Motion Flow-Based Video Retrieval

Chih-Wen Su, Hong-Yuan Mark Liao, *Senior Member, IEEE*, Hsiao-Rong Tyan,
Chia-Wen Lin, *Senior Member, IEEE*, Duan-Yu Chen, and Kuo-Chin Fan, *Member, IEEE*

***Abstract*—In this paper, we propose the use of motion vectors embedded in MPEG bitstreams to generate so-called "motion flows", which are applied to perform video retrieval. By using the motion vectors directly, we do not need to consider the shape of a moving object and its corresponding trajectory. Instead, we simply "link" the local motion vectors across consecutive video frames to form motion flows, which are then recorded and stored in a video database. In the video retrieval phase, we propose a new matching strategy to execute the video retrieval task. Motions that do not belong to the mainstream motion flows are filtered out by our proposed algorithm. The retrieval process can be triggered by query-by-sketch or query-by-example. The experiment results show that our method is indeed superb in the video retrieval process.**

***Index Terms*—Motion analysis, video retrieval.**

I. Introduction

THE development of video technology in recent years has become a very important research field [2], [6], [14]–[16], [21]. Considering the rapid increase in digital video content, an efficient way to access and manipulate the information in a vast database has become a challenging and timely issue. Some commercial search engines, such as Google [19] and Yahoo! [20], have started to extend their services to video searching on the Internet, and it is already possible to search for video clips by inputting keywords. However, commonly adopted features, such as color, texture, or shape, are still insufficient to describe the rich visual content of a video clip. In the past few years, the area of content-based multimedia retrieval has attracted worldwide attention. Some experimental systems, for example, QBIC [1], Virage [9], PhotoBook [7], VisualSeek [8], Video-Q [2], and Netra-V [10] have successfully applied semantic-based visual content to multimedia database retrieval. Among the different types of features used in previous content-based video retrieval (CBVR) systems, the motion feature has played a very important role. A motion-based feature can be further processed into a feature that covers spatial and temporal characteristics simultaneously. Such a feature has a better chance of effectively executing video retrieval tasks.

A large number of motion-based video retrieval systems have been proposed in the past decade [1], [2], [5], [6], [11]–[13]. VideoQ [2] is a notable approach that directly addresses motion-based characterization of video content. In fact, most existing CBVR systems utilize motion as one of the features when executing a search process. The major difference among these systems is the way they extract and manage a motion-based feature in the video retrieval process. We can classify existing motion descriptors into two types: 1) statistics-based and 2) object-based. Researchers who work with the first type use statistics to analyze the tendency and distribution of local motion. For example, Fablet *et al.* [6] used causal Gibbs models to represent the spatio-temporal distribution of the dynamic content in a video shot. Ma and Zhang [11] generated a multi-dimensional vector by measuring the energy distribution of a motion vector field. In MPEG-7 [13], some fundamental bases of motion activity, such as intensity, direction, and spatial-temporal distribution, have been adopted to retrieve content from video databases. With regard to object-based motion descriptors, Chang *et al.* [2] proposed grouping regions that are similar in color, texture, shape, and motion together to form a video object. They adopted the query-by-sketch mechanism (QBS) to retrieve video content from a database. The constituents of their video database comprise a set of trajectories formed by linking the centroids of video objects across consecutive frames. In [5], Dagtas *et al.* proposed using a combination of trajectory- and trail-based models to characterize the motion of a video object. They adopted a Fourier transform-based similarity metric and a two-stage comparison strategy to search for similar trajectories. Although a number of video object segmentation algorithms [14]–[16] have been proposed in the past decade, this issue is still a very challenging task due to its complexity and ill-posed nature.

Both statistics-based and object-based approaches have their own advantages and drawbacks. A statistics-based approach is usually fast in computation. However, its drawback is the poor power on characterizing relational features. An object-based motion descriptor, on the other hand, needs to record the trajectory of a video object; therefore, segmentation of video objects is indispensable to the process. However, it is well known that the segmentation of video objects is an ill-posed problem that

Manuscript received December 12, 2005; revised March 15, 2007. This work was supported in part by the Ministry of Economic Affairs of Taiwan under Grant Number 96-EC-17-A-02-S1-032, and in part by the National Science Council of Taiwan under Grant NSC95-2422-H-001-007. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Yong Rui.

C.-W. Su and D.-Y. Chen are with the Institute of Information Science, Academia Sinica, Nankang, Taipei, 115, Taiwan (e-mail: lucas@iis.sinica.edu.tw; dychen@iis.sinica.edu.tw).

H.-Y. M. Liao is with the Institute of Information Science, Academia Sinica, Nankang, Taipei, 115, Taiwan and the Department of Computer Science, National Chiao-Tung University, Hsinchu, 300, Taiwan (e-mail: liao@iis.sinica.edu.tw).

H.-R. Tyan is with the Institute of Information and Computer Engineering, Chung Yuan University, Chung-Li, 320, Taiwan, R.O.C.

C.-W. Lin is with the Department of Electrical Engineering, National Tsing-Hua University, Hsinchu 30013, Taiwan, R.O.C. (e-mail: cwlin@ee.nthu.edu.tw).

K.-C. Fan is with the Department of Computer Science and Information Engineering, National Central University, Chung-Li 320, Taiwan, R.O.C. (e-mail: kcfan@csie.ncu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2007.902875

makes it impossible to "automatically" and "correctly" derive the centroids of video objects across consecutive frames in every case. On the other hand, it is not feasible to calculate the above-mentioned trajectory, because the process is very time consuming. For some real world applications, such as web searches, video annotation must be efficient and precise, since its results can be used to make several comparisons. Therefore, we propose a new approach that makes the CBVR process as accurate as possible. Since most stored videos are compressed, we directly utilize the motion vectors embedded in an MPEG bitstream to develop a motion descriptor. It is known that motion vectors only record the direction and magnitude of movement between corresponding macroblocks of two consecutive anchor frames, as they are only comprised of local data that does not have much semantic meaning. In this study, we utilize the consistency of motion direction, the color distribution, and the overlapping area between macroblocks of two consecutive frames to "link" all neighboring motion vectors. These linked motion vectors form so-called "motion flows", which contain more semantic meaning than the original motion vector data. Since a large moving object may occupy several macroblocks and produce multiple motion flows, we also propose an algorithm to reduce motion flows that are similar in shape to one or more representative motion flows. We approximate these representative motion flows by generating control points and storing them in a database as models. In addition, our approach allows multiple representative motion flows for an object, if it contains several independent moving parts.

