accuracy values versus increase in the number of trajectories within a class. There are $C = 20$ classes in the system. Simulation results show that our system performance deteriorates slightly for high number of trajectories in a class. This problem can be resolved by using a hierarchical representation, where we first separate all the trajectories in the system into a small number of classes, then repeatedly divide each class into smaller classes until each class has a sufficiently small number of trajectories. We can now use the null space representation for each class in the nested hierarchical tree structure to obtain a scalable repre-

sentation whose performance is robust as the number of trajectories in the system increases. Figure.3(c) shows Precision vs. Recall curves for indexing and retrieval problem by using 40 classes, each class having 20 trajectories. For retrieval problems, we compute the distance of the query trajectory to any other trajectory using PCNSA on NSI as $D(X_i,Y) = \|W_{NSA,i}(X_i - Y)\|$, where Y is the query trajectory. This distance is then used to find α nearest trajectories, where α is a user specified parameter. There are two curves in Figure 3(c), one is with using PCA on NSI directly, where PCA is basically used for dimension reduction. As it can be seen from Figure 3(c) that the result of using PCNSA on NSI is much superior to the one using PCA on NSI directly. The visual illustration of the query and the three most similar retrieval are shown in Fig. 4.

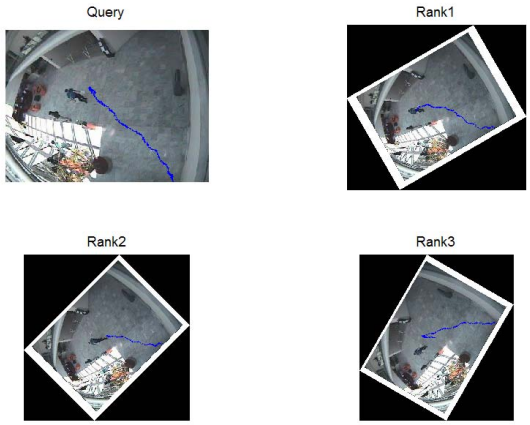

Fig. 4 Visual illustration for retrieval results with 20 classes with motion trajectories.

1.3.3 Tensor Null Space Invariance Representations (TNSI)

Among affine view-invariance systems, majority of them represent affine view-invariance in a single dimension, thus limiting the system to only single object motion based queries and single dimension affine view-invariance. In many applications, it is not only the individual movement of an object that is of interest, but also the motion patterns that emerge while considering synchronized or overlapped movements of multiple objects. For example, in sports video analysis, we are often interested in a group activity involving motion activity of multiple players, rather than the activity of each individual player. Moreover, due to camera movement, same motion trajectory has completely different representations from different viewing angles. Therefore, a highly efficient classification and retrieval system which is invariant to multidimensional affine transformations is highly desirable. In

this subsection, we propose a novel fundamental mathematical framework for tensor null space invariants and use this framework for the important application of view-invariant classification and retrieval of motion events involving multiple motion trajectories.

Let us denote the tensor $A \in R^{I_1 \times I_2 \ldots I_{N-1} \times I_N}$ as the multi-dimensional data. Elements of A are denoted as $a_{i_1 i_2 \ldots i_N}$. A generalization of the product of two matrices is the product of a tensor and a matrix. The *mode-n product* of a tensor $A \in R^{I_1 \times I_2 \ldots \times I_n \ldots I_{N-1} \times I_N}$ by a matrix $U \in R^{I_n \times J_n}$, denoted by $A \times_n U$, is a tensor $B \in R^{I_1 \times \ldots I_{n-1} \times J_n \times I_{n+1} \ldots \times I_N}$ whose entries are:

$$(A \times_n U)_{i_1 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} = \sum_{i_n} a_{i_1 \ldots i_{n-1} i_n i_{n+1} \ldots i_N} u_{i_n j_n} \tag{36}$$

The mode-n product $B = A \times_n U$ can be computed via the matrix multiplication $B_{(n)} = UA_{(n)}$, followed by a re-tensorization to undo the mode-n flattening.

As described in the equations (8)-(10), applying the affine transformation T on the matrix $M_1 = TM$, the null spaces of M_1 and M are identical. Similarly, applying the affine transformation T_m, T_n on the mth, nth unfolding of the multi-dimensional data M, respectively, if the resulting tensor null space Q is invariant in both dimensions, the it is referred to as *mode-m,n invariant*. Let us derive the mathematical formulation of the mode-1,2,3 invariant tensor Q for three dimensional data $M \in R^{I_1 \times I_2 \times I_3}$. To be rotation invariant, we have:

$$M_{(1)} \times Q_{(3)} = 0 \ , M_{(2)} \times Q_{(2)} = 0 \ , M_{(3)} \times Q_{(1)} = 0 \ , \tag{37}$$

where $M_{(1)}, M_{(2)}, M_{(3)}$ are the unfolding of the three order tensor M into matrices with the dimension $I_2 I_3 \times I_1$, $I_1 \times I_3 I_2$ and $I_1 \times I_2 I_3$, respectively, and $Q_{(3)}, Q_{(2)}, Q_{(1)}$ are the corresponding unfoldings of the tensor Q. The condition for invariance of translation for the mode-1,2,3 invariant tensor Q:

$$\sum Q_{(1)} = 0, \sum Q_{(2)} = 0, \sum Q_{(3)} = 0 \tag{38}$$

Combining the conditions for rotational and translational invariance, we can solve the TNSI Q subject to the mode-1,2,3 affine view-invariant. If the order of the tensor M is 2, it is easy to show that the condition boils down to the Stiller's one dimensional null space invariants [6]. It is also easy to extend the result to the case of the Nth order tensor with mode-$I_a, \ldots, I_k$ affine view-invariant.

We align each trajectory as two rows in a matrix according to x and y coordinates, and the number of rows of a matrix is set to be twice the number of the objects in the motion event under analysis.

$$M = (M_{i,j})_{i=1,2,\ldots,2J; j=1,2,\ldots,P} \ , \tag{39}$$

In the above equation, P denotes the temporal length of normalized trajectories, J represents the number of trajectories within one motion event. Finally, multiple trajectory matrices are aligned in the direction orthogonal to the plane spanned by

them, and form a three dimensional matrix, or tensor. We refer to it as *Motion Event Tensor T*. as shown in Fig. 5.

$$T = (T_{i,j,k})_{i=1,2,\ldots,2J;j=1,2,\ldots,P;k=1,2,\ldots,K} \ , \tag{40}$$

where K is the number of motion event samples (trajectory video clips).

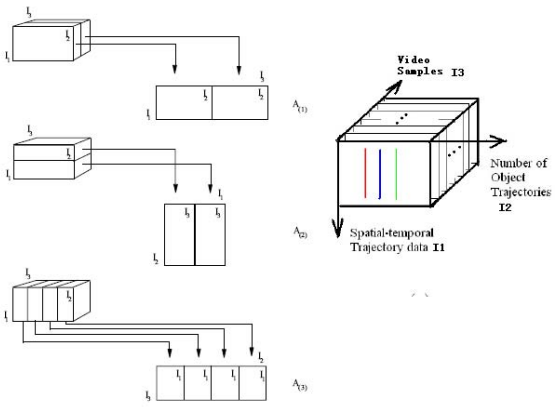

Fig. 5 Flattening a 3^{rd} order motion event tensor for multiple trajectories representation.

1.4 Other Representations

In other literature, Zhang et al [8] use multiple different criteria like color content change and zoom-in type of effects in shot content. A technique for shot content representation and similarity measure using subshot extraction and representation is presented in [9]. They use two content descriptors, Dominant Color Histogram (DCH) and Spatial Structure Histogram (SSH), to measure content variation and to represent subshots. Delp et al [10]represent a shot using a tree structure called shot tree, formed by clustering frames in a shot. This approach unifies the problem of scene content representation for both browsing and similarity matching where for browsing only the root node of the tree (keyframe) is used, while for similarity matching two or three levels of tree can be used employing the standard tree matching algorithms. In addition, there are other view invariant representations such as geometry invariants (GI) [11].

1.5 Open Problems and Future Trends

From the above review, it can be seen that that there have been advances in the field of video content modelling and representation, semantic level of visual information representation. However, there are still many open research issues that need to be addressed to make the full use out of visual information retrieval systems. In the following, we identify several open problems and provide a brief summary of future trends.

