3-2003

Motion analysis and segmentation through spatio-temporal slices processing

Chong-wah NGO
Singapore Management University, cwngo@smu.edu.sg

Ting-Chuen PONG

Hong-Jiang ZHANG

Motion Analysis and Segmentation Through Spatio-Temporal Slices Processing

Chong-Wah Ngo, *Member, IEEE*, Ting-Chuen Pong, and Hong-Jiang Zhang, *Senior Member, IEEE*

***Abstract*—This paper presents new approaches in characterizing and segmenting the content of video. These approaches are developed based upon the pattern analysis of spatio-temporal slices. While traditional approaches to motion sequence analysis tend to formulate computational methodologies on two or three adjacent frames, spatio-temporal slices provide rich visual patterns along a larger temporal scale. In this paper, we first describe a motion computation method based on a structure tensor formulation. This method encodes visual patterns of spatio-temporal slices in a tensor histogram, on one hand, characterizing the temporal changes of motion over time, on the other hand, describing the motion trajectories of different moving objects. By analyzing the tensor histogram of an image sequence, we can temporally segment the sequence into several motion coherent subunits, in addition, spatially segment the sequence into various motion layers. The temporal segmentation of image sequences expeditiously facilitates the motion annotation and content representation of a video, while the spatial decomposition of image sequences leads to a prominent way of reconstructing background panoramic images and computing foreground objects.**

***Index Terms*—Motion segmentation, spatio-temporal slices, tensor histogram.**

I. Introduction

In the past decade, theories for acquiring, manipulating, transmitting and storing video data have been successfully developed and applied. Nevertheless, the methodology for annotating and representing visual information is still in its infancy. The objective of this paper is to present a new way of video parsing for motion annotating and segmentation through the pattern analysis of image slices in a spatio-temporal volume. There are three major issues that need to be addressed toward this goal: video partitioning, video characterization and video segmentation (see Fig. 1 for an illustration). By integrating these components, video representation, clustering and retrieval can be realized in a concrete way. The novelty of these approaches lies in the utilization of patterns in spatio-temporal slices, which on one hand is effective in exploring temporal events along a larger temporal scale; on the other hand, this method is efficient since possibly only few selected slices are needed to be processed for certain applications.

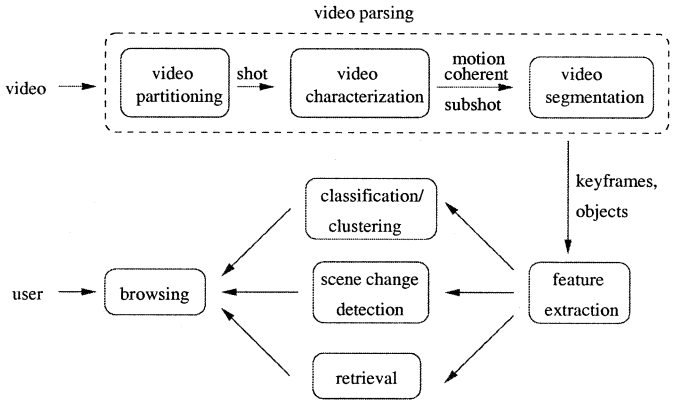


Fig. 1. Content-based video analysis.

Analysis of spatio-temporal slices for computational vision tasks has been proposed since 1985 [1]. Previous works include visual motion model [1], [25], epipolar plane image analysis [4], video tomography [2], [30], texture modeling [13], periodicity analysis [14], camera work analysis [12], and monitoring and surveillance applications [11]. The findings from this paper contribute to the areas of studies the temporal motion characterization and segmentation, spatial motion (background versus foreground) segmentation, formulated directly on the pattern analysis of spatio-temporal slices. Our work on video partitioning can be found in [16]–[18]. These proposed works are particularly useful for video representation since qualitative, rather than quantitative information, which provides essential cues for describing the visual world, can be acquired in an inexpensive manner.

A. Applications

Our approach is mainly targeted for the content-based video representation and retrieval. Fig. 1 illustrates the major flow of content-based video analysis. A video is first *partitioned* into shots, where each shot is an uninterrupted segment of image frames with continuous camera motions. Since a shot may have more than one camera motion, these shots are further *temporally segmented* into motion coherent subshots through video characterization. Each subshots is *characterized* according to its camera motion. A major advantage of this step is that each subshot can be represented compactly by a few selected or newly constructed keyframes through the annotated motion. For instance, a sequence with camera panning is well summarized if a new image can be formed to describe the panoramic view of

Manuscript received December 1, 2000; revised October 28, 2002. This work was supported in part by RGC under Grants HKUST661/95E, HKUST6072/97E, and CityU1072/02E. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Christine Guillemot.

C.-W. Ngo is with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong (e-mail: cwngo@cs.cityu.edu.hk).

T.-C. Pong is with the Department of Computer Science, The Hong Kong University of Science and Technology, Kowloon, Hong Kong (e-mail: tcpong@cs.ust.hk).

H.-J. Zhang is with Microsoft Research Asia, Beijing 100080, China (e-mail: hjzhang@microsoft.com).

Digital Object Identifier 10.1109/TIP.2003.809020

TABLE I
VIDEO REPRESENTATION THROUGH KEYFRAME SELECTION AND CONSTRUCTION

Motion Type	Action
static	select one frame
pan or tilt	construct a new panoramic image
zoom	select the first and last image
multiple motion (e.g., pan + static)	segment foreground and background scenes
indeterministic	select one frame

a scene; a sequence with camera zoom is well represented by selecting two frames before and after the zoom. Table I summarizes this motion-driven scheme in representing the content of subshots.

In video segmentation, we focus on the issues of analyzing the subshots that contain multiple motion. In our approach, these subshots are *spatially segmented* into foreground and background objects. For effective video representation and retrieval, the background objects can be utilized by a scene change detection algorithm to group shots that are taken place in the same site [19]; the foreground objects can be used for video objects clustering and retrieval [20].

In this paper, we refer *video characterization* as the process of temporally segmenting shots into finer subunits and simultaneously characterizing these subunits according to camera motions, and *video segmentation* as the process of spatially segmenting subshots that are composed of multiple motion into foreground and background objects.

B. Related Works

Previous works on temporal motion segmentation include [5] and [12]; works on spatial motion segmentation include [10], [24],and [31]. Bouthemy *et al.* [5] employed the affine motion parameters to describe dominant camera motions; Irani and Anandan [10] discussed various motion models to annotate and represent videos; while Wang and Adelson [31] and Sawhney and Ayer [24] proposed the motion-based decomposition of videos to describe the background and foreground scenes. Most of these approaches are based on the iterative motion parameter estimation from two adjacent frames. It is generally expected that better results can be acquired if more frames are taken into account at the expense of computational time. Perhaps the most similar work to our proposed approach is by Joly and Kim [12] who employed Hough transform to detect lines in spatio-temporal slices. Nevertheless, they do not address the work on spatial motion segmentation. They only select two orthogonal slices for camera motion analysis, which in general do not provide sufficient clues for motion annotation.

Broadly we can categorize the works on spatial motion segmentation as sequential motion estimation [3], [9] and simultaneous motion estimation [6], [24], [31]. The former category begins by computing a dominant motion and then removing pixels corresponding to that dominant motion. The process is iteratively done until a terminal condition is met. In contrast, the latter category allows multiple motion models to simultaneously compete for the support of pixels and these pixels in turn influence the estimation of model parameters. The process is also iteratively done until an objective function is optimized. Typical works include the clustering of optical flow proposed by Wang and Adelson [31], robust M-estimation and minimum description length (MDL) framework by Darrell and Pentland [6], EM algorithm and MDL framework by Sawhney and Ayer [24].

Our work can be categorized under the sequential motion estimation. The proposed techniques are mainly devoted to peculiar sequences involving a dominant motion which can be identified with the background motion. The major difference with other approaches is that instead of analyzing and propagating the results from one frame to another, our approach measures the motion information from all frames of a shot and then determines the dominant motion. Unlike [3] and [9], the proposed approach will work even in the absence of an obvious dominant motion. After a dominant motion is detected, background subtraction and color back-projection is performed to estimate the secondary motion. An assumption is that the motion should contain a certain degree of smoothness such that the holes left in a background image can be minimized.

C. Plan of the Paper

This paper is organized as follows. Section II conducts a close examination on the patterns of spatio-temporal slices which have not yet been thoroughly studied in literature. This yields an enhanced understanding on how to utilize these patterns for motion analysis and segmentation. Section III proposes a two-dimensional (2-D) tensor histogram computation method to represent and analyze the spatio-temporal patterns of slices. The trajectories which appear in tensor histograms can describe both camera and object motions. Subsequently, Section IV describes an algorithm to temporally track and segment these motion trajectories for motion characterization. Section V presents two different approaches for spatial layer segmentation. Approach I exploits the similarity among slices to partition a spatio-temporal volume into motion layers, with each layer being modeled by a tensor histogram. However, this approach cannot handle scenes with cluttered background. As a consequent, Approach II is introduced to judiciously decompose the distribution of a tensor histogram into motion layers. Initially, the background layer of each image frame is registered and mosaicked to form a panoramic image. The foreground objects can then be detected expeditiously. Finally, Section VI summarizes our proposed works and describes future research directions.

II. OUR METHODOLOGY

If we view a video as an image volume with (x, y) image dimension and t temporal dimension, the spatio-temporal slices are a set of 2-D images in a volume with one dimension in t, and the other in x or y, for instance. One example is given in Fig. 2; the horizontal axis is t, while the vertical axis is x. A spatio-temporal slice, by first impression, is composed of color and texture components. On one hand, the discontinuity of color and texture

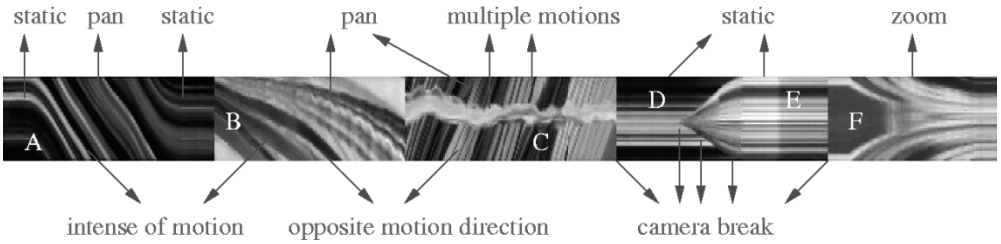


Fig. 2. Patterns in a spatio-temporal slice.

infers the occurrence of a new event; on the other hand, the orientation of texture depicts camera and object motions. While traditional approaches to motion sequence analysis tend to formulate computational methodologies on two or three adjacent frames, spatio-temporal slices provide rich visual cues along a larger temporal scale, which are vividly to be exploited for video processing and representation. The former gives a snapshot of motion field; the later, in contrast, offers a glance of motion events.