Fig. 1 shows the procedure of the proposed approach. We first segment videos into shots [22], [23], eliminate global camera motion, and then extract the residual local motion vectors from each frame. Since motion vectors are dispersed in the spatial domain, we construct motion flows from the vectors to generate continual motion information as a trajectory. Finally, we remove redundant motion flows and store the remainder in the database. When a user wants to query a specific motion, he/she can draw a trajectory on a sketch-based interface, and the system will retrieve a set of trajectories that are similar to the query. Although our method is more close to object-based approach, it has several advantages in comparison with the state of the art object-based approaches [1], [2], [5], [6]. In [1], [2], [5], and [6], the methods they propose all need to spend extra computation power to handle object segmentation or to compute optical flows in the spatial domain. However, the derivation of motion flows in our approach is executed in the compressed domain directly. Since the motion vectors that can be used for establishing motion flows are already encoded in MPEG bitstreams, the derivation process is much easier than calculating video objects in the spatial domain. To compare with other work that also generates trajectories directly from motion vectors, our approach is also better. In [24], Babu and Ramakrishnan proposed a compressed domain video retrieval mechanism using object and global motion descriptors. They generate trajectories from motion vectors and then group homogeneous trajectories to form objects. Since the scheme has to deal with the object segmentation issue, it requires more computation time than our approach. The experiment results also show that our approach is indeed superb.

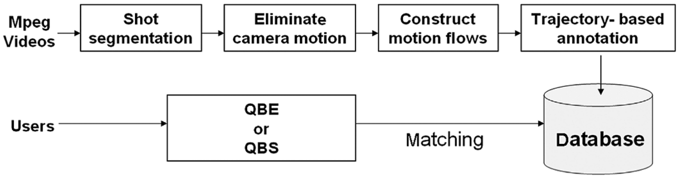


Fig. 1. Procedure of our approach.

The remainder of this paper is organized as follows. In Section II, we describe how to construct motion flows from an MPEG video. In Section III, we introduce a novel comparison algorithm that calculates the similarity degree between two distinct motion flows. We then present our experiment results and conclusions in Sections IV and V, respectively.

II. Constructing Motion Flows From MPEG Bitstreams

In this section, we introduce the method for constructing motion flows from an MPEG bitstream. As mentioned in the previous section, some methods [4]–[6] have been proposed for annotating motion information in a video. Among them, the trajectory-based approach is probably the most popular. However, its unstable nature and high computation cost have discouraged its use as a representation/annotation tool. Furthermore, such schemes take the path formed by linking the centroids of a video object that appears across consecutive anchor frames. Therefore, if a user wants to retrieve the motion contributed by part of a video object, the trajectory-based representation scheme cannot provide a correct retrieval result. In view of this, we propose the direct usage of the motion vectors originally embedded in an MPEG bitstream to construct motion flows in a shot. Although the motion vectors do not always correspond to the "real motion" of objects in a video compared with the optical flow, they are relatively easy to derive.

Before constructing the motion flows, we must perform some preprocessing steps to compensate for camera motion. First, we produce a motion vector field between the last P-frame in the current GOP and the I-frame in the next GOP by using the B-frames between the P- and I-frame, as proposed in [4]. This yields a forward reference motion vector field between any two consecutive anchor-frames (I- and P- frames). It is well known that a motion vector field usually comprises camera motion, object motion, and noise. We assume that the global motion in a video is contributed primarily by camera motion. Thus, we use the following four-parameter global motion model, which is fast and valid for most videos [4], to estimate the camera motion from the motion vector field

$$\overrightarrow{MV_{cam}} = \begin{pmatrix} zoom & rotate \\ -rotate & zoom \end{pmatrix} \bullet \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} pan \\ tilt \end{pmatrix}. \quad (1)$$

Once the four parameters have been estimated, we can find the degree of correspondence between each pixel in the current frame and its counterpart in the previous frame. The foreground pixels in the current frame can be identified if their color changes significantly after the camera motion. In addition, information about local motion is also derivable if one subtracts the estimated camera motion from the original motion vector field. If several foreground pixels within a macroblock survive

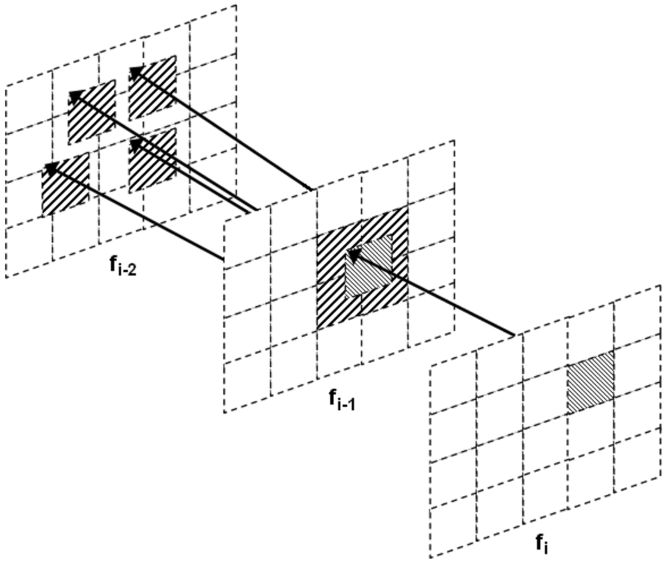


Fig. 2. Current motion vector and the four possible previous motion vectors that could be linked to it.

the erosion process, the local motion corresponding to the macroblock can be regarded as robust. Otherwise, we ignore the local motion, because it is probably from the background.

After the camera motion has been compensated for, we can extract the local motion fields from the bitstreams. Each field represents the movement of a macroblock between two consecutive anchor frames. However, a field is not usually continuous in the spatio-temporal domain, since motion vectors always start at regular places, but they could end anywhere. Thus, when we try to make the necessary connections between two local motion fields, each macroblock in the current frame may overlap with four macroblocks in the previous frame, as shown in Fig. 2. In other words, it is possible that several candidate local motions located in the previous frame could be connected to a local motion in the current frame. Therefore, we use the following three criteria to choose the most reliable candidate for the current local motion: 1) the consistency of motion direction; 2) the color distribution; and 3) the overlapping area between macroblocks in the previous and the current frames. Suppose LM_{Cur} is a local motion and C denotes the set of candidates of LM_{Cur} in the previous anchor frame. The most reliable candidate, LM_{ancestor}, is chosen according to the following rule:

$$LM_{\mathrm{ancestor}} = \arg\min_{m\in C} \|(\alpha\theta_m, \beta\phi_m, (1-\alpha-\beta)\varphi_m)\| \quad (2)$$

where θ, ϕ, and φ denote the angle formed by LM_{Cur} and a possible candidate m, the color histogram difference between macroblocks, and the overlapping area between macroblocks, respectively. Furthermore, all the parameters are normalized so that all values fall within the range [0,1], and α and β are weighted values that balance the influence of the three factors. Once the most reliable candidate of LM_{Cur} has been determined, we link the local motions acquired at different time spots to form a single motion flow for each macroblock.