1. Segmentation of Tensor Null Space Invariants: The dimensionality of feature vectors employed in most systems for representing visual media can be solved by tensor null space invariants (TNSI) [12], but as suggested, the computational burden for TNSI is very heavy. There is a need in practical view invariant systems as to segment the space properly and reduce the dimension of the respresentation.
2. Optimal Sampling Strategy for High Dimensional Representation: In our work [7], we have proposed the optimal sampling scheme for 2D null space representation. Future work will consider the optimal sampling strategy for 3D null space representation.

2 Video Indexing and Retrieval

In this section, we present the concepts, problems and state-of-the-art approaches for content-based video indexing and retrieval (CBVIR). We first overview the general problems of content-based retrieval. Following that, we investigate one of the key problems in video retrieval-video summarization, where we summarize the state-of-the-art approaches for both shot-boundary detection and key frame extractions. Spatial and Temporal Analysis is very important to access video content, we focus on spatial-temporal motion trajectory analysis, and present both single and multiple motion trajectory-based CBVIR techniques, especially, (i) geometrical multiple-trajectory indexing and retrieval (GMIR) algorithm, (ii) unfolded multiple-trajectory indexing and retrieval (UMIR) algorithm, and (iii) concentrated multiple-trajectory indexing and retrieval (CMIR) algorithm. The above algorithms not only remarkably reduce the dimensionality of the indexing space but also enable the realization of fast retrieval systems. Finally, we give a brief overview of other approaches in video retrieval and future trends in this research area.

2.1 Definitions

By definition, a Content-Based Video Indexing and Retrieval (CBVIR) system aims at assisting a human operator (user) to retrieve sequence (target) within a potentially large database [13]. It is a natural extension of the well-known Content-Based Image

Indexing and Retrieval (CBIR) systems. Both systems are aiming at accessing image and video by its content, namely, the spatial (image) and spatial-temporal (video) information. Typical spatial information includes color, texture, edge, etc., while typical temporal information includes motion and change of scenes. Moving from images to video adds several orders of complexity to the retrieval problem due to indexing, analysis and browsing over the inherently temporal aspect of video [14].

2.2 General Problems

One of the key issues in CBVIR is, as pointed out in [14], to bridge the ”semantic gap”, which refers to the gap between low level features (such as color, texture, shape, structure, layout and motion) and high level semantic meanings of content (such as indoor and outdoor, people, or car-rasing scenes). Low level features such as color and textures are easy to measure and compute, however, it is a paramount challenge to connect the low level features to a semantic meaning, especially involving intellectual and emotional aspects of the human operator (user). Another issue is how to efficiently access the rich content of video information, these involves video content summarization, and spatial and temporal analysis of videos, which is discussed in details below.

2.3 Video Summarization

Video summarization is the process of creating a presentation of visual information about the structure of video, which should be shorter than the original video [15]. Typically, a subset of video data such as keyframes or highlights such as entries for shots, scenes, or stories is extracted for compact representation and fast browsing of video content. A *shot* is a set of contiguous frames acquired through a continuous camera recording, and is regarded as the fundamental building block of a video. Several shots consist a *scene*, which is a set of contiguous shots that have a common semantic significance. Finally, a video usually consists of several scenes. Given the significance of shots in videos, extraction of shots, or shot boundary detection is a key step towards video summarization.

2.3.1 Shot boundary Detection and Video Segmentation

Shot boundary detection, or video segmentation, is to identify the frame(s) of videos where a transition takes place from one shot to another [16]. The locations of these changes occur are referred to as a cut or a break. Examples of transitions include cuts, fades, dissolves, wipes and other special effect edits. Some efficient solu-

tions have been proposed for shot boundary detection, for details, we refer to [17] [18][19].

2.3.2 Key Frame Extraction

After video segmentation, key frames are extracted from each shot as representative features for compact summarization of video. Key frame extraction need to be automatic and content-based, such that important video shot content are retained while redundancies are removed after extraction. Several key frame extraction criteria have been proposed, such as heuristic decisions [20], to use intra frame or first frame of video shot as key frame; or visual features [21], such as brightness, color or dominant colors; or motion patterns [22][23], such as dominant motion components of camera or large moving objects. Those key frames are further utilized to summarize video content for efficient indexing and retrieval.

2.4 Spatial and Temporal Analysis

2.4.1 Motion Trajectory Analysis

Object motion has been an important feature for object representation and activity modelling in video applications [24] [25] [26] [27].

An object motion trajectory is usually represented by a two-dimensional(or three-dimensional) L-tuples corresponding to the x- and y-axes in a two-dimensional coordinate system (or x-, y- and z- axes in three-dimensional coordinate system) of the object centroid location at each instant of time, where L is the temporal duration.

$$r^L = \{x[t], y[t]\}, t = 1, ..., L. \tag{41}$$

An object trajectory-based system for video indexing is proposed in [28], in which the normalized x- and y-projections of trajectory are separately processed by wavelet transform using Haar wavelets. Chen et al. [29] segment each trajectory into subtrajectories using fine-scale wavelet coefficients at high levels of decomposition. A feature vector is then extracted from each subtrajectory and Euclidean distances between each subtrajectory in the query trajectory and all the indexed subtrajectories are computed to generate a list of similar trajectories in the database. Bashir et al [30] [3] proposed a Principle Component Analysis (PCA)-based approach to object motion trajectory indexing and retrieval, which has been shown to provide a very effective method for indexing and retrieval of single object motion trajectory.

2.4.2 Multiple Motion Trajectory Indexing and Retrieval

For a video clip consisting of multiple moving objects, multiple object motion trajectories are usually extracted by using some tracking algorithms [31] [32] or motion sensor devices. Fig. 6 depicted three video clips (a) and their corresponding multiple motion trajectories (b). Typically for a video clip of length L with M moving objects, the multiple motion trajectories are represented as a set $\mathscr{S}$ of M motion trajectories

$$\mathscr{S} = \{r_1^L, r_2^L, r_3^L, ..., r_M^L\}. \tag{42}$$

For a video database consisting of many video clips, each video clip has an unique

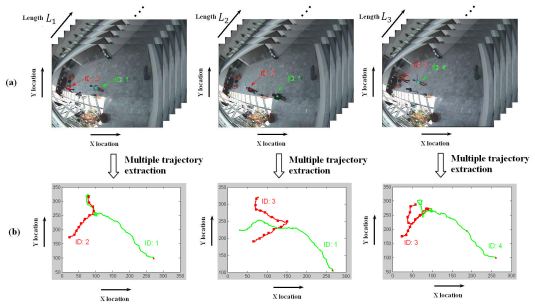

Fig. 6 Video clips and their corresponding multiple motion trajectories: (a)video clips 1, 2, 3; (b) their corresponding multiple motion trajectories.

corresponding set of multiple motion trajectories, which characterizing the dynamics and motion pattern of multiple objects within that particular video clip. Thus by indexing and retrieving a set of multiple object trajectories, we are able to index and retrieve its corresponding video clip in the database.

In our approach, each set of multiple object trajectories $\mathscr{S} = \{r_1^L, r_2^L, r_3^L, ..., r_M^L\}$, extracted from a particular video clip of length L with M objects, is modelled as a *Multiple Trajectory Matrix* $\mathscr{M}$ of size $2L$ by M.

We firstly smooth out each of the noisy trajectories by applying wavelet transform using Daubechies wavelet DB4, and taking coarse coefficients corresponding to the low subband in a three level decomposition. After that, we concatenate $x-$ and $y-$ location information of each object trajectory $r_i^L (i = 1, 2, ..., M)$, into one column vector, we refer to as *Single Trajectory Vector* $\mathscr{V}_i$.

$$\mathscr{V}_i = (x_i[1], x_i[2], ..., x_i[L], y_i[1], y_i[2], ..., y_i[L])^T \tag{43}$$

We then align the single trajectory vectors as columns of a matrix, where the number of columns is set to be the number of objects in the particular set of multiple object trajectories. We name this matrix as *Multiple Trajectory Matrix* $\mathscr{M}$.

$$\mathscr{M} = [\mathscr{V}_1 | \mathscr{V}_2 | \mathscr{V}_3 | ... | \mathscr{V}_M] \tag{44}$$

Here, each multiple trajectory matrix represents the motion dynamics of a group of objects within one particular video clip, please note that the order of single trajectory vectors placed in the multiple trajectory matrix is very important, for simplicity, we focused exclusively on the simplest case, where orders of multiple trajectories in both the dataset and the query are known. However, our tensor-space representation can be easily extended to the case where correspondense of multiple trajectories between the dataset and the query is unknown. For more details, please refer to [33].

