Shot Change Detection via Local Keypoint Matching

Chun-Rong Huang, *Member, IEEE*, Huai-Ping Lee, and Chu-Song Chen, *Member, IEEE*

***Abstract*—Shot change detection is an essential step in video content analysis. However, automatic shot change detection often suffers from high false detection rates due to camera or object movements. To solve this problem, we propose an approach based on local keypoint matching of video frames. This approach aims to detect both abrupt and gradual transitions between shots without modeling different kinds of transitions. Our experiment results show that the proposed algorithm is effective for most kinds of shot changes.**

***Index Terms*—Invariant local feature, matching, recognition, shot change detection.**

I. Introduction

THE rapid development of storage and multimedia technologies has made the retrieval and processing of videos relatively easy. Temporal segmentation is a fundamental step in video processing, and shot change detection is the most basic way to achieve it. However, while *hard cuts* (abrupt transitions) can be easily detected by finding changes in a color histogram, gradual transitions such as *dissolves*, *fades*, and *wipes* are hard to locate.

Many shot change detection studies focus on finding low-level visual features, e.g., color histograms and edges, and then locate the spots of changes in those features. For example, Zabih *et al.* [1] used the disappearance and appearance of outgoing and incoming edges to detect scene breaks. Motion estimation techniques, such as optical flow, are also used to find transitions [2], since shot changes imply motion changes. Bouthemy *et al.* [3] measured the number of pixels that belong to the part undergoing dominant motion to predict shot changes. Gargi *et al.* [4] investigated the efficacy of several color histogram-based measures for cut detection, and concluded that the histogram-intersection measure [5] is the most effective method. Shen and Delp [6] used the histogram differences of DCT coefficients to detect hard cuts. Surveys of early approaches can be found in [4] and [7].

The above approaches are useful for hard cuts, but they are prone to error when detecting gradual changes because they are very sensitive to object or camera movements. Ngo *et al.* [8] proposed a novel video segmentation method that used spatio–temporal slices to recognize camera motion, zooming, hard cuts, and so on. Bescós [9] combined deterministic (e.g., the sum of absolute differences), statistical parametric (e.g., likelihood ratios) and statistical nonparametric (e.g., Pearson's homogeneity test) metrics and applied them to DC images to detect abrupt and gradual transitions. Boccignone *et al.* [10] proposed an interesting method based on the observation of the human eyes when making comparisons between two pictures. Human eyes focus on a certain object in a particular order when comparing pictures. This is called Focus of Attention (FOA). In [10], the authors tried to find FOA sequences by calculating *saliency maps* for each frame, and then detected changes in the FOA sequences. Yeo and Liu [11] extracted the DC sequences from MPEG compressed videos and used three detection schemes to distinguish hard cuts, dissolves, and flashlights. Cernekova *et al.* [12] used mutual information (MI) to measure information transported from one frame to another. Abrupt transitions and fades between two shots lead to a low level of MI. To distinguish fades from abrupt transitions, the authors further exploited joint entropy as interframe information. This approach achieves an impressive performance on shot change detection, but it can only be used for cases of abrupt transitions and fades. Park *et al.* [13] presented a method for shot change detection by using the scale invariant feature transform (SIFT) [14]. This method can detect transitions by matching neighbor video frames but still suffers from fast object motions or sudden lighting changes as mentioned in [13]. In addition, the computation load of SIFT causes difficulty of building an efficient detector.

While many shot detection methods consider both abrupt and gradual transitions, some researchers have focused on detecting gradual transitions only. For example, Wu *et al.* [15] detected horizontal and vertical wipes by computing pixel-wise DC coefficient differences between continuous I and P frames; and Pei and Chou [16], [17] employed the macroblocks of P and B frames to detect dissolves and wipes in MPEG videos. Fernando *et al.* [18] analyzed the linear and quadratic behavior of dissolves and fades. Based on the results, they computed the ratio between the second derivative of the variance curve and the first derivative of the mean curve to identify dissolves and fades. They also analyzed line diagrams of common wipe transitions [19], [20]. Nam and Tewfik [21] used B-spline to fit each pixel of a sequence, and considered time instants with a high interframe standard deviation and a low fitting error as candidates for locating transition intervals. Lienhart [22] proposed a multiresolution method for time series analysis, and then applied pattern classification techniques to train a model for dissolve detection. Recently, Su *et al.* [23] utilized the monotonicity of intensity changes during transitions to detect dissolves. They used a binomial distribution model to distinguish dissolves from motions so that their algorithm can tolerate fast motions.