A. Spatio-Temporal Patterns

Fig. 2 shows a spatio-temporal slice extracted from a video composed of six shots. By careful observation of the patterns inherent in this slice, we envisage the possibilities of accomplishing various vision tasks by analyzing these patterns, possibly in a more effective and efficient way. In brief, the cues that are directly perceived from these patterns are as follows.

- *Shot boundaries* are located at places where the color and texture in a slice show dramatic changes. These changes may involve more than one frame as indicated by the boundary of shots D and E.
- *Camera motion* is inferred directly from the texture pattern. For instance, horizontal lines depict stationary camera and object motion; slanted lines depict camera panning. Fig. 3 further shows various patterns due to camera and object motions. A static sequence exhibits horizontal lines across $\mathbf{H}$ and $\mathbf{V}$; while camera panning and tilting results in one slice indicating the speed and direction of the motion, and the other slice explores the panoramic information [23]. For zooming, the lines in slices are either expanded in or out in a V-shape pattern.
- *Multiple motions* are perceived when two dissimilar texture patterns appeared in a shot, as shown in shot C. In this shot, the middle region describes a nonrigid object motion, while the background region indicates camera panning.
- *Motion direction* is indicated by the orientation of slanted lines. In shot B, the camera moves to the left; in shot C, the camera moves to the right.
- *Motion intensity* is proportional to the gradient of slanted lines. For instance, the speed of panning in shot A is faster than shot B.

It is not surprising to find that some conventional computer vision and image processing algorithms can be applied to analyze these patterns. Shot boundaries can be detected by color and texture segmentation (video partitioning) [16]–[18]; the type of camera motion, and the direction, velocity, and acceleration of motion can be estimated through the orientation and gradient of slanted lines (video characterization) [21]; motion layers can be obtained by decomposing dissimilar color and texture regions in the spatio-temporal slices of a shot (video segmentation).

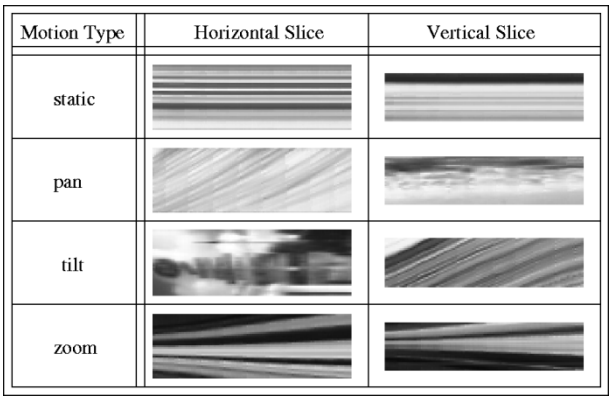


Fig. 3. Patterns in both horizontal and vertical slices.

In this paper, our algorithms are formulated directly in 2-D space, i.e., (x,t) and (y,t) space. These algorithms can be extended to three-dimensional (3-D) space, i.e., 3-D space. Better results are generally expected in the 3-D formulation, however, with heavy computational load. Our algorithms in the 2-D formulation are both efficient and effective particularly for analyzing sport videos where motion is mainly restricted to camera pan, tilt, zoom, and the combination of pan and object movement.

B. Computational Domain: Compressed vs Uncompressed

A digital video, in general, is composed of 25 to 30 frames per second. Processing a one hour video is, hence, burdened with heavy computation due to huge set of data (90 000 to 108 000 frames). However, as digital video becomes more pervasive, an efficient way of analyzing, annotating, and browsing video will be in high demand. As a result, processing videos by using the information extracted directly from MPEG data has become popular in literature [22], [28], [33], [34]. Working directly with the compressed data, on one hand, greatly reduces the processing time, while on the other hand, enhances storage efficiency. This is because the amount of data is significantly reduced, and only the partial decoding of compressed data[1] is needed.

For computational and storage efficiency, we propose to process and analyze spatio-temporal slices directly in the compressed video domain (MPEG domain). Slices can be obtained

[1]For instance, inverse quantization, decoding Huffman code, reconstruction of DC sequence from P-frames and B frames. Notice that Inverse Discrete Cosine Transform (IDCT) can be omitted in partial decompression.

from the DC image[2] volume which is easily constructed by extracting the DC components[3] of MPEG video. The resulting data is smoothed while the amount is reduced by 64 times in the MPEG domain. For an image volume of size $M \times N \times T$, the DC image volume has size $(M/8) \times (N/8) \times T$. As a result, spatio-temporal slices extracted horizontally and vertically from a DC image volume have size $(M/8) \times T$ and $(N/8) \times T$, respectively.

III. Motion Analysis

In our approach, the local orientations of temporal slices are estimated by the structure tensor introduced in [7], [11]. The global orientations are further described by tensor histograms. By modeling the trajectories in tensor histograms, our approach is capable of classifying motion types as well as separating different motion layers.

For our application, we process all slices, both horizontal and vertical, in a volume to analyze the spatio-temporal patterns due to various motions. For the ease of understanding, a horizontal slice with dimension (x, t) is denoted as $\mathbf{H}$, and a vertical slice with dimension (y, t) is denoted as $\mathbf{V}$. In addition, a horizontal slice $\mathbf{H}$ at the location $y = i$ is denoted as $\mathbf{H}_{|y=i}$, and a pixel located at $\mathbf{H}_{|y=i}$ is written as $\mathbf{H}(x, t)_{|y=i}$. Since both $\mathbf{H}$ and $\mathbf{V}$ will be processed in the same way, we only describe the analysis of $\mathbf{H}$ slices in the remaining paper.

A. Structure Tensor

Structure tensor computation has been extensively studied and applied to computer vision [7], [11], [15]. Local structure of an n-dimensional space can be represented by a symmetric tensor of the form [7]

$$\Gamma = Avv^T \tag{1}$$

where v is a vector in n-dimensional space and A is a constant greater than zero. In 2-D space, tensor represents local structure as an ellipse, as shown in Fig. 4. The major axis (e_1) of the ellipse estimates the orientation of local structure while the shape describes the variation of orientation.

One way to analyze the variation of orientation is to decompose the matrix in (1) into eigenvectors $(e_1, e_2, \ldots, e_n)$ and eigenvalues $(\lambda_1, \lambda_2, \ldots, \lambda_n)$. The eigenvectors correspond to the principle directions while the eigenvalues encode the size and shape of an ellipsoid. Compared with vector or scalar representation, structure tensors offer the following advantages.

- Since the variation of orientation can be computed, the confidence in the estimation is inherently encoded. For instance, in a 2-D local structure, a larger λ_1 when compared to λ_2 results in a higher confidence.
- As shown in (1), orientation can be defined only in modulo 180°, i.e., changing the sign of x yields the same representation.

In temporal slices, local structure is observed due to the change of pixel intensity. Hence, the vector v in (1) can be replaced by the gradient vector. Consequently, the local structure of a slice $\mathbf{H}$ is represented by tensor Γ as

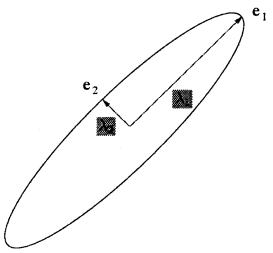


Fig. 4. Tensor representation.

$$\Gamma = \begin{bmatrix} \mathbf{J}_{xx} & \mathbf{J}_{xt} \\ \mathbf{J}_{xt} & \mathbf{J}_{tt} \end{bmatrix} \tag{2}$$

where

$$\mathbf{J}_{xx} = \sum_{x',t' \in w} \hat{\mathbf{H}}_x^2 (x - x', t - t')$$
$$\mathbf{J}_{xt} = \sum_{x',t' \in w} \hat{\mathbf{H}}_x (x - x', t - t') \hat{\mathbf{H}}_t (x - x', t - t')$$
$$\mathbf{J}_{tt} = \sum_{x',t' \in w} \hat{\mathbf{H}}_t^2 (x - x', t - t')$$

where $\hat{\mathbf{H}}_x = \partial(\mathbf{G} * \mathbf{H})/\partial x$ and $\hat{\mathbf{H}}_t = \partial(\mathbf{G} * \mathbf{H})/\partial t$ are partial derivatives along the spatial and temporal dimensions respectively. Notice that each slice is smoothed by a Gaussian kernel $\mathbf{G}$ prior to tensor computation to suppress noise. The window of support w is set to 3×3 and centered at each pixel in slices. The purpose of summation in w is to obtain new local information which can be assigned a higher degree of confidence in estimation [7].

The rotation angle θ of Γ indicates the direction of gray level change in w. Rotating the principle axes of Γ by θ, we have

$$\mathbf{R} \begin{bmatrix} \mathbf{J}_{xx} & \mathbf{J}_{xt} \\ \mathbf{J}_{xt} & \mathbf{J}_{tt} \end{bmatrix} \mathbf{R}^T = \begin{bmatrix} \lambda_x & 0 \\ 0 & \lambda_t \end{bmatrix} \tag{3}$$

where

$$\mathbf{R} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

and λ_x, λ_t are eigenvalues. In (3), since we have three equations with three unknowns[4], θ can be solved and expressed as

$$\theta = \frac{1}{2} \tan^{-1} \frac{2\mathbf{J}_{xt}}{\mathbf{J}_{tt} - \mathbf{J}_{xx}}. \tag{4}$$

The local orientation ϕ of a w in slices is computed as

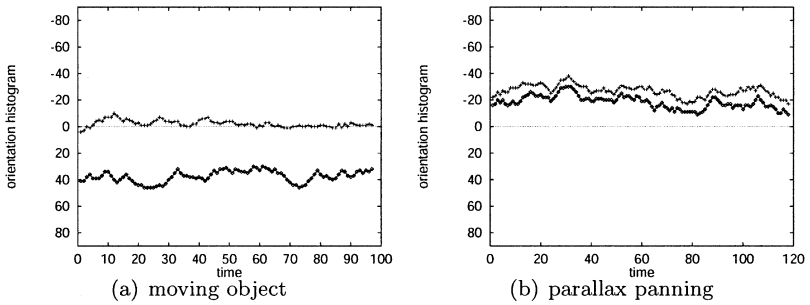
$$\phi = \begin{cases} \theta - \frac{\pi}{2} & \theta > 0 \\ \theta + \frac{\pi}{2} & \text{otherwise} \end{cases} \quad \phi = \left[-\frac{\pi}{2}, \frac{\pi}{2} \right]. \tag{5}$$

[2]DC image is formed by using the first coefficient of each 8×8 Discrete Cosine Transform (DCT) block.