In a motion video database with many video clips, multiple trajectories are firstly extracted for each video, then multiple trajectory matrix is constructed. For compact representation, multiple trajectory matrices of the same number of columns (video clips with the same number of acting objects), are grouped together. Then each group of video clips are resampled to a median size for further processing. The median size is the median of lengths from all video clips within the same group. Let there be N sets of M trajectories extracted from N video clips, and the original lengths of each set of M trajectories be L_1, L_2,...,L_N. Suppose the desired median length after sampling is L'. For each set of M trajectories, we resample the whole set of trajectories to the median length L'. For each trajectory within the same set, we first use 2D fourier transform to get coefficients of each trajectory in frequency domain; then we retain the first p biggest coefficients while ignoring the rest; finally we perform L'-point inverse 2D fourier transform to get the re-sampled trajectory with desired length L'. Fig. 7 (b) depicts three sets of 2-trajectories S_1, S_2 and S_3, extracted from three video clips of lengths L_1,L_2 and L_3, respectively; while Fig. 7 (c) shows the resampled multiple trajectory sets with the same median size L'. The re-sampling process is a necessary step to transform different sets of trajectories with varying lengths to the same format, such that they can be assembled into a compact form (e.g. matrix or tensor form) for further efficient analysis.

After resampling, within each group, multiple trajectory matrices are of the same size, then they are aligned in the direction orthogonal to the plane spanned by them, and form a three-dimensional matrix, or tensor [34]. We refer to it as *Multiple Trajectory Tensor* $\mathscr{T}$

$$\mathscr{T} = (T_{i,j,k})_{i=1,2,\ldots,2L';j=1,2,\ldots,M;k=1,2,\ldots,K.} \tag{45}$$

where L and M are the same as previous defined, K is the depth of the tensor, referring to total number of video clips indexed so far.

Fig. 7 depicts an example of constructing a multiple trajectory tensor by using three set of multiple trajectories. The reason to align multiple trajectory matrices and form a three-dimensional tensor is, to assemble as much as possible date into a compact form and extract intrinsic and common characteristics of data for efficient analysis. By assembling multiple trajectory into three-dimensional tensors, the multiple trajectory data spans a three-dimensional tensor-space.

Please note that our approach is built on trajectory representation by instantaneous x- and y- coordinates of object centroid at each frame. While processing trajectories, it is assumed that the frame of reference for all of the trajectories is held

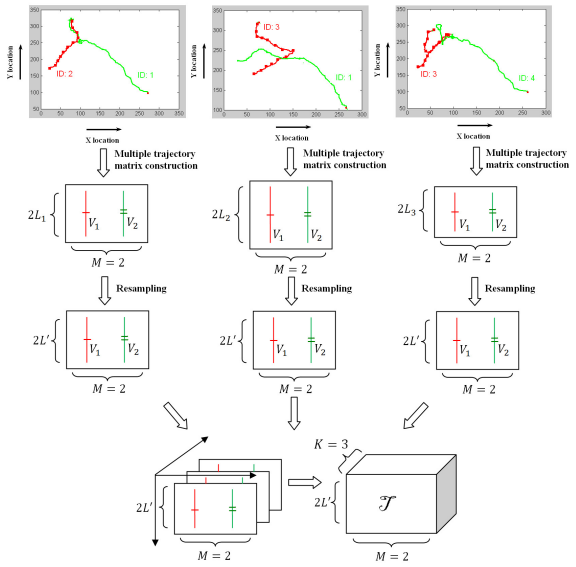

Fig. 7 Tensor-Space Representation of Multiple-Object Trajectories: (a) Three sets of multiple trajectories $\mathscr{S}_1$,$\mathscr{S}_2$,$\mathscr{S}_3$, extracted from three video clips displayed in Fig. 1. (b) Three corresponding *Multiple Trajectory Matrices* $\mathscr{M}_1$, $\mathscr{M}_2$, $\mathscr{M}_3$. (c) Three *Multiple Trajectory Matrices* $\mathscr{M}'_1$, $\mathscr{M}'_2$, $\mathscr{M}'_3$ after resampling. (d) *Multiple Trajectory Tensor* $\mathscr{T}$ constructed from $\mathscr{M}'_1$, $\mathscr{M}'_2$, $\mathscr{M}'_3$.

fixed during the generation of trajectories. This is consistent with the fixed-camera scenario. For the moving camera case, such as PTZ cameras or airborne surveillance, an additional step of trajectory registration will be needed to generate trajectories which are all registered to a common frame of reference. Our algorithms can then be used for indexing and retrieval of the registered trajectories from multiple objects for the moving camera scenario.

2.4.3 Global and Segmented Multiple Trajectory Tensor

We present two types of multiple trajectory tensors based on two types of multiple trajectory data used.

The “Global Multiple Trajectory Tensor” is constructed by using sets of multiple trajectories extracted from video clips, as shown in Fig. 6. Here, “Global” refers to the multiple trajectories with global lengths.

The “Segmented Multiple Trajectory Tensor” is constructed by using sets of multiple subtrajectories. We jointly segment multiple trajectories with global lengths into atomic ”units” which are called *multiple subtrajectories*. Those multiple subtrajectories represent certain joint motion patterns of multiple objects within a certain time interval. The segmentation points that define the joint segmentation of multiple trajectories depend on the segmentation technique and in general shall correspond

to changes in joint motion patterns of multiple objects, such as changes in velocity (1st order derivative) and/or acceleration (2nd order derivative). Our segmentation technique ensures that only joint change of motions of multiple objects would be chosen as segmentation point, based on spatial curvatures of multiple trajectories. The spatial curvature of a 2-D trajectory is given by:

$$\Theta[t] = \frac{x'[t]y''[t] - y'[t]x''[t]}{[(x'[t])^2 + (y'[t])^2]^{3/2}}. \tag{46}$$

The value of curvature at any point is a measure of inflection point, an indication of concavity or convexity in the trajectory. For multiple trajectories, curvature data is calculated for each trajectory. For example, for $M(\geq 1)$ trajectories, there would be M curvatures. We segment the multiple trajectories by applying a moving window scheme and a hypothesis testing method on their spatial curvatures. Let X and Y be two non-overlapping windows of size n, where X contains the first n M-dimensional curvature vector samples, and Y contains the next n samples. Each M-dimensional curvature vector consists of curvatures from M trajectories. Let Z be the window of size $2n$ formed by concatenating window X and window Y. Then we perform a likelihood ratio hypothesis test to determine if the two windows X and Y have data drawn from the same distribution. If curvatures of multiple trajectories in X and Y are from different distributions, then there would be a change of concavity or convexity of multiple trajectories.

Specifically, we have two hypothesis:

$$\begin{cases} H_0 : f_x(X;\theta_x) = f_y(Y;\theta_y) = f_z(Z;\theta_z). \\ H_1 : f_x(X;\theta_x) \neq f_y(Y;\theta_x). \end{cases}$$

Assume that curvature data in each window forms an i.i.d random variable and the data is M-dimensional jointly Gaussian. We first compute the maximum likelihood estimator of mean and variance for each hypothesis,

$$L_0 = \left[\frac{1}{(2\pi)^{\frac{M}{2}}|\Sigma_3|^{\frac{1}{2}}}\right]^{2n} \exp\{-\frac{1}{2}\sum_{i=s}^{s+2n}(\underline{k_i} - \underline{\mu_3})^T \Sigma_3^{-1}(\underline{k_i} - \underline{\mu_3})\}$$

$$L_1 = \left[\frac{1}{(2\pi)^M|\Sigma_1|^{\frac{1}{2}}|\Sigma_2|^{\frac{1}{2}}}\right]^{n} \exp\{-\frac{1}{2}[\sum_{i=s}^{s+n}(\underline{k_i} - \underline{\mu_1})^T \Sigma_1^{-1}(\underline{k_i} - \underline{\mu_1}) + \sum_{i=s+n+1}^{s+2n}(\underline{k_i} - \underline{\mu_2})^T \Sigma_2^{-1}(\underline{k_i} - \underline{\mu_2})]\}. \tag{47}$$

then the likelihood ratio is defined as,

$$\lambda_L = \frac{L_0}{L_1}. \tag{48}$$

We finally calculate the distance d between the distributions of X and Y. We define the distance function d between X and Y as

$$d(s) = -log(\lambda_L) = -nlog(\frac{|\Sigma_1|^{1/2}|\Sigma_2|^{1/2}}{|\Sigma_3|})$$