Since a large moving object may cover several similar motion flows, we remove redundant flows and preserve one or more representative motion flows from all the similar motion flows. Initially, we select a motion flow a that has the longest duration in a shot. Suppose a starts from time t_i and ends at time t_j, and let $\{B\}$ be a set of motion flows whose start and end times are both within the duration of t_i and t_j. We remove a motion flow b, belonging to $\{B\}$, if it satisfies the following condition:

$$\frac{\sum_{t\in[t_s,t_e]} \|(a_t - a_{t_s}) - (b_t - b_{t_s})\|}{t_e - t_s} < \epsilon \quad (3)$$

where t_s and t_e are, respectively, the start and end times of motion flow b; a_t and b_t denote the spatial position of a and b at time t respectively; and a_{t_s} and b_{t_s} denote the spatial position of a and b at time t_s respectively. If the average of the relative spatial distance between a and b is lower than a given threshold ϵ, we consider b to be a subsegment of a. As b can be retrieved from a by a partial matching process, it is redundant and can be removed. We then consider a as the representative of the motion flow removed from $\{B\}$ and store it in the database. Next, we again select the motion flow with the longest duration from the remaining motion flows, and repeat the above process until all motion flows are either stored or removed. Here, we only consider motion flows longer than 3 s in order to avoid the effects of noise and reduce the size of the database.

Fig. 3(b) shows different representative motion flows of the video clip shown in Fig. 3(a). The original motion flows (with $\epsilon = 0$) are shown on the left-hand side, and the motion flows after applying the removal process using different thresholds are shown in the middle and on the right-hand side, respectively. It is obvious that the value of ϵ controls the number of representative motion flows. One may ask: what is the optimal value of ϵ that is most appropriate for different kinds of test videos. To analyze this issue, we used three test videos to conduct a series of experiments. Among the three videos, video 1 and video 2 (Fig. 4) were 30 min long and video 3 was 60 min in length. Fig. 4 shows three different curves. These three curves indicate the number of representative motion flows derived by using different ϵ values. According to the three curves, it is not difficult to explain why the ϵ value was set to 0.1 for all subsequent experiments. As we can see from the curves corresponding to video 1 and video 2, the total number of motion flows were 11968 and 7176, respectively, when the value of ϵ was 0. When the value of ϵ was set to 0.1, their corresponding number of motion flows dropped to 2578 and 2519, respectively. From this point, when one increased the value of ϵ to 0.2, 0.3, and 0.4, the total number of representative motion flows didn't drop significantly. A similar situation occured in the case of video 3. Therefore, we suggest to use $\epsilon = 0.1$ for all subsequent experiments. Since this issue is basically an ill-posed problem, we believe the best way for deriving an appropriate ϵ value is through a large number of experiments. In addition to the issue discussed above, we also found several reasons that may influence the curves shown in Fig. 4. First, the size of the moving objects in a video will influence the number of representative motion flows a lot. The larger the moving object is, the faster the number of representative motion flows decreased when one changes the ϵ value. Second, the complexity of a video also influence the number of representative motion flows. If a video is very complicated (a lot of object motion), then its corresponding number of motion flows will be larger.

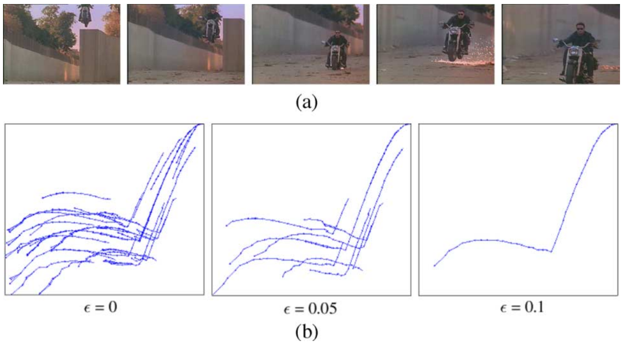


Fig. 3. (a) Original video clip. (b) Representative motion flows selected with threshold $\epsilon = 0$, 0.05, and 0.1.

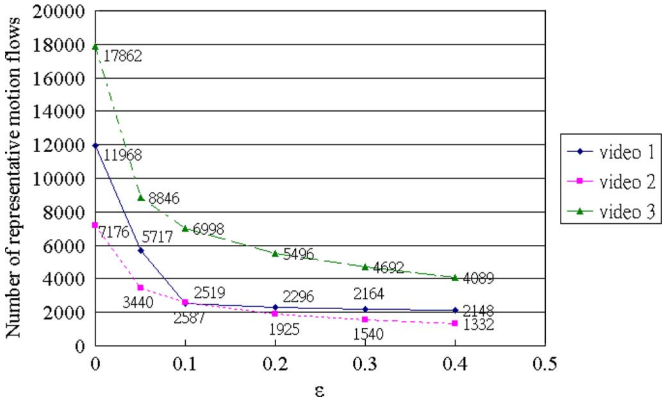


Fig. 4. Relationship between ϵ and the number of representative motion flows.

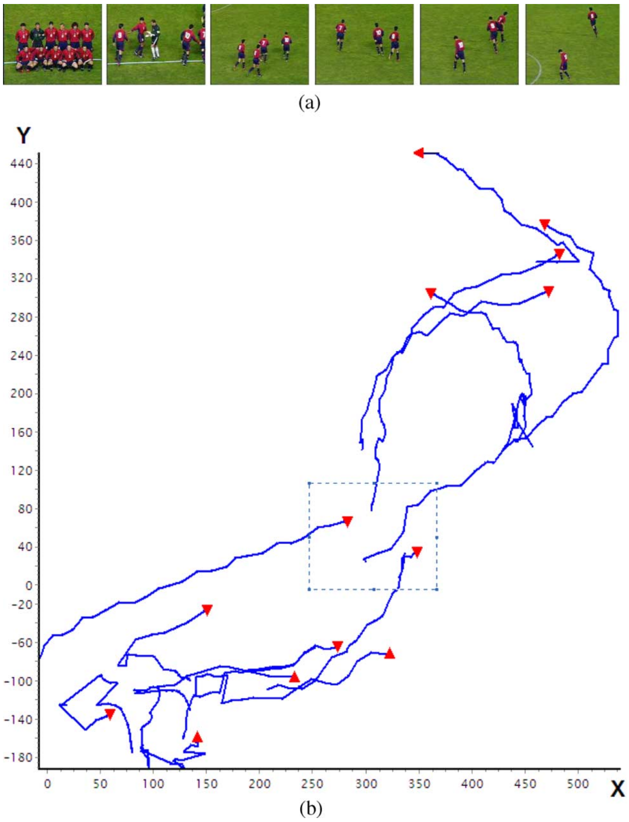


Fig. 5. (a) Original video clip. (b) Representative motion flows derived from the video sequence in (a).