[3]The algorithm introduced by Yeo & Liu [32] is applied to estimate DC components from P-frames and B-frames.

[4]The three unknowns are θ, λ_x and λ_t. The three equations are

$$\mathbf{J}_{xx} \cos^2\theta + \mathbf{J}_{tt} \sin^2\theta - \mathbf{J}_{xt} sin2\theta = \lambda_x$$
$$\frac{1}{2}(\mathbf{J}_{xx} - \mathbf{J}_{tt}) \sin 2\theta + \mathbf{J}_{xt} \cos 2\theta = 0$$
$$\mathbf{J}_{xx} \sin^2\theta + \mathbf{J}_{tt} \cos^2\theta + \mathbf{J}_{xt} sin2\theta = \lambda_t$$



(a) moving object (b) parallax panning

Fig. 5. Motion trajectories in the tensor histograms.

It is useful to add in a certainty measure to describe how well ϕ approximates the local orientation of w. The certainty c is estimated as

$$c = \frac{(\mathbf{J}_{xx} - \mathbf{J}_{tt})^2 + 4\mathbf{J}_{xt}^2}{(\mathbf{J}_{xx} + \mathbf{J}_{tt})^2} = \left(\frac{\lambda_x - \lambda_t}{\lambda_x + \lambda_t}\right)^2 \tag{6}$$

and $c = [0, 1]$. For an ideal local orientation, $c = 1$ when either $\lambda_x = 0$ or $\lambda_t = 0$. For an isotropic structure i.e., $\lambda_x = \lambda_t$, $c = 0$.

B. Tensor Histogram

The distribution of local orientations across time inherently reflects the motion trajectories in an image volume. We can construct two tensor histograms, one for all horizontal slices and the other for all vertical slices, to model the motion distribution. Denote $\phi(x,t)_{|y=i}$ and $c(x,t)_{|y=i}$, respectively, as the local orientation and the associated certainty value of a pixel at location $\mathbf{H}(x,t)_{|y=i}$, a 2-D tensor histogram $\mathbf{M}(\phi,t)$ is expressed as

$$\mathbf{M}(\hat{\phi},t) = \begin{cases} \sum_i \sum_x \sum_t c(x,t)_{|y=i}, & \text{if } \phi(x,t)_{|y=i} = \hat{\phi} \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

which means that each pixel in slices votes for the bin (ϕ, t) with its certainty value c. The resulting histogram is associated with a confidence measure of

$$\mathbf{C} = \frac{1}{T \times M \times N} \sum_{\phi} \sum_{t} \mathbf{M}(\phi,t) \tag{8}$$

where T is the temporal duration and $M \times N$ is the image size. In principle, a histogram with low $\mathbf{C}$ should be rejected for further analysis.

Motion trajectories can be traced by tracking the histogram peaks over time. These trajectories can correspond to i) object and/or camera motions and ii) motion parallax with respect to different depths. Fig. 5 shows two examples, in (a) one trajectory indicates the nonstationary background, and one indicates the moving objects; in (b) the trajectories correspond to parallax motion.

IV. Temporal Motion Segmentation

Tensor histograms offer useful information for temporally segmenting and characterizing motions. The algorithm starts by tracking a dominant trajectory along the temporal dimension. A dominant trajectory $p(t) = \max_{-\pi/2<\phi<\pi/2}\{\mathbf{M}(\phi,t)\}$ is defined to have

$$\frac{\sum_{t=k}^{k+15} p(t)}{\sum_{t=k}^{k+15} \sum_{\phi} \mathbf{M}(\phi,t)} > \tau. \tag{9}$$

In (9), the dominant motion is expected to stay steady approximately for 15 frames (0.5 s). The threshold value $\tau = 0.6$ is empirically set to tolerate camera jitter. After a dominant trajectory is detected, the algorithm simultaneously segments and classifies the dominant motion trajectory. A sequence with static or slight motion has a trajectory of $\phi = [-\phi_a, \phi_a]$. Ideally, ϕ_a should be equal to 0. The horizontal slices of a panning sequence form a trajectory at $\phi > \phi_a$ or $\phi < -\phi_a$. If $\phi < -\phi_a$, the camera pans to the right; if $\phi > \phi_a$, the camera pans to the left. A tilting sequence is similar to a panning sequence, except that the trajectory is traced in the tensor histogram generated by vertical slices. The parameter ϕ_a is empirically set to $\pi/36$ (or 5° degree) throughout the experiments. For zoom, the tensor votes are approximately symmetric at $\phi = 0$. Hence, instead of being modeled as a single trajectory, the zoom is detected by

$$\frac{\sum_{\phi} \sum_{t>0} \mathbf{M}(\phi,t)}{\sum_{\phi} \sum_{t<0} \mathbf{M}(\phi,t)} \approx 1. \tag{10}$$

Figs. 6(a) and 7(c) shows the temporal slices of two shots which consist of different motions over time, while Figs. 6(b) and 7(d) shows the corresponding tensor histograms. In Fig. 6, the motion is segmented into two subunits, while in Fig. 7, the motion is segmented into three subunits.

A. Experiments

To verify the effectiveness of the proposed algorithm, we conduct an experiment on an MPEG-7 standard video, *Nhkvideo.mpg*. The video consists of 15 000 frames. The video partitioning approach introduced in the previous chapter is employed to partition the video into 45 shots. Table II summarizes the performance of the proposed approach. Throughout the experiment, camera rotation in shot 0 and shot 44 of *Nhkvideo.mpg* are falsely detected as zoom sequences. Similarly, in shot 11 the combination of object rotation and camera tilting has falsely been detected as zoom. In shots 7, 20, 31, and

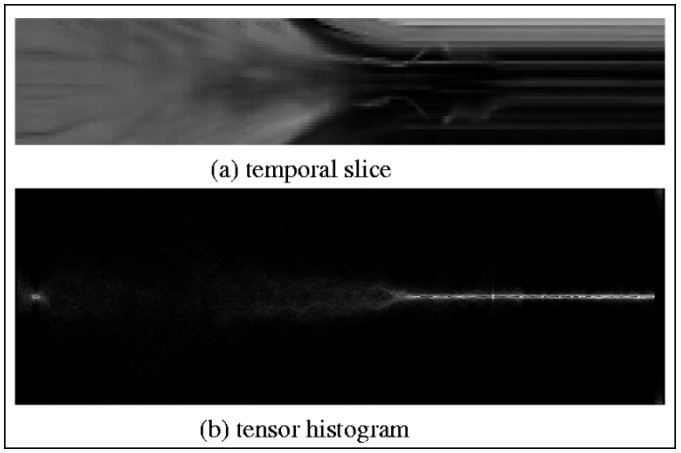


Fig. 6. Zoom followed by static motion.

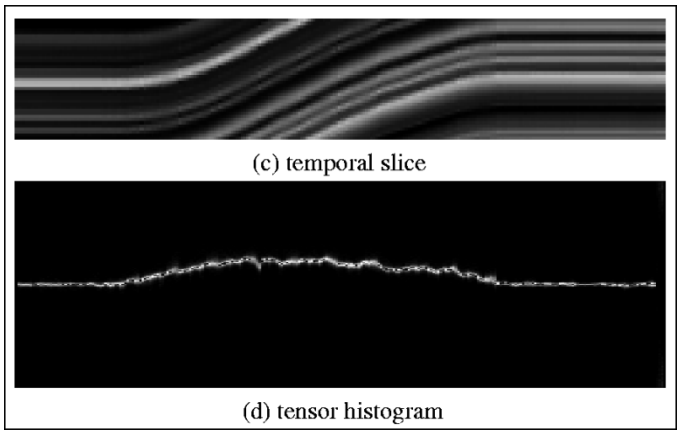


Fig. 7. Static, pan, and static motions.

40, the combination of camera pan and zoom has been falsely detected as pan motion. Some examples are shown in Fig. 8. In shot 43, the pan subunit is undetected since the corresponding slices are mostly occupied by homogeneous regions.

On a Pentium III platform with one processor and 128M main memory, the algorithm takes about 12 min to compute the tensor histograms of all shots, and takes less than one second to analyze and classify the camera motion of a tensor histogram. On average, the algorithm processes 20 frames per second. The speed can be further improved by selecting a subset of slices for the tensor histogram computation.

V. Spatial Segmentation

After motion characterization, subshots with multiple motion[5] are further processed by spatial segmentation. To filter out those subshots that are not suitable[6] for analysis (e.g., subshots where their motion spreads over the bins of tensor histograms), a simple algorithm is carried out to track the motion trajectories in a tensor histogram prior to segmentation. The algorithm first looks for $\Phi(t) = \arg\max_{\phi}\{\mathbf{M}(\phi, t)\}$ which is the histogram peak at time t, and then trace the trajectory by searching for next $\Phi(t+1) = \arg\max_{\Phi(t)-3\le\phi\le\Phi(t)+3}\{\mathbf{M}(\phi, t+1)\}$ at time $t+1$. If one of the resulting trajectory satisfies (9) with $\tau = 0.1$, spatial segmentation will be carried out. Fig. 5 shows the motion trajectories that are tracked by this simple algorithm.

In this section, we propose two different approaches for spatial segmentation. The first approach utilizes the color similarity among temporal slices while the second exploits the motion trajectories inherently exist in tensor histograms. The difference between these two approaches lies on the trade off between simplicity and effectiveness. The first approach is simple and efficient, however, may not function properly if the background is cluttered. The second approach, on the other hand, is comparatively robust with slightly more computational load.

A. Approach I: Exploiting Color Similarity

Fig. 9 shows an overview of this approach. The idea is to partition a 3-D image volume into several subvolumes by clustering

[5]Subshots that are not annotated as static, zoom, pan and tilt in Section IV are assumed to have multiple motion. Under this assumption, background components at various depths may be segmented into multiple layers when the camera pans.

[6]Typical examples are subshots with large area of homogeneous region or scenes with many moving objects.