$$+\frac{1}{2}\Big[\sum_{i=s}^{s+2n}(\underline{k_i}-\underline{\mu_3})^T\Sigma_3^{-1}(\underline{k_i}-\underline{\mu_3}) - \sum_{i=s}^{s+n}(\underline{k_i}-\underline{\mu_1})^T\Sigma_1^{-1}(\underline{k_i}-\underline{\mu_1})$$

$$-\sum_{i=s+n+1}^{s+2n}(\underline{k_i}-\underline{\mu_2})^T\Sigma_2^{-1}(\underline{k_i}-\underline{\mu_2})\Big]. \quad (49)$$

where $\underline{\mu_i}$, Σ_i $(i = 1,2,3)$ are mean column vectors and variance matrices of M-dimensional Gaussian distributions, which represent distributions of data in windows X,Y and Z, respectively. s is the start point of samples in window X, where $s = 1,2,...,L-2n$. $\underline{k_i}$ is curvature column vector which consists of curvature samples from M trajectories at time instant i, $\underline{k_i} = [\Theta_1[i],...,\Theta_M[i]]^T$. The distance d is large if the data in window X and Y have different distributions. The windows are moved by m $(< n)$ samples and the process is repeated in the same manner. A 1-D vector of distance function is then formed. The segmentation positions are chosen at the locations along the trajectory where the likelihood ratio d attains a local maximum. To select the local maxima, the 1-D vector of likelihood ratio d is partitioned into segments and the global maximum is selected within each partition to represent the segmentation location. The threshold α within each segment is chosen such that only the global maximum of the likelihood ratio d is selected to represent the segmentation location within each partition. Figure 8 displays the segmentation results of a set of multiple trajectories. Please note that the segmentation positions are chosen when both curvatures of the two trajectories have a sudden change, e.g. at segmentation points 2 and 4, which ensures that in our segmentation only joint change of motions of multiple objects would be chosen as segmentation point.

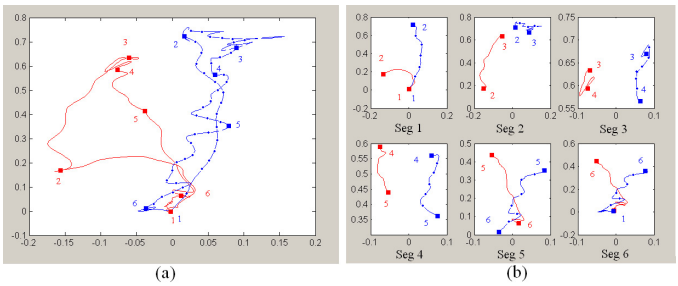

Fig. 8 Example of segmentation of a set of global multiple trajectories (2 trajectories) into segmented multiple subtrajectories. (a) a set of 2 trajectories with global length. (b) 6 sets of segmented multiple subtrajectories, segmented from (a). In both figures, solid squares with numbers indicate segmentation positions, the horizontal axis is x- location, the vertical axis is y- location within the scene.

2.4.4 Multiple-Trajectory Indexing and Retrieval Algorithms

We propose three multiple-object trajectory indexing and retrieval algorithms, that mainly differ in terms of representation techniques: (i) geometrical multiple-trajectory indexing and retrieval (GMIR) algorithm, (ii) unfolded multiple-trajectory indexing and retrieval (UMIR) algorithm, and (iii) concentrated multiple-trajectory indexing and retrieval (CMIR) algorithm.

2.4.5 Geometrical Multiple-Trajectory Indexing and Retrieval (GMIR) Algorithm

In the following we outline the Indexing and Retrieval processes of the GMIR algorithm.
Indexing

1. Generate geometric bases G_1, G_2 and G_3 from multiple trajectory tensor $\mathscr{T}$. The geometric bases represent the principle axes of variation across each coordinate axis in a 3D coordinate system. Specifically, G_1 spans the space of spatial-temporal multiple trajectory data, G_2 spans the space of object trajectory cardinality, G_3 spans the space of sets of multiple trajectories. They can be obtained by using various tensor analysis techniques, such as[35].
2. Project each multiple trajectory matrix onto the first two bases G_1 and G_2 and transform the multiple trajectory data into low-dimensional subspace defined by those two bases:
$$M_{Coeff} = G_1^T \times M_{Multi-Traj} \times G_2. \tag{50}$$
3. Use the coefficient matrices obtained in step 2 as indices of their corresponding multiple-trajectory matrices in the database.

Retrieval

1. Project multiple trajectories in the input query on the 2 bases G_1 and G_2, and get the GMIR coefficient matrix:
$$M_{QueryCoeff} = G_1^T \times M_{TrajQuery} \times G_2. \tag{51}$$
2. Calculate the Euclidean distance norm D_{GMIR} between the GMIR coefficients of query multiple-trajectory and the GMIR coefficients matrices stored in the database, and return the ones that have the distances within a threshold.
$$D_{GMIR} = ||(M_{Coeff} - M_{QueryCoeff})||^2. \tag{52}$$

where in eqns.(6)-(8), G_i is the i^{th} geometric base, G_i^T is transpose of G_i, M_{Coeff} and $M_{QueryCoeff}$ are the coefficient matrices of multiple object trajectory in multiple trajectory tensor and query multiple-trajectory, $M_{Multi-Traj}$ and $M_{TrajQuery}$ are multiple trajectory matrices in multiple trajectory tensor and query multiple object trajectory, respectively.

2.4.6 Unfolded Multiple-Trajectory Indexing and Retrieval (UMIR) Algorithm

The procedure of generating data-dependent bases of a multi-dimensional matrix or tensor using unfolded multiple-trajectory indexing and retrieval (UMIR) algorithm can be viewed as a recursive "unfold" process of tensor. By viewing slices of tensor, which are matrices, as vectors, the tensor can be viewed as a matrix, then first use SVD on the matrix, and then use SVD on the slice-matrices, as shown below:

$$\mathscr{T} = \sigma_1 \times_1 T_{1\ldots(N-1)} \times_2 U_N. \tag{53}$$

$$T_{1\ldots(N-1)} = \sigma_2 \times_1 T_{1\ldots(N-2)} \times_2 U_{N-1}. \tag{54}$$

$$\ldots$$

$$T_{12} = \sigma_{N-1} \times_1 U_1 \times_2 U_2. \tag{55}$$

Where in the above eqns., $\times_n$ refers to the mode-n product [36]. The indexing and retrieval processes of the UMIR algorithm are summarized below:
Indexing

1. Generate unfolded bases U_1, U_2 and U_3 from multi-dimensional data, by recursive "unfolding" of tensor $\mathscr{T}$:

$$\mathscr{T} = \sigma_1 \times_1 T_{12} \times_2 U_3. \tag{56}$$

$$T_{12} = \sigma_2 \times_1 U_1 \times_2 U_2. \tag{57}$$

2. Project each multiple trajectory matrix onto the 2 bases U_1 and U_2 and transform the multiple trajectory data into low-dimensional subspace spanned by those two bases:

$$M_{Coeff} = U_1^T \times M_{Multi-Traj} \times U_3. \tag{58}$$

3. Use the coefficient matrices obtained in step 2 as indices of their corresponding multiple-trajectory matrices in the database.

Retrieval

1. Project query multiple trajectories on the 2 bases U_1 and U_3, and get UMIR coefficient matrix:

$$M_{QueryCoeff} = U_1^T \times M_{TrajQuery} \times U_3. \tag{59}$$

2. Calculate the Euclidean distance norm D_{UMIR} between the UMIR coefficients of query multiple-trajectory and those of each multiple-trajectory input indexed in the database, and return the ones that have the distances within a threshold.

$$D_{UMIR} = ||(M_{Coeff} - M_{QueryCoeff})||^2. \tag{60}$$

Where in eqns.(9)-(16), $T_{1\ldots N}$ is a matrix indexed by 1 to N, and U_i are unfolded bases. U_i^T is transpose of U_i $(i = 1,2,3)$. M_{Coeff1}, M_{Coeff2}, $M_{QueryCoeff1}$,

$M_{QueryCoeff2}$ are coefficient matrices; $M_{Multi-Traj}$, $M_{TrajQuery}$ are the multiple-trajectory matrix and query multi-trajectory matrix.