Fig. 5 shows another example of representative motion flows derived from a video sequence containing multiple moving objects and global camera motion. Since the movement of each object is different, several representative motion flows are generated concurrently to represent the way the objects move. The extraction of motion flows is therefore much faster and easier than deriving a trajectory, but there still exist two problems when we construct motion flows from motion vectors. First, the occlusion of multiple moving objects can cause a macroblock to be intra-coded, which means we could miss motion information. Thus, a complete motion flow for each moving object may not be derived. The region surrounded by dotted lines in Fig. 5 illustrates the above situation. The sudden termination of motion flows is apparently caused by the occlusion of two of the football players. Second, since we remove the camera motion before extracting the local motion from a video, the motion flow will break off if a moving object stops temporarily.

III. Coarse-to-Fine Trajectory Comparison

Having constructed the motion flows from each shot, we propose a new algorithm that compares the degree of similarity between a query trajectory and the trajectories formed by the motion flows in a database. Since we are looking for "similar" clips in the database, some geometric transformation, such as scaling or translation, should be handled. Here, we do not consider rotation invariance because the direction, i.e., up-and-down or left-and-right, usually has semantic meaning in a video. For example, if one wants to query a "jump" motion, the trajectory should start from a lower position, pass through a higher position and then return to the lower position. Also, the issue of partial matching must be handled if a user provides an incomplete query. In the following, we propose a simple, but fast, algorithm for comparing two distinct trajectories (motion flows).

To reduce the time complexity and minimize the storage space required, we remove redundant points from a trajectory, leaving only a few necessary points to represent it. We then choose a famous top-down method—the Douglas-Peucker algorithm [3]—to select the necessary control points from the trajectory. The algorithm starts by using a straight line (called the anchor line) to connect the start and end points of the trajectory. Once the perpendicular distance between any intermediate point and the anchor line is larger than a given threshold, the trajectory is split into two segments via the farthest intermediate point. The process continues until all the perpendicular distances are smaller than a pre-set threshold. Finally, the chosen intermediate points and the two end points are reserved as the control points of the trajectory.

Traditionally, researchers have used (x, y, t) to denote the position of a control point on a trajectory in a spatio-temporal domain. Unlike conventional approaches, we use six positive real numbers $(x^+, x^-, y^+, y^-, d, t)$ to represent a control point on a trajectory, where d denotes the cumulative length of the trajectory from the first control point to the current control point; and $+/-$ denotes the cumulative positive/negative movement along the x- or y-axis from the first control point to the current control point. Now, let Q and D be the trajectories of the query and a model in the database, respectively. In a QBS case, we normalize the length of both trajectories into a unit length before comparing them. This guarantees the requirement of scale invariance. Therefore, the parameters d, x^+, x^-, y^+ and y^- of each control point on the two trajectories have to be normalized by dividing them by the length of Q and D, respectively.

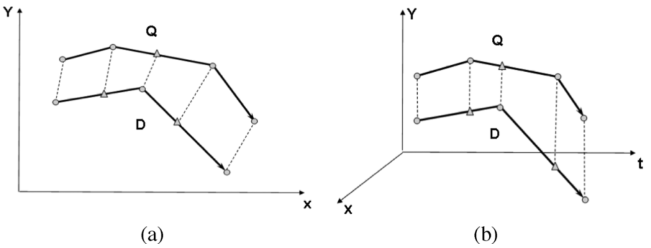
According to the six parameters of a control point, d and t are utilized to make a fair comparison. We align both Q and D by calculating the length d or duration t from the first control point. For each control point on $Q(D)$, we interpolate a corresponding point that has the same cumulative length onto $D(Q)$. In a query-by-sketch (QBS) video retrieval system, the "d" value is used as the basis to conduct the alignment task, because we only consider the similarity between Q and D in the spatial domain, as shown in Fig. 6(a). On the other hand, the "t" value plays a crucial role when a video retrieval system incorporates the query-by-example (QBE) approach, as shown in Fig. 6(b). The control points and the corresponding points are labeled by circles and triangles, respectively. In this scenario, the insertion of corresponding points on Q and D is dependent on either "d" or "t". Now, for each control point on the trajectory $Q(D)$, we can interpolate a corresponding point located on $D(Q)$. Assume the total number of control points and their corresponding points located on Q and D are both N. Let $Q' : \{Q'_1, Q'_2, \ldots, Q'_N\}$ and $D' : \{D'_1, D'_2, \ldots, D'_N\}$ be the set of points (including the control points and inserted corresponding points) located on Q and D, respectively. We call the points in set Q' and D' "check points", each of which can be represented by (x^+, x^-, y^+, y^-) in the spatial domain. In order to compare two arbitrary trajectories, we define a metric as follows:

$$EstDist_{i,j}^{Q',D'} = \left|\left(Q'_j - Q'_i\right) - \left(D'_j - D'_i\right)\right| \bullet \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix} \quad (4)$$

where i and j $(i < j)$ denote the ith and the jth check points of two partial trajectories located on Q' and D', respectively; $Q'_j - Q'_i$ and $D'_j - D'_i$ represent the difference between the ith and the jth check points of Q' and D', respectively; and $(Q'_j - Q'_i) - (D'_j - D'_i)$ is a 1×4 vector, and its subsequent part in (4) is a 4×2 matrix. We take the absolute value of each element of $(Q'_j - Q'_i) - (D'_j - D'_i)$ before the matrix operation is executed, because we want to accumulate all movements including negative ones. Therefore, the operation on the right-hand side of (4) generates a 1×2 vector. Also, $\|EstDist_{i,j}^{Q',D'}\|$ is only a rough estimation of the distance between two partial trajectories (each trajectory runs from the i-th check point to the j-th check point) located on Q' and D', respectively. With the above distance metric, we can define the total distance metric between Q and D as follows:

$$Dist(Q, D) = \sum_{i=1}^{N-1} \left\| EstDist_{i,i+1}^{Q',D'} \right\|. \quad (5)$$

The norm defined in (5) is an L_2-norm.



(a) (b)

Fig. 6. Alignment task in (a) QBS and (b) QBE.