In our approach, we consider the original definition of a shot. A shot usually contains a series of interrelated frames taken consecutively by a single camera and therefore represents a contin-

Manuscript received December 29, 2006; revised April 14, 2008 and November 20, 2007. Current version published October 24, 2008. This work was supported by the National Science Council of Taiwan, R.O.C., under Grant NSC 96-3113-H-001-011. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Ling Guan.

C.-R. Huang and C.-S. Chen are with the Institute of Information Science, Academia Sinica, Taipei, Taiwan, R.O.C. (e-mail: nckuos@iis.sinica.edu.tw; song@iis.sinica.edu.tw).

H.-P. Lee is with the Department of Computer Science, University of North Carolina, Chapel Hill, NC 27514 USA (e-mail: lhp@cs.unc.edu).

Digital Object Identifier 10.1109/TMM.2008.2001374

1520-9210/$25.00 © 2008 IEEE

uous action in time and space. Instead of comparing the low-level feature differences, the most intuitive approach to shot change detection is to recognize objects and scenes. If the same objects or scenes appear in consecutive frames, we may consider that there is no shot change. We propose a new unified approach that detects shot changes based on this concept, instead of changes in some low-level features. If we can match the same objects in two adjacent frames, there should not be any transitions between them. With object tracking techniques, we can minimize the influence of object and camera motion; therefore, we can detect both abrupt transitions and gradual transitions. In the following sections, we present the proposed feature matching method, as well as our algorithm for finding shot changes.

II. Feature Matching for Finding the Correspondence Between Images

The goal of feature matching is to match points on the same object in multiple images. Image correspondence obtained in this manner is useful in several fields of computer vision and image processing, such as object recognition, 3-D structure reconstruction from images, image retrieval, building of panoramas, and augmented reality. To reduce the ambiguities inherent in matching, points to be matched must have some distinctive features that distinguish them from other points; moreover, a feature must be invariant to transformations, such as translation, rotation, and scaling, so that the object can still be detected after it moves.

Many works have addressed the problem of finding robust feature points and reliable image correspondence. Zhang *et al.* [24] showed how to match Harris corners [25] over a large image range. They used a correlation window around each corner to select likely matches, and then removed outliers by using geometric constraints derived from epipolar geometry. Although highly accurate image correspondence can be achieved by this approach, the assumption that objects in images follow a single rigid motion restricts the technique's application to general matching.

Instead of finding image correspondence based on the rigid-motion assumption, some recent approaches have tried to find accurate matches by identifying distinctive features and descriptors. Schmid and Mohr [26] used rotationally invariant descriptors of the local image regions to match Harris corners. This method allows features to be matched under arbitrary orientation changes, but it is still sensitive to image scale changes. Lowe [14] proposed the SIFT descriptor that is invariant to both scale and rotation. Under this approach, keypoints (salient corners) are computed through the detection of scale-space extremes in a series of difference-of-Gaussian (DoG) images. Local descriptors are then built for each keypoint based on a weighted histogram of edge orientations from a patch of pixels in the keypoint's local neighborhood. In [27], it has been shown that SIFT is one of the most effective approaches when scale and viewpoint changes occur. Various extensions of SIFT have been proposed. For example, Ke and Sukthankar introduced PCA-SIFT [28], which applies principal components analysis (PCA) to a normalized gradient patch. The gradient location-orientation histogram (GLOH) [27] computes the SIFT descriptor for a log-polar location grid and then reduces the size of the descriptor with PCA. The primary focus of these extensions is to provide more distinctive and compact descriptors in order to improve the matching accuracy and processing speed. Instead of using edge orientations, Huang *et al.* [29], [30] proposed using the contrast context histogram (CCH), which is more efficient to compute, to find image correspondence. We use CCH to detect shot changes because its matching accuracy is comparable to that of SIFT, but it requires much less computation time, as shown in [29], [30]. We discuss CCH in detail in the Section II.A.

To our best knowledge, the work in [13] is the only one in the past that employed local keypoint matching for shot change detection. This method used SIFT to find image correspondence, and applied a fixed threshold to the number of matched points of neighboring video frames to find the transitions. Our method differs from this method in several aspects. First, our method does not rely on a fixed threshold of the number of matched points; the threshold applied in our method is varied with the local maxima and minima of the number of matches, which can handle the variations of transitions better. Second, we do not match neighboring frames or frames apart from a fixed period only, but also match nonadjacent frames inferred by shot-change interval estimation, which can further increase the detection accuracy. Third, our method can find both the shot boundaries and the transition intervals of shots. Details of our method can be found in Sections II-B, II.- and III. We have compared our method with that in [13] in the experimental results of Section IV-D, and show that our method performs considerably better.