2.4.7 Concentrated Multiple-Trajectory Indexing and Retrieval (CMIR) Algorithm

The Indexing and Retrieval processes used in the CMIR algorithm are outlined below:

Indexing

1. Generate concentrated bases C_1, C_2 and C_3 from multiple trajectory tensor $\mathscr{T}$ by minimizing the following sum-of-squares loss function:

$$\min_{C_1,C_2,C_3} \sum_{i,j,k} ||\mathscr{T}_{ijk} - \sum_{r}^{R} c_{ir}^1 c_{jr}^2 c_{kr}^3||^2. \tag{61}$$

   Where c_{ir}^1,c_{jr}^2, c_{kr}^3 are i, j and k-th column vectors of C_1, C_2 and C_3, respectively. The data-dependent bases can be obtained by solving the above equation. One solution to extract those bases is called Parallel Factors Decomposition (PARAFAC)[37].
2. Project each multiple trajectory matrix onto the 2 bases C_1 and C_2 and transform the multiple trajectory data into low-dimensional subspace spanned by those two bases:

$$M_{Coeff} = C_1^T \times M_{Multi-Traj} \times C_2. \tag{62}$$

3. Use the coefficient matrices obtained in step 2 as indices of their corresponding multiple-trajectory matrices in the database.

Retrieval

1. Project query multiple trajectories on the 2 bases C_1 and C_2, and get CMIR coefficient matrix:

$$M_{QueryCoeff} = C_1^T \times M_{TrajQuery} \times C_2. \tag{63}$$

2. Calculate the Euclidean distance norm D_{UMIR} between the UMIR coefficients of query multiple-trajectory and those of each multiple-trajectory input indexed in the database, and return the ones that have the distances within a threshold.

$$D_{CMIR} = ||(M_{Coeff} - M_{QueryCoeff})||^2. \tag{64}$$

Where in eqns.(14)-(17) C_i^T is transpose of $C_i i = 1,2,3$, $M_{Multi-Traj}$ and $M_{TrajQuery}$ are trajectory matrices of multiple-trajectory in the database and query multiple-trajectory, M_{Coeff} and $M_{QueryCoeff}$ are their corresponding coefficient matrices.

The experiment shown in Fig. 9 demonstrates the retrieval results on two trajectories in CAVIAR [38] dataset . The query is represented in Fig. 9(a). The most-similar and second-most-similar retrieval results are illustrated in Figs. 9(b) and

9(c), respectively. In this case, the query depicts a video clip “two people who meet, fight and chase each other”. The most-similar retrieved result show in Fig. 9(b) is a video clip in which “two people meet, walk together and split apart”; whereas the second-most similar retrieved result portrayed in Fig. 9(c) is a video clip in which “two people meet, fight and run away”. We can see the retrieved multiple trajectories are visually quite similar with the query. Figure 10 shows another retrieval result

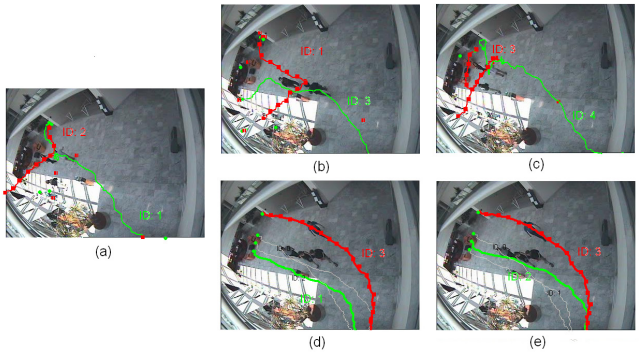

Fig. 9 Retrieval results for *CAVIAR* dataset (INRIA) using the proposed CMIR algorithm for multiple trajectory representation (2 trajectories): (a) the query; (b) the most-similar retrieval; (c) the second-most-similar retrieval; (d) the most-dissimilar retrieval; (e) the second-most-dissimilar retrieval.

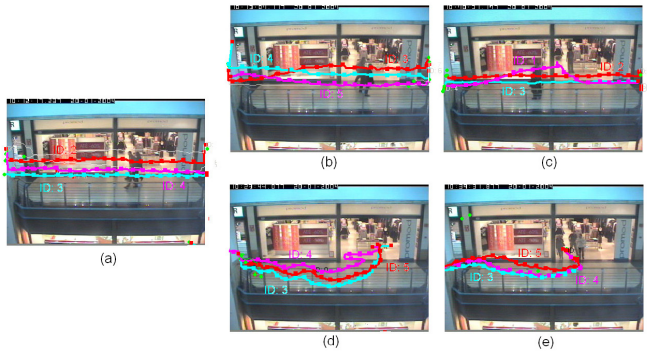

Fig. 10 Retrieval results for *CAVIAR* dataset (Shopping Center in Portugal) using the proposed CMIR algorithm for multiple trajectory representation (3 trajectories): (a) the query; (b) the most-similar retrieval; (c) the second-most-similar retrieval; (d) the most-dissimilar retrieval; (e) the second-most-dissimilar retrieval.

on *CAVIAR* data for three-trajectory case. Trajectory data are selected from the clips of shopping center in Portugal (2nd subset in CAVIAR). In this scenario, the query

depicts a video sequence “3 persons walking in the corridor”. The most-similar retrieved result show in Fig. 10(b) is a video clip in which “Another 3 persons walking in the corridor”; whereas the second-most similar retrieved result portrayed in Fig. 10(c) is a video clip in which “3 people walking together along the corridor”. On contrast, the most-dissimilar retrieval and the second-most-dissimilar retrieval are depicted in Figs. 10 (d) and (e), respectively. Both of retrieved dissimilar results visually vary a lot from the query.

2.5 Open Problems and Future Trends

From the above review, we see the potential and promising future of video indexing and retrieval systems, however, there are still many open research issues that need to be addressed to make the full use out of visual information retrieval systems. In the following, we identify several open problems and provide a brief summary of future trends.

1. Dynamic matching, updating of query and databases: the practical utility of a robust CBVIR system must address the problem of dynamic updating of video databases and feature spaces, as well as dynamic matching of queries and databases [39].
2. Low to High Level Semantic Gap: Visual feature based techniques at the low-level of abstraction, mostly from the contribution of signal processing and computer vision communities have been explored in the literature. Current research efforts are more inclined towards high-level description and retrieval of visual content. Most of the techniques at high level of abstraction assume the availability of high-level representation and process that information for indexing. The techniques that bridge this semantic gap between pixels and predicates are a field of growing interest. Intelligent systems are needed that take low-level feature representation of the visual media and provide a model for the high-level object representation of the content.

3 Video Classification and Recognition

Video classification differs from video indexing and retrieval, since in video classification, all videos are put into categories, and each video is assigned a meaningful label. While in video indexing and retrieval, the aim is to accurately retrieve videos that match a user’s query. Many automatic video classification algorithms have been proposed, most of them can be categorized into four groups: text-based approaches, audio-based approaches, visual-based approaches, and combination of text, audio and visual features. Many standard classifiers, such as Gaussian Mixture Models (GMM), Bayesian, support vector machines (SVM), neural networks and hidden Markov Models (HMMs) have been applied in video classification and recognition.

For more details, we refer to [40]. We propose a novel distributed multi-dimensional hidden Markov Model (DHMM) for modelling of interacting trajectories involving multiple objects. Our model is capable of conveying not only dynamics of each trajectory, but also interactions information between multiple trajectories, while requiring no further semantic analysis.

3.1 Model-based Classification

3.1.1 Hidden Markov Model

Hidden Markov model (HMM) is very powerful tool to model temporal dynamics of processes, and has been successfully applied to many applications such as speech recognition [41], gesture recognition [42], musical score following [43]. F. Bashir et al. [44] presented a novel classification algorithm of object motion trajectory based on 1D HMM. They segmented single trajectory into atomic segments called subtrajectories based on curvature of trajectory, then the subtrajectories are represented by their principal component analysis (PCA) coefficients. Temporal relationships of subtrajectories are represented by fitting a 1D HMM. However, all the above applications rely on a one-dimensional HMM structure. Simple combinations of 1D HMMs can not be used to characterize multiple trajectories, since 1D models fail to convey interaction information of multiple interacting objects. The major challenge here is to develop a new model that will semantically reserve and convey the "interaction" information.

3.1.2 Multi-dimensional Distributed Hidden Markov Models

We propose a novel distributed multi-dimensional hidden Markov Model (DHMM) for modelling of interacting trajectories involving multiple objects. In our model, each object-trajectory is modelled as a separate Hidden Markov process; while "interactions" between objects are modelled as dependencies of state variables of one process on states of the others. The intuition of our work is that, HMM is very powerful tool to model temporal dynamics of each process (trajectory); each process (trajectory) has its own dynamics, while it may be influenced by or influence others. In our proposed model, "influence" or "interaction" among processes (trajectories) are modelled as dependencies of state variables among processes (trajectories). Our model is capable of conveying not only dynamics of each trajectory, but also interactions information between multiple trajectories, while requiring no semantic analysis. Figure 11 demonstrates examples of multiple interactive motion trajectories.