The advantage of the proposed representation scheme is that we do not really need to compare the check points pair by pair. One factor to be noted, however, is that all the elements that form the vector of a check point on a trajectory are positive and the magnitude of each one is accumulated from the beginning. Therefore, if we choose an intermediate check point Q'_k in Q' and its corresponding check point D'_k in D', we can be sure that

$$\left\| EstDist_{i,j}^{Q',D'} \right\| \le \left\| EstDist_{i,k}^{Q',D'} \right\| + \left\| EstDist_{k,j}^{Q',D'} \right\|. \quad (6)$$

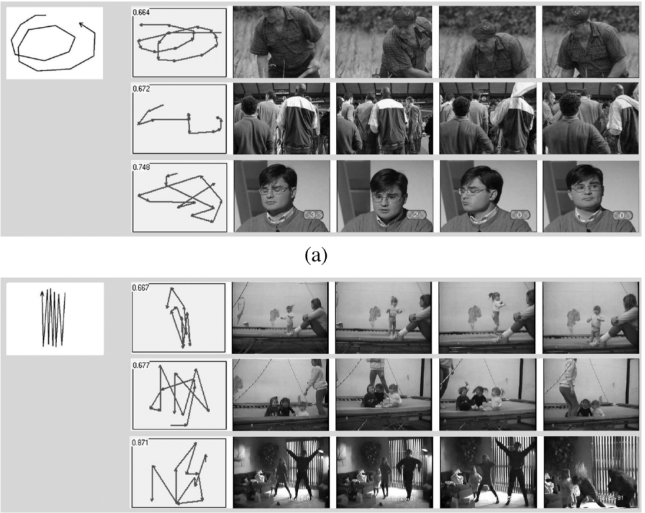
Equation (6) tells us that a coarse-to-fine search strategy is feasible for a trajectory-based query. In the first step of the comparison between Q' and D', we simply check the value of $\|EstDist_{1,N}^{Q',D'}\|$. This step only needs to consider four check points Q'_1, D'_1, Q'_N, and D'_N. Since the value of $Dist(Q, D)$ must be equal to or larger than that of $\|EstDist_{1,N}^{Q',D'}\|$, we can quickly determine that trajectory D is not similar to Q if the returned value of $\|EstDist_{1,N}^{Q',D'}\|$ is larger than a predefined threshold δ.

Once the value of $\|EstDist_{1,N}^{Q',D'}\| < \delta$, we seek the second check points on Q' and D', respectively, by checking Q_2 and D_2. If Q_2 is chosen as Q'_2, we can insert D'_2 to the right position between D_1 and D_2 and vice versa. Furthermore, Q' and D' can be divided into four subtrajectories by Q'_2 and D'_2. In this case, we only compute the sum of $\|EstDist_{1,2}^{Q',D'}\|$ and $\|EstDist_{2,N}^{Q',D'}\|$ as the distance between the subtrajectories. If the distance between two distinct subtrajectories is still larger than a predefined threshold δ, D will be filtered out. Otherwise, we insert Q'_3 and D'_3 to further compute $\|EstDist_{2,3}^{Q',D'}\|$ and $\|EstDist_{3,N}^{Q',D'}\|$. The above newly computed distances replace the value of $\|EstDist_{2,N}^{Q',D'}\|$, and the process is executed repeatedly until the computed distance is larger than δ, or there are no more intermediate check points within each subtrajectory.

IV. Experiment Results

To test the effectiveness of our method, we conducted two sets of experiments: one on a video database generated from a compressed 3-hour MPEG-7 test video sequence in MPEG-1 format, and the other on a compressed 4-hour MPEG-2 surveillance video taken at eight different locations.

The 3-h test video contained more than one thousand shots of different subjects, including news, sports, documentaries, and home videos. In this set of experiments, we extracted approximately six thousand motion flow-based trajectories from the one thousand video shots based on a preset threshold $\epsilon = 0.1$. If ϵ is selected as an improper high value, most of the motion flows will be eliminate and some representable motion would be no longer exist in the database. Oppositely, if we select a lower value for



(a)

(b)

Fig. 7. (a) Example of the retrieval result using query-by-sketch. (b) Another example of the same process.

ϵ, more motion flows will be kept in the database and thus help increase the recall of query result. But redundant motion flows will also increase the query time and decrease the precision of query result at the same time. On the other hand, the weight values of α and β in (2) were set 0.2 and 0.3, respectively, in all experiments. Thus, the average number of motion-flow trajectories extracted from each shot was six. To simplify the experimental process, we only used the longest trajectory in each shot for comparison. We consider this reasonable because, in most cases, the constituent trajectories of a shot have similar tendencies, and the longest one is usually the most representative. We used our specially designed user interface to sketch a trajectory as an input query. The system computed all related data from the query and compared it with the data extracted from the longest trajectory of each shot. In the experiments, the average response time for a query was less than one second. Fig. 7 illustrates two sets of experiment results, both derived by the QBS approach. In Fig. 7(a), the video database is queried by a cyclic motion and the three most similar candidates are retrieved and listed from top to bottom according to their degree of similarity. The first row of the image sequence shows a farmer using a harvesting tool; the second row shows the movement of basketball players; and the third row shows a man shaking his head. All three retrieved image sequences have a cyclic motion. The advantage of this approach is that our algorithm does not need to segment a moving object in a video clip. To demonstrate the superiority of our approach, suppose we segment the man in the third retrieved image sequence in Fig. 7(a). We cannot detect a cyclic motion, because the trajectory formed by linking the centroids of a video object located in consecutive frames is quite steady. However, our approach can retrieve the cyclic motion caused by the movement of the face. In Fig. 7(b), we show another example in which we adopted a repeated jump motion as the query. The first and second sequences both show the movement of girls jumping on a spring mattress, while the third sequence shows two people dancing in a room. Fig. 8 shows how the performance varies with QBE and QBS in the Precision-Recall diagram, which is a common way for evaluating performance in visual information retrieval. "Precision" measures how precise the search results are, and "Recall" measures the completeness of available relevant results. They are defined as

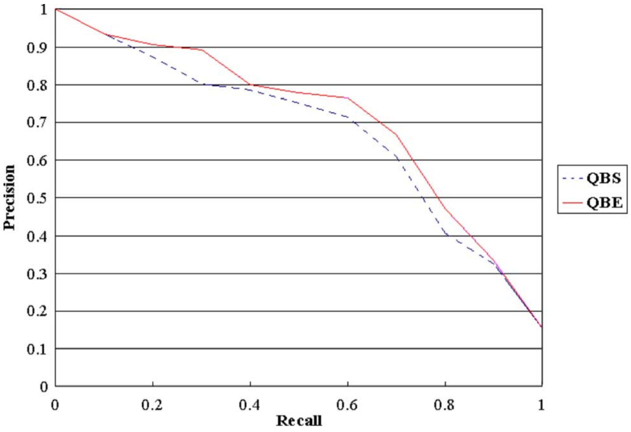


Fig. 8. Precision-Recall diagram for QBE and QBS.

$$Recall = \frac{relevant\ correctly\ retrieved}{all\ relevant}$$
$$Precision = \frac{relevant\ correctly\ retrieved}{all\ retrieved}$$

We manually divided all the test shots into five categories (jumping, cyclic moving, turning, horizontal and vertical moving) according to the style of object motion within the clips. Each category contained 20 to 40 shots. If a shot contained more than two different motion styles, it was not considered in our experiment. According to Fig. 8, it is obvious that the performance of QBE is always a little bit better than that of QBS. In real-world applications, the QBE or QBS retrieval strategy can be applied to surveillance systems. In the future, it is possible to collect hundreds of thousands of trajectory-based video events from different locations. The QBS or QBE search strategy does not intend to identify a single event. Instead, it is intended for retrieving a small set of similar trajectories from the database. Since the search space is significantly reduced, we can apply a more precise (or detailed) strategy to do finer search.