A. Contrast Context Histogram

The main issue in developing invariant local descriptors is how to represent a region effectively and discriminatively. The color histogram [31] is one option for textural description, but it is sensitive to illumination changes. Instead, we consider a technique that computes the contrast values of points within a region with respect to a salient corner. We assume that many salient keypoints (salient corners) have already been extracted from an image I. For each keypoint $\boldsymbol{p}_c$ at the image coordinate (u_c, v_c), we locate an $n \times n$ local region $\boldsymbol{R}$ surrounding $\boldsymbol{p}_c$. Let $\boldsymbol{p}$ denote a pixel at the image coordinate (u, v) in $\boldsymbol{R}$. We compute the contrast value $C(\boldsymbol{p})$ of $\boldsymbol{p}$ in $\boldsymbol{R}$ as

$$C(\boldsymbol{p}) = I(\boldsymbol{p}) - I(\boldsymbol{p}_c) \tag{1}$$

where $I(\boldsymbol{p})$ and $I(\boldsymbol{p}_c)$ are the intensity values of $\boldsymbol{p}$ and $\boldsymbol{p}_c$, respectively. We then construct a descriptor of $\boldsymbol{p}_c$ based on these contrast values, and separate $\boldsymbol{R}$ into several nonoverlapping regions, $\boldsymbol{R}_1, \boldsymbol{R}_2, \ldots, \boldsymbol{R}_t$. Without lost of generality, we use a log-polar coordinate system (r, θ) to perform the division, as shown in Fig. 1. The system, which has been used in a number of previous works [27], [32], is more sensitive to the positions of points close to the center than to those of points farther away. To ensure that the descriptor is invariant to image rotations, the direction of $\theta = 0$ in the log-polar coordinate system is set to coincide with the edge orientation of $\boldsymbol{p}_c$.

Given the importance of representing a subregion $\boldsymbol{R}_i$ efficiently and discriminatively, we consider a histogram-based representation because a histogram is relatively insensitive to nonuniform deformations of a region. An intuitive way to employ the histogram feature is to gather the contrast values in

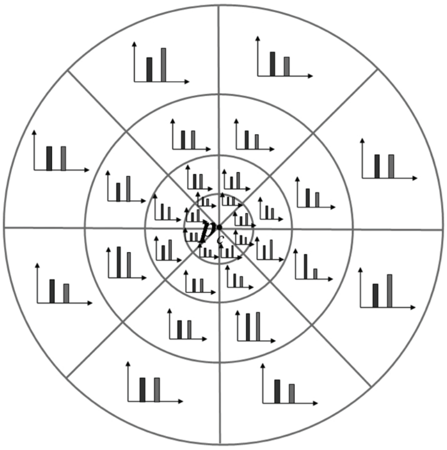


Fig. 1. Log-polar diagram of the CCH descriptors. The center of the coordinate is the salient point $\boldsymbol{p}_c$.

a subregion into a histogram bin. However, summations of positive and negative contrast values may reduce the discriminating response of the bin. Thus, to improve the discriminative ability of the descriptor, we use both positive and negative histogram bins of contrast values for each subregion, as described in the following.

For the subregion $\boldsymbol{R}_i$, we define the positive contrast histogram bin respective to $\boldsymbol{p}_c$ as

$$H_{\boldsymbol{R}_i+}(\boldsymbol{p}_c) = \frac{\sum\{C(\boldsymbol{p}) \,|\, \boldsymbol{p} \in \boldsymbol{R}_i \text{ and } C(\boldsymbol{p}) \geq 0\}}{\#_{\boldsymbol{R}_i+}} \tag{2}$$

where $\#_{\boldsymbol{R}_i+}$ is the number of positive contrast values in $\boldsymbol{R}_i$. In a similar manner, the negative contrast histogram bin is defined as

$$H_{\boldsymbol{R}_i-}(\boldsymbol{p}_c) = \frac{\sum\{C(\boldsymbol{p}) \,|\, \boldsymbol{p} \in \boldsymbol{R}_i \text{ and } C(\boldsymbol{p}) < 0\}}{\#_{\boldsymbol{R}_i-}} \tag{3}$$

where $\#_{\boldsymbol{R}_i-}$ is the number of negative contrast values in $\boldsymbol{R}_i$.