We first provide a solution for non-causal, multi-dimensional HMMs by distributing the non-causal model into multiple distributed causal HMMs. We approximate the simultaneous solution of multiple distributed HMMs on a sequential processor

Fig. 11 Examples of multiple interactive trajectories: (a) "Two people walk and meet". (b) "Two air planes fly towards each other and pass by".

by an alternate updating scheme. Subsequently we extend the training and classification algorithms presented in [45] to a general causal model. A new Expectation-Maximization (EM) algorithm for estimation of the new model is derived, where a novel General Forward-Backward (GFB) algorithm is proposed for recursive estimation of the model parameters. A new conditional independent subset-state sequence structure decomposition of state sequences is proposed for the 2D Viterbi algorithm. The new model can be applied to many problems in pattern analysis and classification. For simplicity, the presentation in this paper will focus primarily on a special case of the our model in two-dimensions, which we referred to as distributed 2D hidden Markov Models (2D DHMMs).

Suppose there are $M \in \mathbb{N}$ interacting objects in a scene. Recall that in our model, each object-trajectory is modelled as a Hidden Markov process of time; while "interactions" between object trajectories are modelled as dependencies of state variables of one process on those of the others. We constrain the probabilistic dependencies of state in one process (trajectory) at time t, on its own state at time t-1, as well as on the states of other processes (trajectories) that "interact" or influence on it at time t and t-1, i.e.

$$Pr(s(m,t)|s(l,t),s(n,1:t-1)) = Pr(s(m,t)|s(n,t-1),s(l,t)) \tag{65}$$

where $m,n,l \in \{1,...,M\}$ are indexes of processes (trajectories), $l \neq m$. The above constrain of state dependencies makes the desired model non-causal, since each process (trajectory) can influence others, there is no guarantee that the influence should be directional or causal. Fig. 4(a) shows an example of our non-causal 2D HMM, which is used to model two interacting trajectories. Each node S(i; t) in the figure represents one state at specific time t for trajectory i, where $t = \{1,2,...,T\}$, i = {1,2}; each node O(i,t) represents observations corresponding to S(i,t), and each arrow indicates transition of states (the reverse direction of it indicates dependency of states). The first row of states is the state sequence for trajectory 1, and the second row corresponds to trajectory 2. As can be seen, each state in one HMM chain (trajectory) will depend on its past state, the past state of the other HMM chain (trajectory), and the concurrent state of the other HMM chain (trajectory).

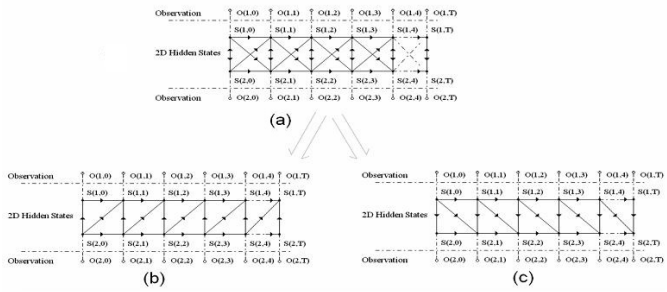

Fig. 12 Distributed 2D Hidden Markov Models: (a) Non-causal 2D Hidden Markov Model. (b) Distributed 2D Hidden Markov Model 1. (c) Distributed 2D Hidden Markov Model 2.

The above model is capable of modelling multiple processes and their interactions, but it is intractable since it is non-causal. We propose a novel and effective solution to it, where we "decompose" it into M causal 2D hidden Markov models with multiple dependencies of states, such that each HMM can be executed in parallel in a distributed framework. In each of the distributed causal HMM, state transitions (or state dependencies) must follow the same causality rule. For example, we distributed the non-causal 2D HMM in fig. 12(a) to two causal 2D HMMs, shown in figs. 12(b) and 12(c), respectively. In fig. 12(b), state transitions follow the same rule, so do state transitions in fig. 12(c). The above rules ensure the homogeneous structure of each distributed HMM, which further enable us to develop relatively tractable training and classification algorithms.

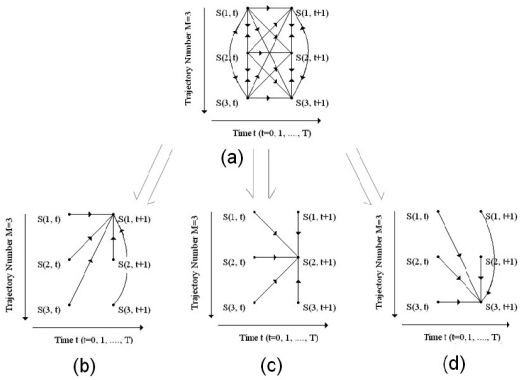

Fig. 13 Distributed 2D Hidden Markov Models with application to 3 trajectories: (a) Non-causal 2D Hidden Markov Model that treat 3 object trajectory as one system(only 2 adjacent time slots of the system states are shown). (b) Distributed 2D Hidden Markov Model 1 for Object Trajectory 1. (c) Distributed 2D Hidden Markov Model 2 for Object Trajectory 2. (d) Distributed 2D Hidden Markov Model 3 for Object Trajectory 3. (Please note in Figures (b) (c) and (d), only state transitions to one state point is shown, other state points follow the same rules, respectively).

When trajectory number M=3, our non-causal 2D hidden Markov model is depicted as in Fig. 13(a), we use the same distributing scheme to get 3 distributed 2d hidden Markov models, as shown in Fig. 13(b), 13(c), and 13(d), respectively. Using the same distributing scheme, we can distribute any non-causal 2D hidden Markov model that characterizing $M(>3)$ trajectories, to M distributed causal 2D hidden Markov models.

3.1.3 DHMM TRAINING AND CLASSIFICATION

Define the observed feature vector set O = {o(m,t), m = 1,2,...,M; t = 1,2,...,T } and corresponding hidden state set S = {s(m,t), m = 1,2,...,M; t = 1,2,...,T }, and assume each state will take N possible values. The model parameters are defined as a set $\Theta = \{\Pi, \mathbf{A}, \mathbf{B}\}$, where Π is the set of initial probabilities of states $\Pi = \{\pi(m,n)\}$; $\mathbf{A}$ is the set of state transition probabilities $A = \{a_{i,j,k,l}(m)\}$, and $a_{i,j,k,l}(m) = Pr(s(m,t) = l | s(m',t) = k, s(m,t-1) = i, s(m',t-1) = j)$, and $\mathbf{B}$ is the set of probability density functions (PDFs) of the observed feature vectors given corresponding states, assume $\mathbf{B}$ is a set of Gaussian distribution with means $\mu_{m,n}$ and variances $\Sigma_{m,n}$, where $m, m' = 1,...,M; m \neq m'; n, i, j, k, l = 1,...,N; t = 1,...,T$. Due to space limit, we will discuss the case $M = 2$. For more details, we refer to [46] [47] [48] [49].

Define $F^{(p)}_{m,n,k,l}(i,j)$ as the probability of state corresponding to observation $o(i-1,j)$ is state m, state corresponding to observation $o(i-1,j-1)$ is state n, state corresponding to observation $o(i,j-1)$ is state k and state corresponding to observation $o(i,j)$ is state l, given the observations and model parameters,

$$F^{(p)}_{m,n,k,l}(i,j) = P\Big(m = s(i-1,j), n = s(i-1,j-1), k = s(i,j-1), l = s(i,j) | O, \Theta^{(p)}\Big), \tag{66}$$

and define $G^{(p)}_m(i,j)$ as the probability of the state corresponding to observation $o(i,j)$ is state m, then

$$G^{(p)}_m(i,j) = P(s(i,j) = m | O, \Theta^{(p)}). \tag{67}$$

We can get the iterative updating formulas of parameters of the proposed model,

$$\pi^{(p+1)}_m = P(G^{(p)}_m(1,1) | O, \Theta^{(p)}). \tag{68}$$

$$a^{(p+1)}_{m,n,k,l} = \frac{\sum_i^I \sum_j^J F^{(p)}_{m,n,k,l}(i,j)}{\sum_{l=1}^M \sum_i^I \sum_j^J F^{(p)}_{m,n,k,l}(i,j)}. \tag{69}$$

$$\mu^{(p+1)}_m = \frac{\sum_i^I \sum_j^J G^{(p)}_m(i,j) o(i,j)}{\sum_i^I \sum_j^J G^{(p)}_m(i,j)}. \tag{70}$$

$$\Sigma_m^{(p+1)} = \frac{\sum_i^I \sum_j^J G_m^{(p)}(i,j)(o(i,j)-\mu_m^{(p+1)})(o(i,j)-\mu_m^{(p+1)})^T}{\sum_i^I \sum_j^J G_m^{(p)}(i,j)}. \quad (71)$$

In eqns. (1)-(6), p is the iteration step number. $F_{m,n,k,l}^{(p)}(i,j)$, $G_m^{(p)}(i,j)$ are unknown in the above formulas, next we propose a General Forward-Backward (GFB) algorithm for the estimation of them.