Fig. 9 shows another experiment on the relationship between the response time and the number of trajectories. Each trajectory contains more than 20 control points in this experiment. Since the QBE process is always achieved by selecting a candidate from the result of QBS, both QBE and QBS have very similar performance on query time. The response times of the above video retrieval queries were very short due to our specially designed comparison procedure. Clearly, our system can retrieve reasonably close motion from a video database through a quick sketch. However, if a sketch is too simple, our system will respond with many hard-to-judge retrieval results. Fig. 10 illustrates an example of this problem. Suppose we query the video database by a simple straight motion from right to left. Several similar candidates will be retrieved, so we will need to spend time browsing and looking for the video clips we requested. This is to be expected, because the less information we provide to the system, the less specific the solutions we will get.

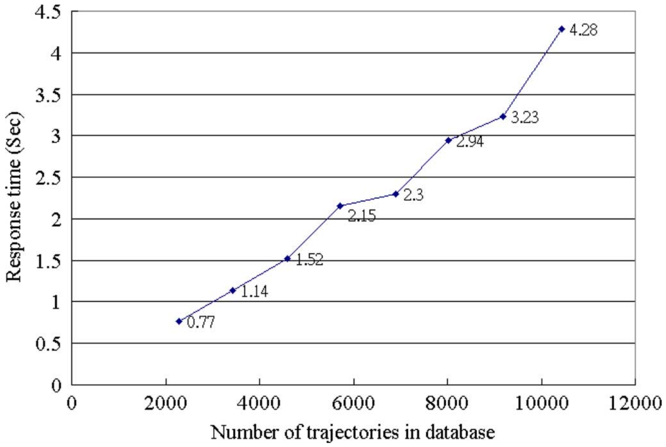


Fig. 9. Relationship between the response time and the number of trajectories.

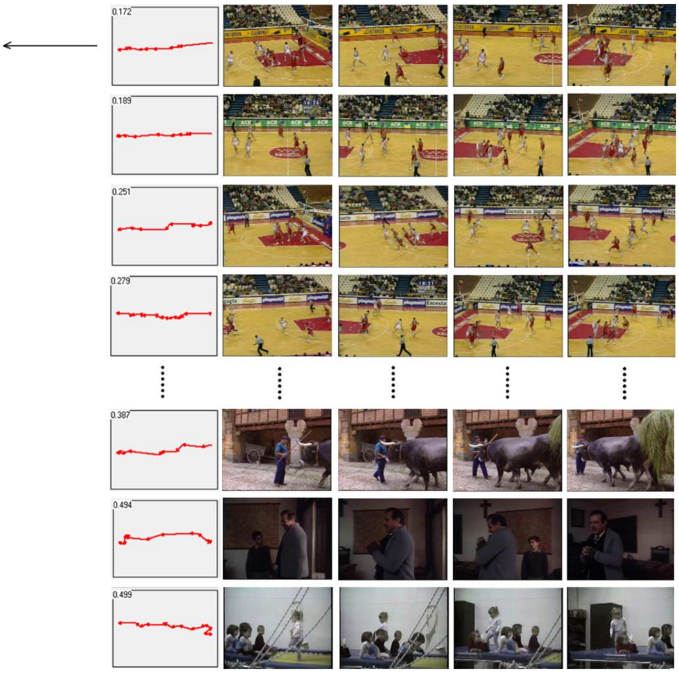


Fig. 10. Retrieval results of simple right-to-left-motions.

Since the query shown in Fig. 10 contains very little information, we can only retrieve very "rough" results. Therefore, if we have more information about the trajectory of a motion, we can draw a "closer" (more complicated) query to retrieve video clips from the database. The retrieved results will then be much closer to the query, as shown by the cases in Fig. 7(a) and Fig. 7(b).

In the first set of experiments, our goal was to refine our approach and demonstrate the flexibility of the proposed system. Then, in the second set of experiments, we narrowed down the domain and applied it to surveillance systems. We took a compressed 4-h MPEG-2 test video from eight different locations. Fig. 11 shows the background views of the eight locations. From the four-hour video, we extracted 108 video clips that contained motion. We call these clips with motion "events." To simplify the QBS process, we used one of the eight locations as the background to assist in drawing a query sketch. The left-hand side of Fig. 12 shows a query sketch of a man climbing a stairway. Among the 108 events, the top three retrieved events were all of the man climbing the stairway at different times in the sequence.

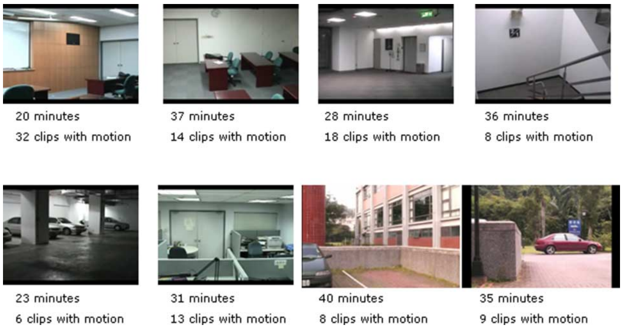


Fig. 11. Surveillance video clips taken at eight different locations.

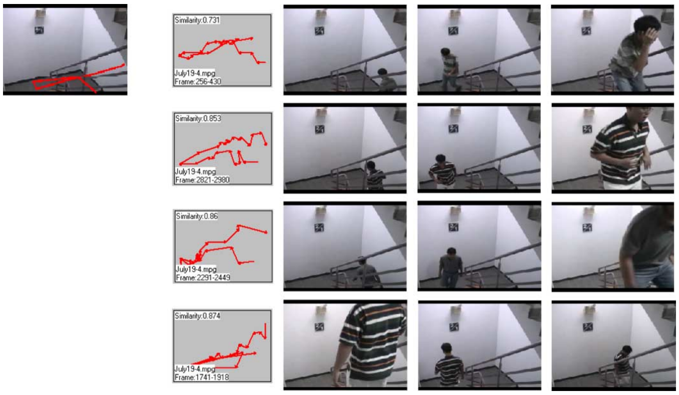


Fig. 12. Example of the retrieval result using query-by-sketch in a stairway scenario. The top four retrieved results (out of 108 video clips) are all related video clips.