TABLE II
Motion Annotation for the Video *NHKVIDEO.MPG*. C Denotes Correct Detection; F Denotes False Detection; M Denotes Missed Detection

shot	static	pan	tilt	zoom	shot	static	pan	tilt	zoom
0				F	23	C			
1		C			24	C			M
2			C		25	C			
3	C			C	26	C		C	
4	C			C	27	C			
5	C	C			28	C			C
6	C	C			29	C	C		C
7		F		M	30	C	C		
8	C				31	C	F		M
9	C				32	C			
10	C				33	C			
11				F	34	C			
12	C				35	C			
13	C				36	C			
14	C				37	C			
15	C				38	C			
16		C		C	39				C
17	C	C			40				M
18	C			C	41				C
19	C	C			42				C
20	C	F		M	43	C	M		C
21	C				44				F
22	C	C			45	C			

TABLE III
Summary of Table II

Motion	C	M	F	Recall	Precision
Static	35	0	0	1.00	1.00
Pan	9	1	3	0.90	0.75
Tilt	2	0	0	1.00	1.00
Zoom	10	5	3	0.67	0.77

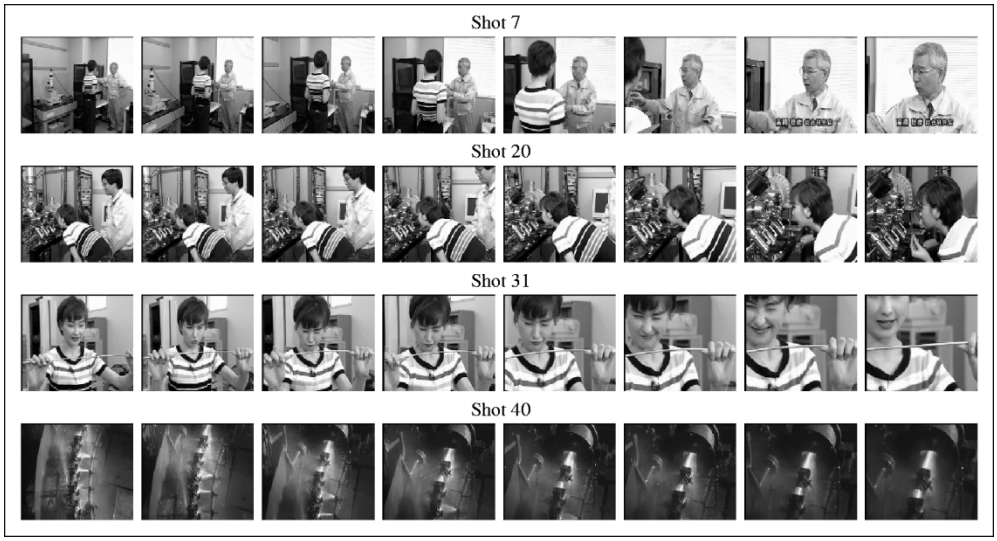


Fig. 8. Examples of false and missed detections.

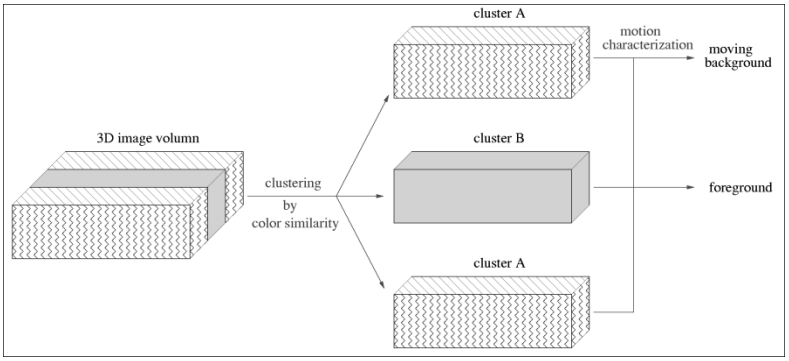


Fig. 9. Segmentation of an 3-D image volume into subvolumes through color similarity.

the temporal slices with similar color content. Ideally, each subvolume corresponds to the evolution of one moving object over time. For indexing and retrieval, these subvolumes can further be characterized and annotated by the method described in Section IV. In Fig. 9, the subvolumes with dominant motion are assumed reflecting the background motion.

We illustrate the idea of this approach by first showing an image sequence which involves object tracking. The sample images of this sequence are given in Fig. 10 while the corresponding horizontal and vertical slices are shown in Fig. 11. The horizontal slices model the camera and object motions, while the vertical slices explore the background panoramic information as well as follow the target object over time. We employ k-mean clustering, as given in Fig. 12, to group similar slices. As the number of clusters k is unknown, the similarities among adjacent slices are first exploited (steps 1 to 3 in Fig. 12) to estimate k. We adopt 3-D color histogram in HSV space for similarity measure. The hue h is quantized to 18 bins, while the saturation s and brightness v components are quantized to 3 bins respectively. The quantization provides 162 ($18 \times 3 \times 3$) distinct color sets. The similarity between two temporal slices $\mathbf{H}_{|y=i}$ and $\mathbf{H}_{|y=j}$ is

$$\sum_{h}\sum_{s}\sum_{v} \min\left(D(h,s,v)_{|y=i}, D(h,s,v)_{|y=j}\right) \qquad (11)$$

based on the color histogram intersection. $D(h,s,v)_{|y=i}$ and $D(h,s,v)_{|y=j}$ are the histograms of $\mathbf{H}_{|y=i}$ and $\mathbf{H}_{|y=j}$, respectively. Experimental results show that the horizontal slices are clustered as one group, while the vertical slices are clustered into two groups.[7] By projecting the clustering results into the original image volume, we obtained two subvolumes. After computing the tensor histograms, one of the subvolume correctly reflects the camera panning information.

We further employ a mosaicking algorithm to illustrate the correctness of the experimental result. The mosaic is constructed by pasting together the DC images based on the displacement computed from the correlation of a few scans in the image subvolume. Fig. 13 shows the mosaicked images; one corresponds with the tracked object, and the other one corresponds to the panning background. The tracked player in Fig. 13(a) is blurred due to 3-D head and body movements.

We carry out another experiment on a moving objects sequence, as shown in Fig. 14. The original image volume is divided into two subvolumes. The tensor histogram of the moving objects subvolume resembles a camera panning sequence, as indicated by the temporal slices in Fig. 14(f) and (g). The mosaicked image of the moving objects are shown in Fig. 15. With the current implementation the total time involved in clustering,

[7]In this experiment, the slices in Fig. 11(j), (k), (l), (p), (q) and (r) are into one group, while the slices in Fig. 11(m), (n) and (o) are into another group.

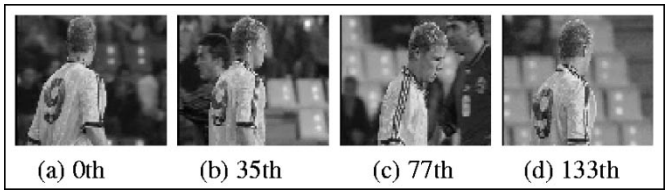


Fig. 10. Object tracking image sequence.

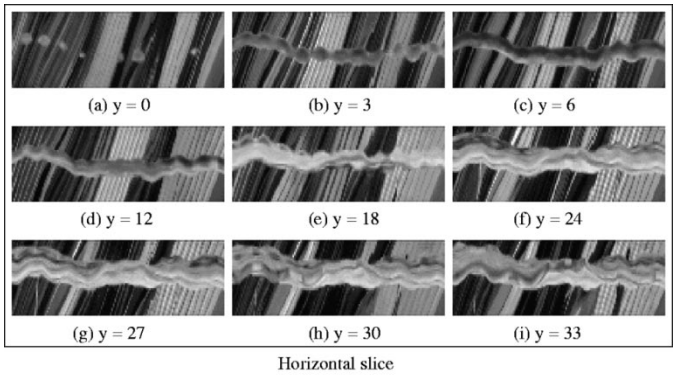


Fig. 11. Spatio-temporal slices of the image sequence in Fig. 10.

1. Compare the color similarity among the adjacent slices i.e., $\mathbf{H}_{|y=i}$ and $\mathbf{H}_{|y=i+1}$.
2. Create a new cluster if two adjacent slices are different in term of color content.
3. Group similar clusters by comparing the cluster centroids. The number of clusters k is determined in this step.
4. Perform k-mean clustering to fine tune the final result if $k > 1$.

Fig. 12. Clustering algorithm.

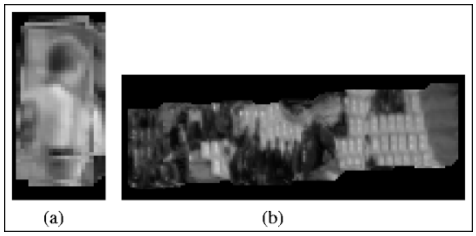


Fig. 13. Segmented motion layers of the image sequence in Fig. 10. (a) Target object and (b) mosaicked background image.

tensor histogram computation and mosaicking is approximately 12 frames per second on a Pentium III platform with one processor and 128M main memory.

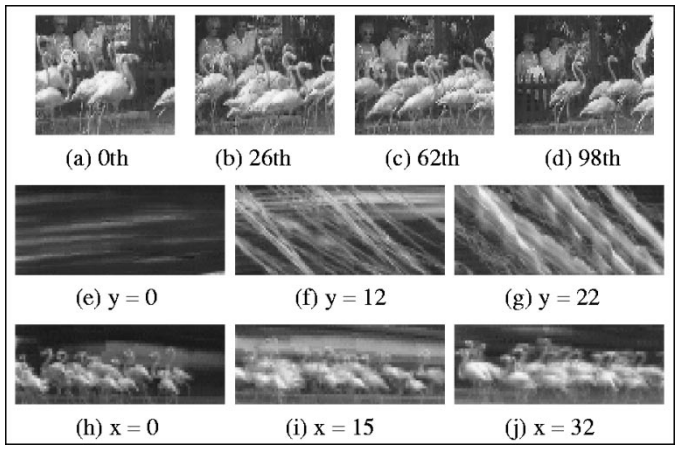


Fig. 14. (a)–(d) Moving objects sequence; (e)–(g) the horizontal temporal slices; and (h)–(j) the vertical temporal slices.

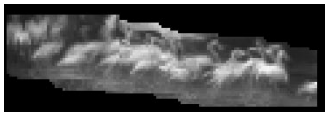


Fig. 15. Mosaicked image of the moving objects in Fig. 14.

B. Approach II: Exploiting Motion Similarity

The approach presented in Section V-A, on one hand, is simple and efficient; on the other hand, it cannot handle cases where the background is composed of various color elements. Fig. 17 shows an example. The color content among the horizontal slices are perceptually different, in addition, the foreground and background objects show some degree of similarity in color. While the correct segmentation is to partition the vertical slices as two groups and the horizontal slices as one group, the previous approach groups the vertical slices into one clusters and the horizontal slices into three clusters. As a result, the foreground and background layers cannot be successfully decomposed.