Forward-Backward algorithm was firstly proposed by Baum et al. [50] for 1D Hidden Markov Model and later modified by Jia Li et al. in [45]. Here, we generalize the Forward-Backward algorithm in [50] [45] so that it can be applied to our model, the proposed algorithm is called General Forward-Backward (GFB) algorithm. Assume the probability of all-state sequence S can be decomposed as products of probabilities of conditional-independent subset-state sequences $U_0, U_1, ...$, i.e., $P(S) = P(U_0)P(U_1/U_0)...P(U_i/U_{i-1})...$, where $U_0, U_1, ..., U_i...$are subsets of all-state sequence in the HMM system, we call them *subset-state sequences*. Define the observation sequence corresponding to each subset-state sequence U_i as O_i.

Define the forward probability $\alpha_{U_u}(u), u = 1,2,...$ as the probability of observing the observation sequence $O_v (v \leq u)$ corresponding to subset-state sequence $U_v (v \leq u)$ and having state sequence for u-th product component in the decomposing formula as U_u, given model parameters Θ, i.e. $\alpha_{U_u}(u) = P\{S(u) = U_u, O_v, v \leq u|\Theta\}$, and the backward probability $\beta_{U_u}(u), u = 1,2,...$ as the probability of observing the observation sequence O_v $(v > u)$ corresponding to subset-state sequence $U_v (v > u)$, given state sequence for u-th product component as U_u and model parameters Θ, i.e. $\beta_{U_u}(u) = P(O_v, v > u|S(u) = U_u, \Theta)$.

The recursive updating formula of forward and backward probabilities can be obtained as

$$\alpha_{U_u}(u) = [\sum_{u-1} \alpha_{U_{u-1}}(u-1)P\{U_u|U_{u-1}, \Theta\}]P\{O_u|U_u, \Theta\}. \quad (72)$$

$$\beta_{U_u}(u) = \sum_{u+1} P(U_{u+1}|U_u, \Theta)P(O_{u+1}|U_{u+1}, \Theta)\beta_{U_{u+1}}(u+1). \quad (73)$$

Then, the estimation formulas of $F_{m,n,k,l}(i,j)$, $G_m(i,j)$ are :

$$G_m(i,j) = \frac{\alpha_{U_u}(u)\beta_{U_u}(u)}{\sum_{u:U_u(i,j)=m} \alpha_{U_u}(u)\beta_{U_u}(u)}. \quad (74)$$

$$F_{m,n,k,l}(i,j) = \frac{\alpha_{U_{u-1}}(u-1)P(U_u|U_{u-1}, \Theta)P(O_u|U_u, \Theta)\beta_{U_u}(u)}{\sum_u \sum_{u-1}[\alpha_{U_{u-1}}(u-1)P(U_u|U_{u-1}, \Theta)P(O_u|U_u, \Theta)\beta_{U_u}(u)]}. \quad (75)$$

For classification, we employ a two-dimensional Viterbi algorithm [51] to search for the best combination of states with maximum a posteriori probability and map each block to a class. This process is equivalent to search for the state of each block using an extension of the variable-state Viterbi algorithm presented in [45]. If we search for all the combinations of states, suppose the number of states in each subset-state sequence U_u is $w(u)$, then the number of possible sequences of states

at every position will be $M^{w(u)}$, which is computationally infeasible. To reduce the computational complexity, we only use N sequences of states with highest likelihoods out of the $M^{w(u)}$ possible states.

For simplicity, we only tested our DHMM model-based multiple trajectory classification algorithm on the M=2 trajectory cases. We test the classification performance of both proposed distributed 2D HMM-based classifier, causal 2D HMM-based classifier and traditional 1D HMM-based classifier on subset of the CAVIAR [38] dataset. Figure 14 displays samples of two classes in the dataset. We report

(a) (b)

Fig. 14 Multiple-trajectories samples of two classes in CAVIAR dataset: (a) multiple-trajectories (M=2) sample and one frame from class 1: "Two people meet and walk together"; (b) multiple-trajectories (M=2) sample and one frame from class 2: "Two people meet, fight and run away".

the classification results in terms of ROC curve [52]. As can be seen, the proposed model outperforms the existing ones.

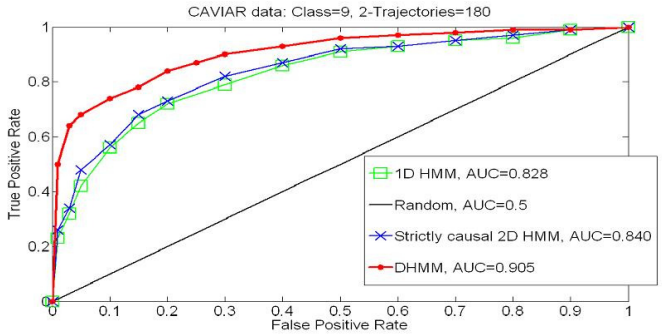

Fig. 15 ROC curve of DHMM, Strictly Causal 2D HMM [45] and 1D HMM for CAVIAR data

3.2 Open Problems and Future Trends

Many works have been done in video classification and recognition, most of them use text, audio or visual features (such as motion trajectory) or combination of features. Despite the success of many methods, there are still many open problems that need to be addressed in video classification and recognition.

1. Robust Modelling of Video Spatial-Temporal Structure: Although various HMM models have been applied to video classification, the rich spatial-temporal structure of video has not yet been fully explored. A robust model that utilizes various features of videos and captures spatial-temporal structure of video is highly desired.
2. Efficient Fusion of Various Features: Combination of more features such as text, audio and visual feature would improve the performance of video classification systems. However, very few work has been done in efficient fusion of different features.

References

1. F. I. Bashir, A. A. Khokhar, and D. Schonfeld, "A Hybrid System for Affine-Invariant Trajectory Retrieval", Proceedings of the 6th ACM SIGMM International Workshop on Multimedia Information Retrieval, New York, New York, 2004.
2. 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, pp. 1912-1919, 2007.
3. F. I. Bashir, A. A. Khokhar, and D. Schonfeld, "Real-time motion trajectory-based indexing and retrieval of video sequences", IEEE Transactions on Multimedia, vol. 9, pp. 58-65, 2007.
4. F. I. Bashir, A. A. Khokhar, and D. Schonfeld, "Real-time affine-invariant motion trajectory-based retrieval and classification of video sequences from arbitrary camera view," ACM Multimedia Systems Journal, Special Issue on Machine Learning Approaches to Multimedia Information Retrieval, vol. 12, pp. 45-54, 2006.
5. The University of California at Irvine Knowledge Discovery in Databases (KDD) archive, [Online]. Available: http://kdd.ics.uci.edu, URL.
6. N. Vaswani, R. Chellappa, "Principal Components Null Space Analysis for Image and Video Classification," IEEE Trans. Image Processing, July 2006.
7. X. Chen, D. Schonfeld and A. Khokhar, "Robust null space representation and sampling for view invariant motion trajectory analysis", IEEE Conference on Computer Vision and Pattern Recognition, 2008.
8. H. Zhang, J. Wu, D. Zhong, and S. W. Smoliar, "An integrated system for content-based video retrieval and browsing", Pattern Recognition, Vol. 30, no.4, pp. 643-658, 1997.
9. T. Lin, H. J. Zhang, and Q. Y. Shi, "Video Content Representation for Shot Retrieval and Scene Extraction", International Journal of Image and Graphics, Vol. 1, No. 3, July 2001.
10. C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic and W. Equitz, "Efficient and Effective Querying by Image Content", Journal of Intelligent Information Systems, 3, 3/4, July 1994, pp. 231-262.
11. T. Ng, S. Chang and M. Tsui, "Using Geometry Invariants for Camera Response Function Estimation", IEEE Conference on Computer Vision and Pattern Recognition, 2008.