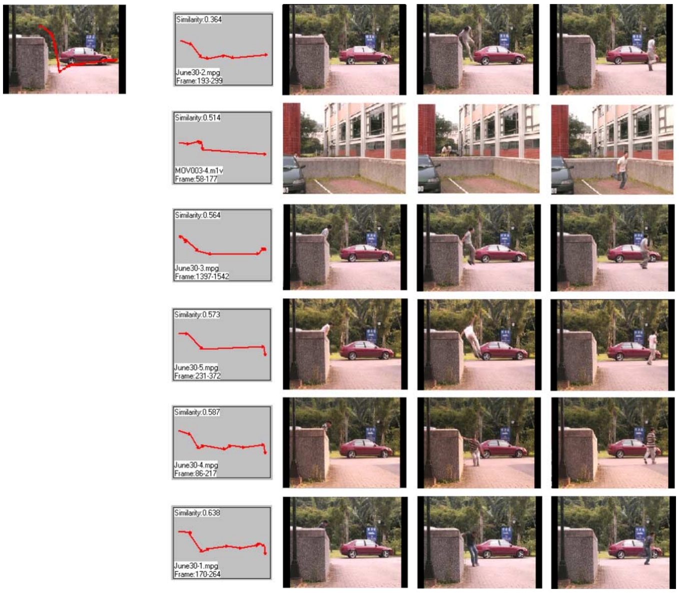


Fig. 13. Example of retrieval results using query-by-sketch on frames from an in-house video surveillance system. Except for the frames in the second row, all the retrieved results (out of 108 video clips) are related to the query trajectory. The trajectory of the clips in the second row is included because it has a very similar shape to those of the other retrieved results.

The fourth retrieved sequence was a descending the stairway event, which was retrieved because of its high degree of similarity to the top three retrieved events. In another experiment, we drew a climbing-wall trajectory with the help of the user interface (as shown in Fig. 13). From the top six retrieved events, it is obvious that the first, third, fourth, fifth, and sixth all relate to the query. The second retrieved event happened at another location; however, its corresponding trajectory was very close to the query sketch.

V. Conclusion

In this paper, we have proposed a new representation scheme for local motion in videos. The major advantage of our approach is that it does not need to perform object extraction and tracking in the representation process. We use the information about motion vectors in an MPEG bitstream directly to generate some trajectory-like motion flows to describe local motion. Since motion vectors are uniformly distributed in each video frame, we can process a case of multiple moving objects in a shot. We have also proposed a new matching algorithm that retrieves similar trajectories more accurately. However, there are still some problems in the process of motion flow generation that must be resolved. First, information about motion vectors is not always reliable. The worst case is when there are multiple moving objects in a macroblock. As a macroblock is usually intra-coded, it loses motion information in the bitstream. Second, since we remove camera motion and extract local motion from a video, if a moving object stops suddenly in a frame, the motion flow(s) will break off, even if the object moves again later. We will address these problems in our future work.

References

[1] M. Flickner *et al.*, "Query by image and video content: The QBIC system," *IEEE Comput. Mag.*, vol. 28, pp. 23–32, Sep. 1995.

[2] 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, Sep. 1998.

[3] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *The Canadian Cartographer*, vol. 10, no. 2, pp. 112–122, 1973.

[4] R. Wang and T. Huang, "Fast camera motion analysis in MPEG domain," in *Proc. Int. Conf. Image Processing (ICIP)*, Oct. 1999, vol. 3, pp. 691–694.

[5] S. Dagtas, W. Al-Khatib, A. Ghafoor, and R. L. Kashyap, "Models for motion-based video indexing and retrieval," *IEEE Trans. Image Process.*, vol. 9, no. 1, pp. 88–101, 2000.

[6] R. Fablet, P. Bouthemy, and P. Perez, "Nonparametric motion characterization using causal probabilistic models for video indexing and retrieval," *IEEE Trans. Image Process.*, vol. 11, no. 4, pp. 393–407, Apt. 2002.

[7] A. Pentland, R. W. Picard, and S. Sclaroff, "Photobook: Content-based manipulation of image databases," *Int. J. Comput. Vis.*, vol. 18, no. 3, pp. 233–254, 1996.

[8] J. R. Smith and S. F. Chang, "VisualSEEk: A fully automated content-based image query system," in *ACM Multimedia Conf.*, Nov. 1996, pp. 87–98.

[9] A. Hamrapur, A. Gupta, B. Horowitz, C. F. Shu, C. Fuller, J. Bach, M. Gorkani, and R. Jain, "Virage video engine," in *SPIE Proc. Storage and Retrieval for Image and Video Databases V*, San Jose, CA, Feb. 1997, pp. 188–197.

[10] Y. Deng and B. S. Manjunath, "NeTra-V: Toward an object-based video representation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 5, pp. 616–627, Sept. 1998.

[11] Y.-F. Ma and H.-J. Zhang, "Motion texture: A new motion based video representation," in *Proc. 16th Int. Conf. Pattern Recognition*, Aug. 11–15, 2002, vol. 2, pp. 548–551.

[12] D.-J. Lan, Y.-F. Ma, and H.-J. Zhang, "A novel motion-based representation for video mining," in *Proc. Int. Conf. Multimedia and Expo*, Jul. 6–9, 2003, vol. 3, pp. 469–472.

[13] B. S. Manjunath, P. Salembier, and T. Sikora, Introduction to MPEG-7: Multimedia Content Description Interface Jun. 2002.

[14] Y. Tsaig and A. Averbuch, "Automatic segmentation of moving objects in video sequences: A region labeling approach," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 7, pp. 597–612, Jul. 2002.

[15] V. Mezaris, I. Kompatsiaris, and M. G. Strintzis, "Video object segmentation using Bayes-based temporal tracking and trajectory-based region merging," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 6, pp. 782–795, Jun. 2004.

[16] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts, and shadows in video streams," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 25, no. 10, pp. 1337–1342, Oct. 2003.

[17] C. Rao, A. Gritai, M. Shah, and T. Syeda-Mahmood, "View-invariant alignment and matching of video sequences," in *Proc. 9th IEEE Int. Conf. Computer Vision*, Oct. 13–16, 2003, vol. 2, pp. 939–945.

[18] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Proc. 18th Int. Conf. Data Engineering*, 26 Feb.–1 Mar. 2002, pp. 673–684.

[19] [Online]. Available: http://www.video.google.com/

[20] [Online]. Available: http://www.search.yahoo.com/

[21] C.-W. Su, H.-Y. M. Liao, K.-C. Fan, C.-W. Lin, and H.-R. Tyan, "A motion-flow-based fast video retrieval system," in *Proc. 7th ACM SIGMM International Workshop on Multimedia Information Retrieval*, Singapore, Nov. 10–11, 2005.

[22] C.-W. Su, H.-Y. M. Liao, H.-R. Tyan, K.-C. Fan, and L.-H. Chen, "A motion-tolerant dissolve detection algorithm," *IEEE Trans. Multimedia*, vol. 7, no. 6, pp. 1106–1113, Dec. 2005.