As a complement, we describe a more general approach to solve this problem in this section. Imagine that there are multiple motion trajectories in a tensor histogram, intuitively these trajectories can be simply back-projected to the spatio-temporal slices to form spatially separated motion layers. Fig. 16 illustrates the major flow of this idea. Given a set of spatio-temporal slices, a 2-D tensor histogram (introduced in Section III-B) is computed. The 2-D histogram is further nonuniformly quantized into a 1-D normalized motion histogram. The histogram consists of seven bins to qualitatively represent the rigid camera and object motions. The peak of the histogram is back-projected onto the original image sequence. The projected pixels are aligned to generate a complete background which may have holes. Foreground objects are then effectively segmented by background subtraction and color back-projection techniques. In contrast to the previous method, this proposed approach successfully decomposes the image sequence shown in Fig. 17 into foreground and background layers.

1) Background Reconstruction: We begin by introducing a technique for locating the regions in frames that correspond to

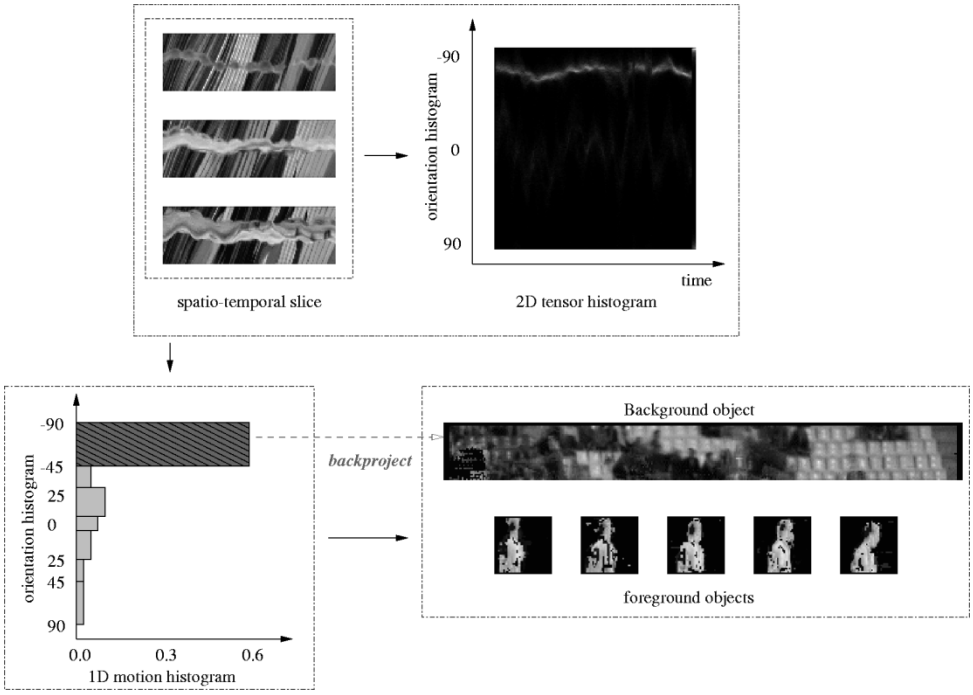


Fig. 16. Scheme for background and foreground segmentation.

a dominant motion. Then we propose a novel approach to compose these regions into a panoramic image, by matching corresponding points across frames directly in spatio-temporal slices.

Quantization of motion histogram: Given a 2-D tensor histogram $\mathbf{M}(\phi, t)$ with temporally coherent motion unit, the tensor orientation ϕ is nonuniformly quantized into seven bins, where

$$\begin{aligned}\Phi_1 &= [-90^\circ, -45^\circ)\\ \Phi_2 &= [-45^\circ, -25^\circ)\\ \Phi_3 &= [-25^\circ, -5^\circ)\\ \Phi_4 &= (-5^\circ, 5^\circ]\\ \Phi_5 &= (5^\circ, 25^\circ]\\ \Phi_6 &= (25^\circ, 45^\circ]\\ \Phi_7 &= (45^\circ, 90^\circ].\end{aligned}$$

The scheme quantifies motion based on its intensity and direction. Φ_1 and Φ_7 represent the most intense motion, while Φ_4 represents no or slight motion. The normalized 1-D motion histogram $\mathbf{N}$ is computed by

$$\mathbf{N}(\Phi_k) = \frac{\sum_{\phi_i \in \Phi_k} \sum_{t} \mathbf{M}(\phi_i, t)}{\sum_{j=1}^{7} \mathbf{N}(\Phi_j)}. \tag{12}$$

Adaptive setting of quantization scale is a difficult problem. Since we assume motion characterization is performed prior to motion segmentation, camera motion is supposed to be coherent and smooth. Thus, the setting should not be too sensitive to the final results. Empirical results indicate that our proposed setting is appropriate for most cases. For the case when a motion trajectory crosses the boundary of two bins, the result of foreground detection will be effected. Nevertheless, this undesired effect can be remedied by the color back-projection technique which will be discussed later.

Tensor back-projection: The prominent peak in a 1-D motion histogram reflects the dominant motion of a sequence, as shown in Fig. 16. By projecting the peak back to the temporal slices $\mathbf{H}_{|y=i}$, we can locate the region (referred to as the layer of support) that induces the dominant motion[8]. The support layer is computed as

$$\mathrm{Mask}(x,t)_{|y=i} = \begin{cases} 1, & \phi \in \hat{\Phi} \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

where

$$\hat{\Phi} = \arg\left\{ \max_{\Phi_k} \mathbf{N}(\Phi_k) \right\}. \tag{14}$$

(x, t) is the location of a pixel in $\mathbf{H}_{|y=i}$. Fig. 18 illustrates an example. The temporal slice in Fig. 18(a) consists of two motions, while the layer of support in Fig. 18(a) locates the region corresponding to the dominant motion (white color).

Tensor back-projection will generally leave holes in the layer of support. The number of holes is mainly dependent on the degree of occlusion due to objects in other layers, and the size and number of regions with no textural information. These holes, nevertheless, can be filled by techniques like morphological filtering, smoothing and interpolation. Besides holes, a support layer at the border of two motion layers cannot be precisely located. This is due to the effect of Gaussian smoothing when computing structure tensors. In addition, the estimation fails when representing a local structure that occupies more than one motion layer since the orientation variation of tensor is large.

Point correspondence and background mosaicking: Once the support layer of a dominant motion is computed, intuitively

[8]This process is equivalent to the quantization of tensor orientation computed for spatio-temporal slices.

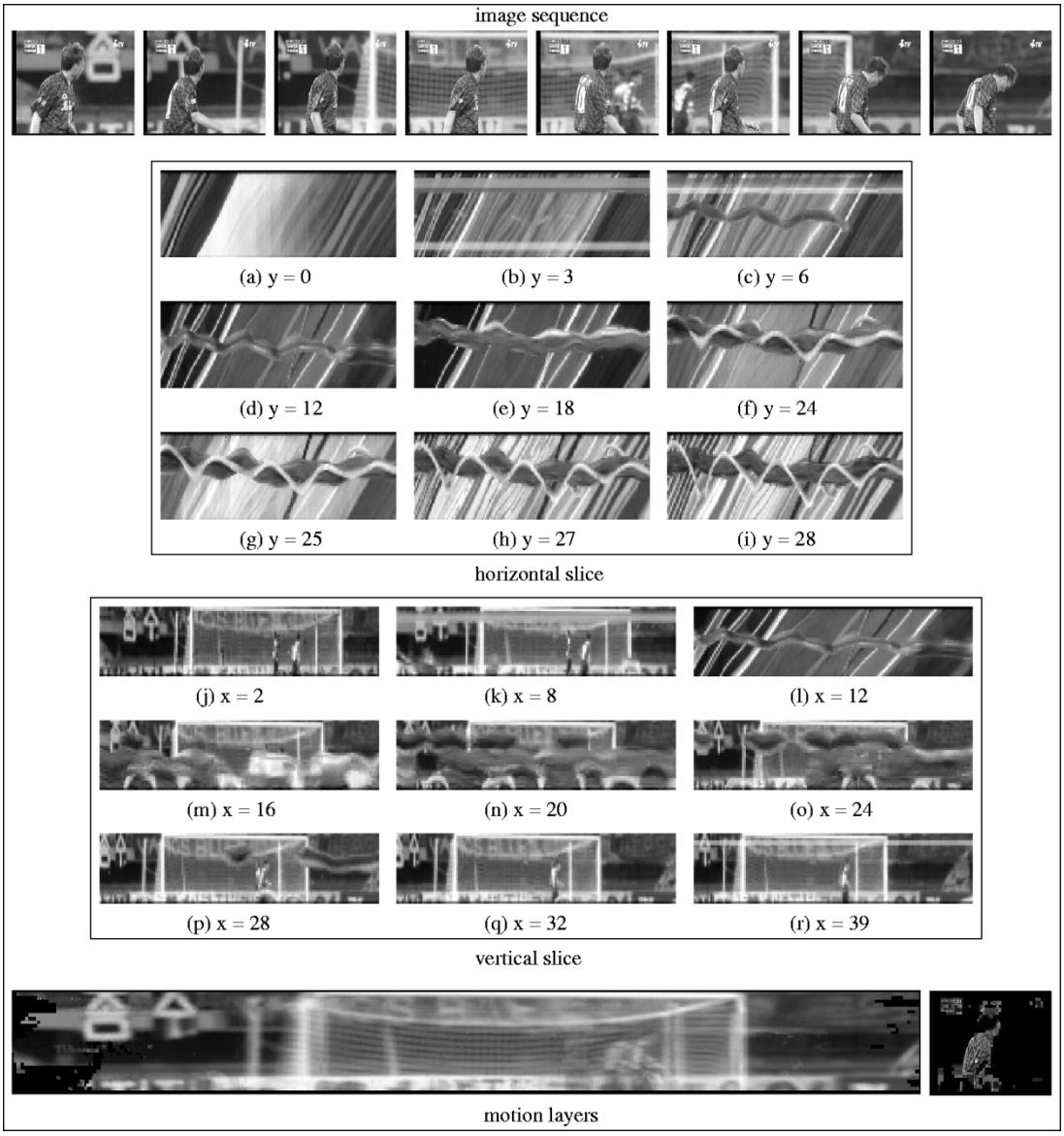


Fig. 17. Results of spatial motion segmentation for the background scene which consists of various color elements.