12. X. Chen, D. Schonfeld and A. Khokhar, "View-Invariant Tensor Null-Space Representation for Multiple Motion Trajectory Retrieval and Classification", IEEE International Conference on Acoustics, Speeach, and Signal Processing, 2009.
13. [Online] Stephane Marchand-Maillet, "Content-based Video Retrieval: An overview", http://viper.unige.ch/ marchand/CBVR/, URL.
14. N. Sebe, M. S. Lew, X. Zhou, T. S. Huang and E. M. Bakker, "The State of the Art in Image and Video Retrieval", in proceedings of the International Conference on Image and Video Retrieval (CIVR), pages 1-8, 2003.
15. N. Sebe, M. S. Lew and A. W. M. Smeulders, "Video retrieval and summarization", editorial introduction, Computer Vision and Image Understanding (CVIU), 2003.
16. Y. Yang and L. Ming, "A Survey on Content based Video Retrieval", Hong Kong University of Science and Technology.
17. J. Yuan, H. Wang, W. Zheng, J. Li, F. Lin and B. Zhang, "A Formal Study of Shot Boundary Detection", IEEE Transactions on Circuit and Systems for Video Technology, pp. 168-186, 2007.
18. A. Hanjalic, "Shot-boundary detection: unraveled or resolved?", IEEE Transactions on Circuit and Systems for Video Technoligy, vol. 12, issue 2, pp. 90-105, 2002.
19. R. Lienhart, "Reliable Transition Detection in Videos: A Survey and Practitioner's Guide", International Journal of Image and Graphics, vol. 1, pp. 469-486, 2001.
20. M. M. Yeung and B. Liu, "Efficient Matching and Clustering of Video Shots", technical report, Princeton University, 1995.
21. H. J. Zhang, S. W. Smoliar and J. H. Wu, "Content-based Video Browsing Tools", SPIE Conference on Multimedia Computing and Networking, San Jose, CA, 1995.
22. W. Wolf, "Key Frame Selection by Motion Analysis", IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 1228-1231, 1996.
23. X. Liu, C. B. Owen and F. Makedon, "Automatic Video Pause Detection Filter", Technical Report PCS-TR97-307, Dartmouth College, Computer Science, Hanover, NH, 1997.
24. S. F. Chang, W. Chen, H. J. Meng, H. Sundaram and D. Zhong, "A Fully Automated Content-Based Video Search Engine Supporting Spatiotemporal Queries", IEEE Trans. Circuits Syst. Video Technol., vol.8, no.5, pp. 602-615, 1998.
25. N. AbouGhazaleh, Y. E. Gamal, "Compressed Video Indexing Based on Object's Motion", Int. Conf. Visual Communication and Image Processing, VCIP'00, Perth, Australia, pp. 986-993, 2000.
26. B. Katz, J. Lin, C. Stauffer and E. Grimson, "Answering questions about moving objects in surveillance videos", in proceedings of 2003 AAAI Spring Symp. New Directions in Question Answering, Palo Alto, CA, 2003.
27. N. Rea, R. Dahyot and A. Kokaram, "Semantic Event Detection in Sports Through Motion Understanding", in proc. 3rd Int. Conf. on Image and Video Retrieval (CIVR), pp. 21-23, 2004.
28. E. Sahouria, A. Zakhor, "A Trajectory Based Video Indexing System For Street Surveillance", IEEE Int. Conf. on Image Processing (ICIP), pp. 24-28, 1999.
29. W. Chen and S. F. Chang, "Motion Trajectory Matching of Video Objects", IS&T/ SPIE, pp. 544-553, 2000.
30. F. I. Bashir, A. A. Khokhar and D. Schonfeld, "Segmented trajectory based indexing and retrieval of video data", in proc. IEEE Int. Conf. Image Processing, pp.623-626, 2003.
31. T. Zhao and R. Nevatia, "Tracking multiple humans in crowded environment", in proc. IEEE Int. Conf. Compt. Vision and Pattern Recognit., vol. 2 (2004), pp. 406-413, 2004.
32. C. Chang, R. Ansari and A. Khokhar, "Multiple Object Tracking with Kernal Particle Filter", in proc. IEEE Int. Conf. Compt. Vision and Pattern Recognit., vol. 1 (2005), pp. 566-573, 2005.
33. X. Ma, F. Bashir, A. Knokhar and D. Schonfeld, "Event Analysis Based on Multiple Interactive Motion Trajectories", IEEE Trans. on Circuits and Syst. for Video Technology, vol 19, no 3, 2009.

34. X. Ma, F. Bashir, A. Knokhar and D. Schonfeld, “Tensor-based Multiple Object Trajectory Indexing and Retrieval”, in proc. IEEE Int. Conf. on Multimedia and Expo.(ICME), toronto, Canada, pp. 341-344, 2006.
35. L. Lathauwer, B. D. Moor and J. Vandewalle, “A multilinear singular value decomposition”, SIAM Journal on Matrix Analysis and Applicat. (SIMAX), vol. 21, issue 4, pp. 1253-1278, 2000.
36. L. D. Lathauwer and B. D. Moor, “From Matrix to Tensor: Multilinear Algebra and Signal Processing”, in proc. 4th IMA Int. Conf. Mathmatics in Signal Process., pp. 1-15, 1996.
37. R. A. Harshman, “Foundations of the PARAFAC procedure: Model and Conditions for an “explanatory” multi-mode factor analysis”, UCLA Working Papers in Phonetics, pp.1-84, 1970.
38. The Context Aware Vision using Image-based Active Recognition (CAVIAR) dataset, [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CAVIAR/, URL.
39. X. Ma, D. Schonfeld, and A. Khokhar, ”Dynamic updating and downdating matrix SVD and tensor HOSVD for adaptive indexing and retrieval of motion trajectories,” IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP09), Taipei, Taiwan, 2009.
40. D. Brezeale and D. J. Cook, ”Automatic Video Classification: A Survey of the Literature”, IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews, vol. 38, no. 3, 2008.
41. L. R. Rabiner, ”A tutorial on hidden markov models and selected applications in speech recognition”, Proceedings of the IEEE, vol. 77, pp. 257-286, 1989.
42. T. Starner and A. Pentland, ”Real-Time American Sign Language Recognition From Video Using Hidden Markov Models”, Technical Report, MIT Media Lab, Perceptual Computing Group, vol. 375, 1995.
43. C. Raphael, ”Automatic Segmentation of Acoustic Musical Signals Using Hidden Markov Models”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, issue 4, pp. 360-370, 1999.
44. F. I. Bashir and A. A. Khokhar and D. Schonfeld, ”HMM based motion recognition system using segmented pca”, IEEE International Conference on Image Processing (ICIP’05), vol. 3, pp. 1288-1291, 2005.
45. J. Li and A. Najmi and R. M. Gray, ”Image classification by a two-dimensional hidden markov model”, IEEE Trans. on Signal Processing, vol. 48, pp. 517-533, 2000.
46. X. Ma, D. Schonfeld and A. Khokhar , ”Distributed multidimensional hidden Markov model: theory and application in multiple-object trajectory classification and recognition”, SPIE International Conference on Multimedia Content Access: Algorithms and Systems, San Jose, California, 2008.
47. X. Ma, D. Schonfeld and A. Khokhar , ”Image segmentation and classification based on a 2D distributed hidden Markov model”, SPIE International Conference on Visual Communications and Image Processing (VCIP 08’), San Jose, California, 2008
48. X. Ma, D. Schonfeld and A. Khokhar , ”Distributed Multi-dimensional Hidden Markov Models for Image and Trajectory-Based Video Classification”,, IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 08’), Las Vegas, Nevada, 2008.
49. X. Ma, D. Schonfeld and A. Khokhar , ”Video Event Classification and Image Segmentation Based on Non-Causal Multi-Dimensional Hidden Markov Models”, IEEE Transactions on Image Processing (T-IP), to appear, May 2009.
50. L. E. Baum, T. Petrie, G. Soules and N. Weiss, ”A maximization technique occuring in the statistical analysis of probabilistic functions of markov chains”, Ann. Math. Stat., vol. 1, pp. 164-171, 1970.
51. D. Schonfeld and N. Bouaynaya, ”A new method for multidimensional optimization and its application in image and video processing”, IEEE Signal Processing Letters, vol. 13, pp. 485-488, 2006.
52. T. Fawcett, ”Roc graphs: Notes and practical considerations for researchers”, Technical Report, HP Labs, HPL-2003-4, 2004.