[23] C.-C. Shih, H.-R. Tyan, and H.-Y. M. Liao, "Shot change detection based on the Reynolds transport theorem," *Lecture Notes in Computer Science*, vol. 2195, pp. 819–824.

[24] R. Venkatesh Babu and K. R. Ramakrishnan, "Compressed domain video retrieval using object and global motion descriptors," *Multimedia Tools and Applic.*, vol. 32, no. 1, pp. 93–113, Jan. 2007.

Chih-Wen Su received the B.S. degree in mathematics and the M.S. degree in computer science, both from Fu-Jen University, Hsinchuang, Taiwan, R.O.C., in 1999 and 2001, respectively, and the Ph.D. degree in computer science and information engineering from National Central University, Chung-Li, Taiwan, in 2006.

He is currently a Postdoctoral Research Fellow with Academia Sinica, Taipei, Taiwan. His research interests are in image and video analysis and content-based indexing and retrieval.

Hong-Yuan Mark Liao (SM'01) received the B.S. degree in physics from National Tsing-Hua University, Hsinchu, Taiwan, R.O.C., in 1981, and the M.S. and Ph.D. degrees in electrical engineering from Northwestern University, Evanston, IL, in 1985 and 1990, respectively.

He was a Research Associate with the Computer Vision and Image Processing Laboratory at Northwestern University during 1990–1991. In July 1991, he joined the Institute of Information Science, Academia Sinica, Taipei, Taiwan, as an Assistant Research Fellow. He was promoted to Associate Research Fellow and then Research Fellow in 1995 and 1998, respectively. From August 1997 to July 2000, he served as the Deputy Director of the institute. From February 2001 to January 2004, he was the Acting Director of the Institute of Applied Science and Engineering Research. He is jointly appointed as a Professor of the Computer Science and Information Engineering Department of National Chiao-Tung University. His current research interests include multimedia signal processing, video surveillance, and content-based multimedia retrieval.

Dr. Liao is the Editor-in-chief of the *Journal of Information Science and Engineering*. He is on the Editorial Board of the *International Journal of Visual Communication and Image Representation*, and the *EURASIP Journal on Applied Signal Processing*. He was an Associate Editor of the IEEE Transactions on Multimedia during 1998–2001. He received the Young Investigators' Award from Academia Sinica in 1998, the Excellent Paper Award from the Image Processing and Pattern Recognition society of Taiwan in 1998 and 2000, the Distinguished Research Award from the National Science Council of Taiwan in 2003, and the National Invention Award in 2004. He served as the Program

Chair of the International Symposium on Multimedia Information Processing (ISMIP'97), the Program Co-Chair of the Second IEEE Pacific-Rim conference on Multimedia (2001), the Conference Co-Chair of the Fifth IEEE International Conference on Multimedia and Exposition (ICME), and as Technical Co-Chair of the IEEE ICME (2007).

Hsiao-Rong Tyan received the B.S. degree in electronic engineering from Chung-Yuan Christian University (CYCU), Chung-Li, Taiwan, R.O.C., in 1984 and the M.S. and Ph.D. degrees in computer science from Northwestern University, Evanston, IL, in 1987 and 1992, respectively.

She is an Associate Professor with the Department of Information and Computer Engineering, CYCU, where she currently conducts research in the areas of computer networks, computer security, and intelligent systems.

Chia-Wen Lin (S'94-M'00-SM'04) received the M.S. and Ph.D. degrees in electrical engineering from National Tsing Hua University (NTHU), Hsinchu, Taiwan, R.O.C., in 1992 and 2000, respectively.

Since August 2007, he has been an Associate Professor with the Department of Electrical Engineering, NTHU. He was with the Department of Computer Science and Information Engineering, National Chung Cheng University (CCU), Taiwan, during 2000–2007. Prior to joining academia, he worked for the Computer and Communications Research Laboratories (CCL), Industrial Technology Research Institute (ITRI), Hsinchu, Taiwan, during 1992–2000, where his final post was Section Manager. From April 2000 to August 2000, he was a Visiting Scholar with Information Processing Laboratory, Department of Electrical Engineering, University of Washington, Seattle. From July 2002 to August 2002, he was a Visiting Professor with Microsoft Research Asia, Beijing, China. He has authored or coauthored over 70 technical papers. He holds a dozen patents with more pending. His research interests include video coding and video networking.

Dr. Lin is a member of the Visual Signal Processing and Communications Technical Committee, the Multimedia Systems and Applications Technical Committee, and the Circuits and Systems Society of the IEEE. He is General Co-Chair of the First International Workshop on Multimedia Analysis and Processing (IMAP), to be held in Hawaii in August 2007. He served as a Guest Editor for the Special Issue on Video Adaptation for Heterogeneous Environments of the *EURASIP Journal on Advances in Signal Processing*. He is a co-author of the paper that won the Young Investigator Award at SPIE VCIP 2005. He received the Young Faculty Awards presented by CCU in 2005 and the Young Investigator Awards presented by National Science Council, Taiwan, in 2006.

Duan-Yu Chen received the B.S. degree in computer science and information engineering from National Chaio-Tung University (NCTU), Taiwan, R.O.C., in 1996, the M.S. degree in computer science from National Sun Yat-Sen University, Taiwan, in 1998, and the Ph.D. degree in computer science and information engineering from NCTU in 2004.

He is currently a Postdoctoral Research Fellow with Academia Sinica, Taipei, Taiwan. His research interests include computer vision, video signal processing, content-based video indexing and retrieval, and multimedia information system.

Kuo-Chin Fan (M'88) was born in Hsinchu, Taiwan, R.O.C. in 1959. He received the B.S. degree in electrical engineering from National Tsing Hua University, Hsinchu, Taiwan, R.O.C., in 1981 and the M.S. and Ph.D. degrees from the University of Florida, Gainesville, in 1985 and 1989, respectively.

In 1983, he joined the Electronic Research and Service Organization (ERSO), Taiwan, as a Computer Engineer. From 1984 to 1989, he was a Research Assistant with the Center for Information Research, the University of Florida. In 1989, he joined the Institute of Computer Science and Information Engineering, National Central University, Chung-Li, Taiwan, where he became a Professor in 1994. From 1994 to 1997, he was the Chairman of the department. He was the Director of the Software Research Center from 1998 to 2000 and Director of the Computer Center from 1999 to 2003, National Central University. Currently, he is the Director of the Communication Research Center. He was the President of the Image Processing and Pattern Recognition Society-Taiwan from 2002 to 2004 and President of the governing board of the International Association of Pattern Recognition (IAPR) starting in 2002. His current research interests include biometrics identification, video surveillance, and intelligent transportation system.

Dr. Fan received three consecutive Outstanding Researcher Awards granted by the National Science Council, Taiwan, during 1998-2000.