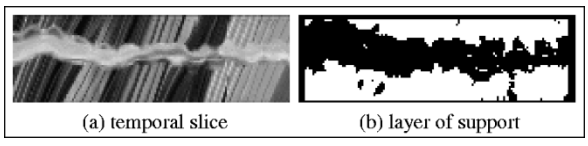


Fig. 18. Support layer of dominant motion.

we can align and paste the corresponding regions to reconstruct the background image. Nevertheless, this is not a trivial issue since theoretically the correspondence feature points need to be matched across frames. This is an ill-posed problem specifically at the regions of no texture information. The problem is further complicated by occluded and uncovered feature points at a particular time instance.

To solve this problem, we propose a method that selects temporal slice $\mathbf{H}_i$ which contains two adjacent scans $\mathbf{H}(x,t)_{|y=i}$ and $\mathbf{H}(x,t+1)_{|y=i}$ with the most textural information at time t, and then perform feature points matching across the two scans. For each time instance t, the criterion for selecting a slice is

$$\hat{\mathbf{H}} = \arg\max_{\mathbf{H}_{|y=i}} \left\{ \frac{\mathbf{C}(t)_{|y=i} + \mathbf{C}(t+1)_{|y=i}}{|n(t)_{|y=i} - n(t+1)_{|y=i}| + 1} \right\} \tag{15}$$

and

$$\mathbf{C}(t)_{|y=i} = \sum_x c(x,t)_{|y=i} \mathrm{Mask}(x,t)_{|y=i}$$
$$n(t)_{|y=i} = \sum_x \mathrm{Mask}(x,t)_{|y=i}$$

where $c(x,t)_{|y=i}$ is the certainty measure of a tensor at $\mathbf{H}(x,t)_{|y=i}$. The value $c_{|y=i}$ indicates the richness of texture of the surrounding of a pixel located at (x,t). In practice, $\mathbf{C}(t)_{|y=i} > 0$ and $n(t)_{|y=i} \geq 2$.

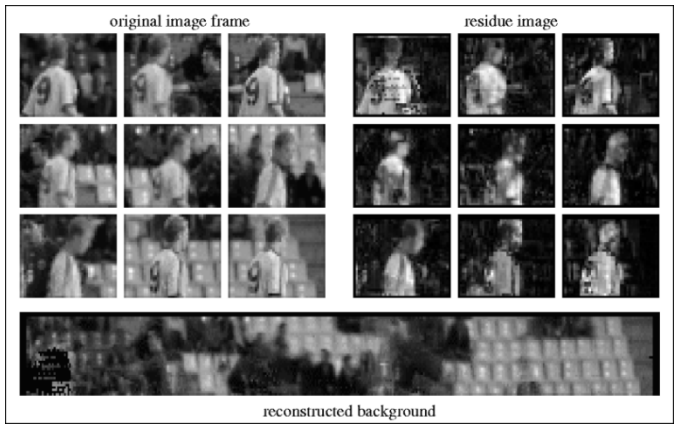


Fig. 19. Results of foreground detection by background subtraction.

We adopt the motion model involves only translation[9] when aligning and pasting two image frames. Denote $\hat{d}(t)$ as the translation vector at time t, $\hat{d}(t)$ is directly computed from two scans by

$$\hat{d}(t_1) = \arg\min_d \left\{ \text{med} |\mathbf{H}(x,t)_{|y=i} - \mathbf{H}(x+d, t+1)_{|y=i}| \right\} \tag{16}$$

where med is a robust median estimator employed to reject outliers. The sign of d is dependent on the $\hat{\Phi}_k$ in (14) which indicates the motion direction. From (16), it is interesting to note that the problem of occlusion and uncovered regions is implicitly solved due to the use of support layer and robust estimator. Naturally the occluded region at frame i can be filled by the uncovered region at frame $j \neq i$. An example of a mosaicked background reconstructed from 140 frames is shown in Fig. 19.

In practice, however, the visual effect of a mosaicked image may suffer from the ghosting effect [26] (or double images) and the holes. The former problem is due to the variation of depth in the background scene, and the error during feature points registration. The latter problem is because of the occluded or unexplored background regions. These problems can be partially solved by the deghosting algorithm [26] and interpolation techniques.

2) Foreground Detection: The reconstruction of a background object, in principle, has substantially facilitated the task of foreground detection. In this section, we introduce two different methods, namely background subtraction and color back-projection, to approximately segment the foreground objects. These two methods are finally combined to arrive at a better solution in locating the objects.

Background subtraction: The simplest approach to detect foreground objects is by subtracting image frames from a reconstructed background. Denote $\mathbf{Bg}$ as a reconstructed background and $\mathbf{I}$ as an image frame[10] indexed by $X = (x, y)$ and time t, we write

$$\mathbf{R}(X,t) = |\mathbf{Bg}(X + \mathbf{d}(t)) - \mathbf{I}(X,t)| \tag{17}$$

[9]This model is suitable for our applications (as indicated in Table I), particularly for analyzing sport videos and home videos. For multiple motion case, these videos typically involve camera pans to track a person or an object moving in front of a static background.

[10]$\mathbf{I}(x,i,t)$ is also the pixel location of the horizontal slice $\mathbf{H}(x,t)_{|y=i}$.

where $\mathbf{d}(t) = \sum_{i=0}^{k-1} \hat{d}(i)$ and $\mathbf{R}$ is a residue image. If $\mathbf{Bg}(X + \mathbf{d}(t))$ is a hole, $\mathbf{R}(X,t)$ will be filled by the value 255. Fig. 19 shows the results of background subtraction. The residue images contain some noise due to the ghosting effect. This noise, in practice, can be removed by either a threshold setting or morphological filtering techniques.

Color back-projection: Suppose the approximate region of a foreground object is known, we can actually replace the color values of that region by its color distribution probabilities. In this case, the dominant color of a foreground object will have a high probability, while the subregions not belonging to the foreground object should ideally have values close to zero. Thus, we can automatically prune the approximate region, whilst effectively locating the foreground object.

In our scheme, the support layer of a foreground object Mask_f can be simply obtained by inverting the support layer of a background object[11] Mask_b, i.e.,

$$\text{Mask}_f(X,t) = \begin{cases} 1, & \text{if } \text{Mask}_b(X,t) = 0 \\ 0, & \text{if } \text{Mask}_b(X,t) = 1. \end{cases} \tag{18}$$

Our approach computes a 3-D normalized color histogram for the region $\mathcal{R}$ supported by Mask_f throughout a sequence, and then projects the probability values $[0,1]$ in the histogram back to $\mathcal{R}$ to obtain $\hat{\mathcal{R}}$. In other words, each pixel is replaced by its color value weighted by a probability. The probability is computed from the color histogram of the detected foreground objects obtained through background subtraction. Let $\mathcal{H}$ be a normalized histogram, and N_k be the k quantized color value, mathematically we have

$$\begin{aligned} &\text{project} : \mathcal{H}(N_k) \\ &\quad = \sum_t \sum_X \frac{1}{\mathcal{A}} \text{ for } \forall_{X,t} \{ \mathcal{Q}(\mathcal{R}(X,t)) = N_k \} \end{aligned} \tag{19}$$

$$\text{back-project} : \hat{\mathcal{R}}(X,t) = \mathcal{H}(\mathcal{Q}(\mathcal{R}(X,t))) \tag{20}$$

where $\mathcal{A}$ is the area of $\mathcal{R}$, while function $\mathcal{Q}$ is the color quantization. In (19), a normalized color histogram $\mathcal{H}$ is computed for the region $\mathcal{R}$. In (20), the value of each pixel in $\mathcal{R}$ is replaced by its corresponding probability value in $\mathcal{H}$. Fig. 20 shows the examples on Mask_f, and the projected image as a result of color back-projection.

The color back-projection idea is similar to Swain's color indexing scheme [27] and Huang's spatial color index algorithm [8]. They assume that the template of an object is known beforehand; a similar object in a new image is first recognized by color histogram intersection, and then located by color back-projection and convolution. In our case, the histogram intersection and template convolution are not necessary since the initial object location is approximately known.

Foreground image computation: Background reconstruction is always imperfect due to the ghosting effect, as a result, noise removal after background subtraction can be a dirty task. Likewise, the drawback of color back-projection is amplified when the foreground and background are some how similar in color; the color histogram need to be finely quantized in order to distinguish the color of foreground and background objects.

[11]$\text{Mask}_b(X,t) = \text{Mask}_b(x,i,t) = \text{Mask}(x,t)_{|y=i}$ in (13).

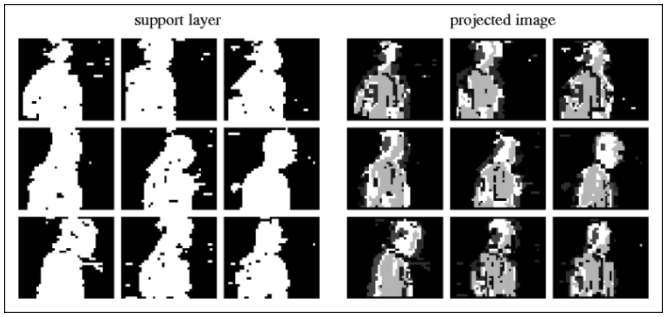


Fig. 20. Results of foreground detection by color back-projection. The corresponding original image frames are shown in Fig. 19.

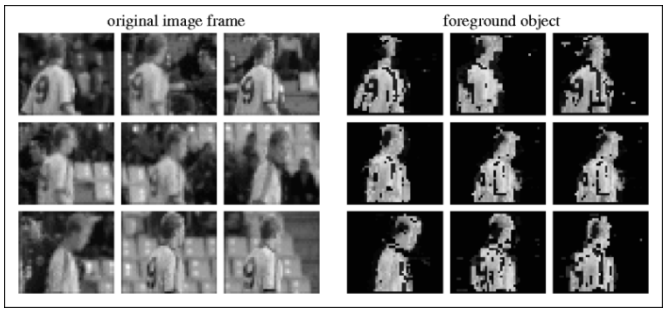


Fig. 21. Segmentation of foreground objects.

As a compromise, the two approaches can be linearly combined to trade-off their disadvantage. Denote $\hat{\mathbf{R}}$ as the normalized residue image of an image frame $\mathbf{I}$, a foreground image $\mathcal{F}$ is computed by

$$\mathcal{F}(X,t) = \mathbf{Pr}(X = \text{Foreground}, t) \times \mathbf{I}(X,t) \qquad (21)$$

where

$$\mathbf{Pr}(X = \text{Foreground}, t) = \frac{1}{2}\{\hat{\mathbf{R}}(X,t) + \hat{\mathcal{R}}(X,t)\} \qquad (22)$$

is the probability of a pixel $\mathbf{I}(X,t)$ belongs to a foreground object. In (21), ideally, the background pixels of $\mathbf{I}$ should be set to zero, while the foreground pixels should be set to a value closed to the color value of $\mathbf{I}$.

Fig. 21 shows the computed foreground objects, together with the original image frames. Compared with Figs. 19 and 20, the noise effect is minimized.

3) Experiment: We conduct experiments on six image sequences and each sequence has approximately 150–250 frames. In Fig. 22–27, the original image frames, the computed foreground images, and the mosaicked background of these image sequences are shown. In each tested sequence, there is a foreground object (player) running across the background, while the camera zooms and pans either to the left or the right to track the foreground object. The background is not totally rigid, it may introduce slight motion due to the movement of audiences (Fig. 22) or other players who stand in front of background (Figs. 24 and 26). Notice that although the computed foreground and background images are processed directly in a DC image sequence, these images can be easily scaled up by decompressing their MPEG sequences.

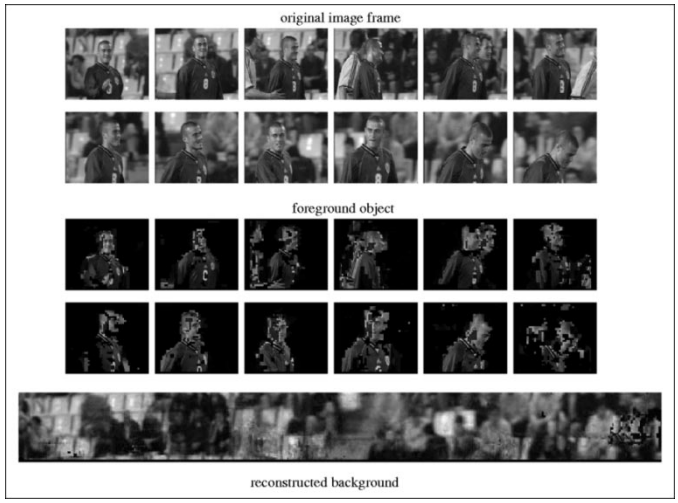


Fig. 22. Experiment 1.

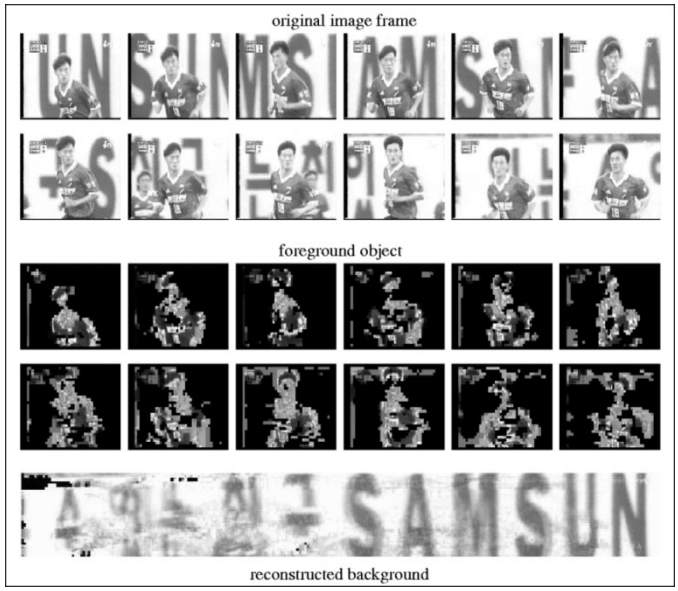


Fig. 23. Experiment 2.

As seen from these figures, the foreground and background objects are in general correctly separated. It is worthwhile mentioning the following three observations.

- Homogeneous regions are not correctly segmented, as shown in Figs. 24, 25 and 27. This is a general problem for most vision tasks. Nevertheless, this undesired effect has been minimized using the color back-projection technique, in the cases where the foreground and background do not share similar coloring.
- The noise introduced by a background object will create a ghosting effect in the reconstructed background image. This in turn affects the results of foreground detection. One obvious example can be found in Fig. 26. In the first few frames, the segmented results are not satisfactory. However, as the targeted foreground object moves away from the noisy background, the segmented results are better. This implies that certain degree of post-processing, such as object tracking, is necessary to improve the results.

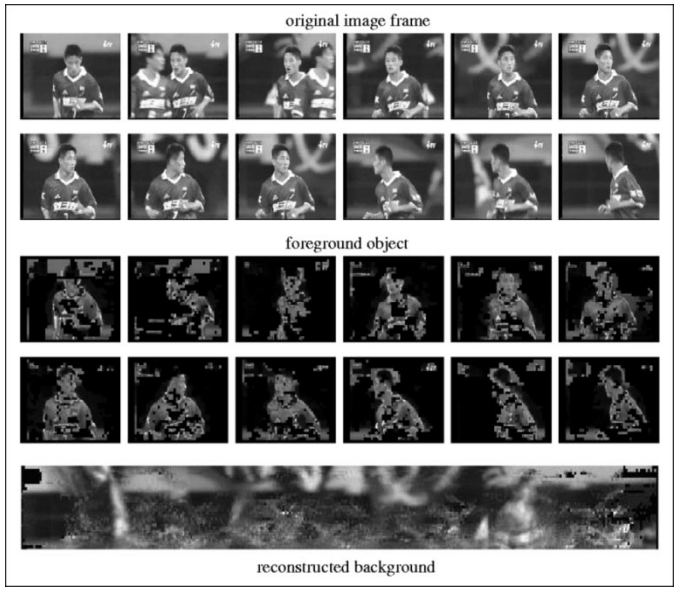


Fig. 24. Experiment 3.

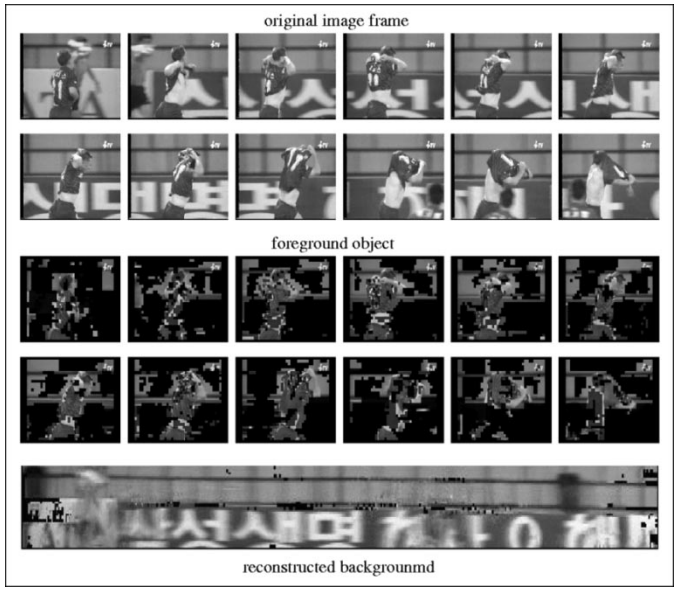


Fig. 25. Experiment 4.

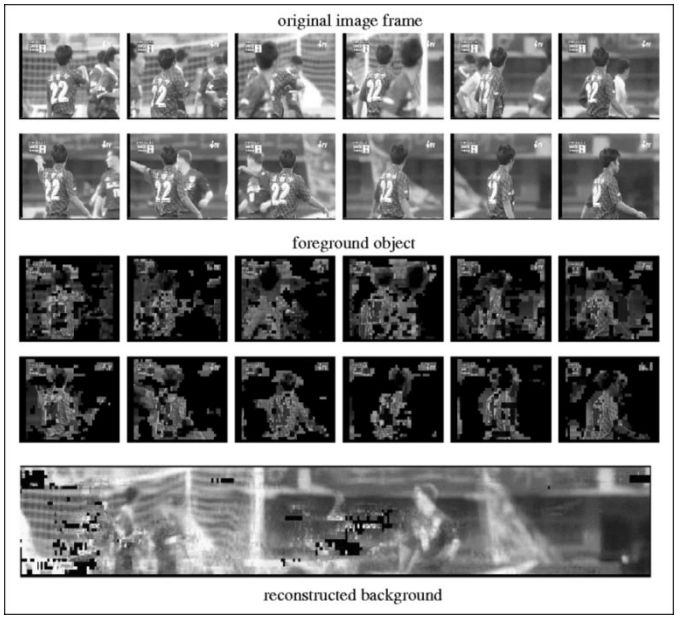


Fig. 26. Experiment 5.

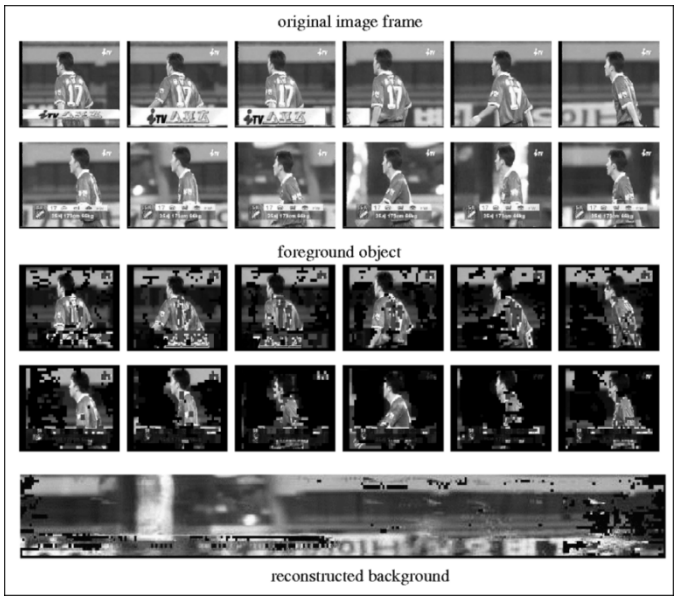


Fig. 27. Experiment 6.

- The color back projection technique can hinder the segmentation results, as shown in Fig. 25. After the player takes off his shirt, some portions of the body are not correctly segmented.

The experiments are conducted on a Pentium III platform with one processor and 128M main memory. The algorithm takes approximately 10 frames per second for background reconstruction and foreground detection. In brief, the proposed techniques are both effective and efficient mainly for sequences involving translational dominant motions. These sequences happen frequently in sport and home videos.

VI. Conclusion

Based on the pattern analysis of slices in spatio-temporal image volume, we have presented novel approaches in temporally and spatially segmenting the content of videos. In terms of effectiveness, the proposed works are suitable for content-based video representation, since qualitative information can be extracted directly from slices to describe the content of videos. In terms of efficiency, since slices are extracted from MPEG DC sequences for pattern analysis, the proposed methods can achieve reasonable speed due to the significant reduction of input size.

We have considered the problems of characterizing motion and separating motion layers by utilizing the patterns in slices. The proposed motion analysis method, on one hand, employs structure tensor to compute the local orientation of slices; while on the other hand, utilizes tensor histograms to capture the temporally and spatially separated motion patterns across time. A temporal motion segmentation method has hence been

proposed to track the motion trajectories in tensor histograms to describe camera motions over time. Throughout experiments, this method is found to be robust to static, pan and tilt types of camera motions. In addition, to spatially partition an image volume into different motion layers, the similarity of slices is exploited. We have demonstrated the approach with several image sequences. While the method is simple and efficient, it cannot handle cases where the background is cluttered. To solve this problem, we have further proposed a more general motion-based foreground and background layers decomposition method. The basic idea to obtain a background support layer is by back-projecting the dominant motion trajectory in a tensor histogram to spatio-temporal slices. Compared to most motion segmentation methods, the proposed approach is efficient since the problems of motion estimation and motion segmentation are decoupled and there is no iterative procedure involved.

Several issues have been left out in this paper and ultimately intended for future works. These issues include modeling the relationship among slices and robust tracking of foreground object. The structure tensor computation approach described in this paper can be more efficiently implemented if the redundancies among slices are carefully explored. For some cases, probably only a few selected slices instead of a whole image volume are sufficient for motion computation. In addition, besides horizontal and vertical slices, diagonal slices are also useful particularly for diagonal motion. To adaptively decide which slices should be selected, a pre-filtering of MPEG motion vectors to obtain initial cues may be useful, and this requires further investigation.

The foreground object computation algorithm can be effectively done by assuming the camera motion is smooth. Consider a sequence taken by a hand held camera, it may undergo camera jitter at certain duration. We can, however, derive a method to track objects beginning from the image frames that undergo smooth camera motion, and then propagate the results forward or backward to track objects during the period when the camera suffers from jitter. The selection of initial tracking period can be inferred from the motion trajectories of tensor histograms. The degree of smoothness and the certainty value of a partial segment of trajectory is a good indication of whether a camera has undergone jitter.

REFERENCES

[1] E. H. Adelson and J. Bergen, "Spatiotemporal energy models for the perception of motion," *J. Opt. Soc. Amer.*, vol. 2, no. 2, pp. 284–299, Feb. 1985.

[2] A. Akutsu, Y. Tonomura, and H. Hamada, "Videostyler: Multi-dimensional video computing for eloquent media interface," in *Proc. Int. Conf. on Image Processing*, vol. 1, 1995, pp. 330–3.

[3] S. Ayer, P. Schroeter, and J. Bigun, "Segmentation of moving objects by robust motion parameter estimation over multiple frames," in *Eur. Conf. on Computer Vision*, 1994.

[4] R. C. Bolles and H. H. Baker, "Epipolar plane image analysis: An approach to determining structure from motion," *Int. J. Comput. Vis.*, vol. 1, no. 1, pp. 7–55, 1987.

[5] P. Bouthemy, M. Gelgon, and F. Ganansia, "A unified approach to shot change detection and camera motion characterization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, pp. 1030–1044, July 1999.

[6] T. Darrel and A. P. Pentland, "Cooperative robust estimation using layers of support," *IEEE Trans. Pattern Recognit. Machine Intell.*, vol. 17, no. 5, pp. 474–87, 1995.

[7] G. H. Granlund and H. Knutsson, *Signal Processing for Computer Vision*. Norwell, MA: Kluwer, 1995.

[8] J. Huang, S. R. Kuma, M. Mitra, W. J. Zhu, and R. Zabih, "Spatial color indexing and applications," *Int. J. Comput. Vis.*, vol. 35, no. 3, pp. 245–68, 1999.

[9] M. Irani, B. Rousso, and S. Peleg, "Computing occluding and transparent motions," *Int. J. Comput. Vis.*, vol. 12, no. 1, pp. 5–16, Feb. 1994.

[10] M. Irani and P. Anandan, "Video indexing based on mosaic representation," *Proc. IEEE*, vol. 86, pp. 905–921, May 1998.

[11] B. Jähne, *Spatio-Temporal Image Processing: Theory and Scientific Applications*. New York: Springer-Verlag, 1991.

[12] P. Joly and H. K. Kim, "Efficient automatic analysis of camera work and microsegmentation of video using spatiotemporal images," *Signal Process.: Image Commun.*, no. 8, pp. 295–307, 1996.

[13] F. Liu, "Modeling Spatial and Temporal Texture," Ph.D. dissertation, MIT, Cambridge, MA, 1997.

[14] F. Liu and R. W. Picard, "Finding periodicity in space and time," in *Proc. IEEE Int. Conf. on Computer Vision*, 1998, pp. 376–383.

[15] G. Medioni, M. S. Lee, and C. K. Tang, *A Computational Framework for Segmentation and Grouping*. Amsterdam, The Netherlands: Elsevier, 2000.

[16] C. W. Ngo, T. C. Pong, and R. T. Chin, "Video partitioning by temporal slice coherency," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, pp. 941–953, Aug. 2001.

[17] ——, "Detection of gradual transitions through temporal slice analysis," in *Proc. Computer Vision and Pattern Recognition*, vol. 1, 1999, pp. 36–41.

[18] ——, "A robust wipe detection algorithm," in *Asian Conf. Computer Vision*, vol. 1, 2000, pp. 246–51.

[19] C. W. Ngo, T. C. Pong, and H. J. Zhang, "Motion-based video representation for scene change detection," *Int. J. Comput. Vis.*, vol. 50, no. 2, Nov. 2002.

[20] ——, "On clustering and retrieval of video shots," in *Proc. ACM Conf. on Multimedia*, 2001.

[21] C. W. Ngo, T. C. Pong, H. J. Zhang, and R. T. Chin, "Motion characterization by temporal slice analysis," *Comput. Vis. Pattern Recognit.*, vol. 2, pp. 768–773, 2000.

[22] N. V. Patel and I. K. Sethi, "Compressed video processing for cut detection," *Proc. Inst. Elect. Eng.*, vol. 143, no. 5, pp. 315–23, Oct. 1996.

[23] S. Peleg and J. Herman, "Panoramic mosaics by manifold projection," in *Proc. Computer Vision and Pattern Recognition*, 1997, pp. 338–343.

[24] H. S. Sawhney and S. Ayer, "Compact representations of videos through dominant and multiple motion estimation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, no. 8, pp. 814–830, 1996.

[25] E. P. Simoncelli, "Distributed Representation and Analysis of Visual Motion," Ph.D., MIT, 1993.

[26] H. Y. Shum and R. Szeliski, "Panoramic Image Mosaics, Tech. Rep.," Microsoft Research, MSR-TR-97-23, 1997.

[27] M. J. Swain and D. H. Ballard, "Color indexing," *Int. J. Comput. Vis.*, vol. 7, no. 1, pp. 11–32, 1991.

[28] Y. P. Tan, D. D. Saur, S. R. Kulkarni, and P. J. Ramadge, "Rapid estimation of camera motion from compressed video with application to video annotation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 133–46, Feb. 2000.

[29] A. M. Tekalp, *Digital Video Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1997, ch. 11.

[30] Y. Tonomura, A. Akutsu, K. Otsuji, and T. Sadakata, "Videomap and video spaceicon: Tools for anatomizing video content," in *Proc. INTERCHI*, 1993, pp. 131–136.

[31] J. Wang and E. Adelson, "Layer representation for motion analysis," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1993, pp. 361–366.

[32] B. L. Yeo and B. Liu, "On the extraction of DC sequence from MPEG compressed video," in *IEEE Int. Conf. on Image Processing*, vol. 2, Oct. 1995, pp. 260–63.

[33] B. L. Yeo, "Efficient processing of compressed images and video," Ph.D., Princeton Univ., Princeton, NJ, 1996.

[34] H. J. Zhang, A. Kankanhalli, and S. W. Smoliar, "Automatic partitioning of full-motion video," *ACM Multimedia Syst.*, vol. 1, no. 1, pp. 10–28, 1993.

Chong-Wah Ngo (M'02) received the Ph.D. degree from the Hong Kong University of Science & Technology (HKUST) in 2000, and the B.S. degree with honors in 1994 and M.S. in 1996, both in computer engineering, from Nanyang Technological University, Singapore.

Since 2002, he has been an Assistant Professor in the City University of Hong Kong (CityU). Before joining CityU, he was a Postdoctoral Visitor in Beckman Institute, University of Illinois at Urbana Champaign and was a Research Associate in HKUST. He was with Information Technology Institute, Singapore, in 1996, and was with Microsoft Research China as a Summer Intern in 1999. His current research interests include image and video indexing, computer vision, and pattern recognition.

Ting-Chuen Pong received the Ph.D. degree in computer science from Virginia Polytechnic Institute and State University, Blacksburg, in 1984.

In 1991, he joined the Hong Kong University of Science and Technology, where he is currently a Professor in computer science, Associate Vice-President for Academic Affairs, and Head of the W3C Office in Hong Kong. He served as an Associate Dean of Engineering from 1999 to 2002 and the Director of the Sino Software Research Institute at HKUST from 1995 to 2000. Before joining HKUST, he was an Associate Professor in Computer Science at the University of Minnesota, Minneapolis, MN. His research interests include computer vision, pattern recognition, multimedia computer, and IT in education.

Dr. Pong is a recipient of the Annual Pattern Recognition Society Award in 1990. He has served as Program Co-Chair of the Web and Education track of the Tenth International World Wide Web Conference in 2001, the Third International Computer Science Conference in 1995, and the Third Asian Conference on Computer Vision in 1998.

Hong-Jiang Zhang (M'91–SM'97) received the Ph.D degree from the Technical University of Denmark and the B.S. from Zheng Zhou University, China, both in electrical engineering, in 1982 and 1991, respectively.

From 1992 to 1995, he was with the Institute of Systems Science, National University of Singapore, where he led several projects in video and image content analysis and retrieval and computer vision. He also worked at MIT Media Lab in 1994 as a Visiting Researcher. From 1995 to 1999, he was a Research Manager at Hewlett-Packard Labs, where he was responsible for research and technology transfers in the areas of multimedia management, intelligent image processing, and Internet media. In 1999, he joined Microsoft Research, China, where he is currently a Senior Researcher and the Assistant Managing Director mainly in charge of media computing and information processing research. He has authored three books, over 120 referred papers and book chapters, seven special issues of international journals in multimedia processing, content-based media retrieval, and Internet media, as well as numerous patents or pending applications. He currently serves on the editorial boards of five professional journals and a dozen committees of international conferences.

Dr. Zhang is a member of ACM.