By combining the contrast histograms of all the subregions into a single vector, the CCH descriptor of $\boldsymbol{p}_c$ in association with its local region $\boldsymbol{R}$ can be defined as follows:

$$\text{CCH}(\boldsymbol{p}_c) = (H_{\boldsymbol{R}_1+}, H_{\boldsymbol{R}_1-}, \ldots, H_{\boldsymbol{R}_t+}, H_{\boldsymbol{R}_t-}). \tag{4}$$

In [29], [30], the CCH descriptor was evaluated by using a large set of images undergoing various geometric and photometric transformations. The evaluation results show that the CCH descriptor is computationally efficient and highly accurate in determining feature correspondence.

B. Locating Transitions by Matching Adjacent Frames

The first part of our algorithm locates the time instants at which shot changes take place. We observe that objects or scenes are replaced during transitions, even though they may be moving or rotating within the shot. Most methods of shot change detection produce many false alarms when objects or cameras move, as they can only detect changes in some overall features between the same image locations of adjacent frames in a video. Although such features will change dramatically during transitions, they will also change when something moves in a single shot. The advantage of feature matching is that it is invariant to affine transformations; thus, we can even match objects after they have moved. An additional advantage is that we do not have to design a detector for each kind of transition. Since a shot change, i.e., a transition, indicates a change of objects in the scene, multiple kinds of shot changes can be detected in a unified manner.

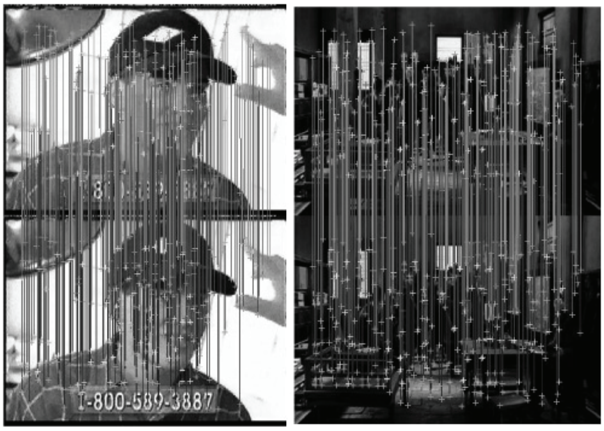


Fig. 2. Some feature matching results between adjacent frames.

In our algorithm, each frame is preprocessed by keypoint detectors. The keypoints are extracted by detecting Harris corners [25] at each level of a multiscale Laplacian pyramid [33]. At each level of the pyramid, the Harris corner detector is used to localize points on the 2-D image plane. Then, a salient keypoint is selected by detecting the local maxima in a 7×7 region. Fig. 1 illustrates the contrast context histogram of a salient keypoint $\boldsymbol{p}_c$ under the log-polar coordinate system. A local region $\boldsymbol{R}$ is divided into several subregions by quantizing r and θ of the log-polar coordinate system. For each subregion, a 2-bin contrast histogram, introduced above, is constructed. A CCH descriptor of $\boldsymbol{p}_c$ is then computed as follows:

$$\text{CCH}(\boldsymbol{p}_c) = \left(H_{r_0\theta_0+}, H_{r_0\theta_0-}, \ldots, H_{r_k\theta_{l-1}+}, H_{r_k\theta_{l-1}-}\right) \tag{5}$$

where $r_i = 0, \ldots, k$, $\theta_j = (2\pi/l)m$, $m = 0, \ldots, l-1$, and $\text{CCH}(\boldsymbol{p}_c) \in R^{2(k+1)l}$. In our implementation, we used $k = 3$ and $l = 8$, resulting in a 64-dimension descriptor, as illustrated in Fig. 1.

We produce lists of keypoints and their local descriptor vectors for each frame. Then, we perform keypoint matching for each pair of adjacent frames, which yields a 1-D signal of numbers of the matched points. Formally, for the i-th frame F_i, there is a list of keypoints, key_i, comprised of the locations and the 64-dimensional descriptor vectors of the keypoints found in F_i. The matching between key_i and key_{i+1} is based on the nearest neighbor method, and the distance between two keypoints is defined as the included angle of the corresponding 64-dimension descriptors. Each keypoint $\text{key}_{i,j}$ in F_i is matched to the keypoint $\text{key}_{i+1,k}$ in F_{i+1} that has the shortest distance to $\text{key}_{i,j}$. However, if the shortest distance is longer than a predefined threshold, the keypoint $\text{key}_{i,j}$ is not matched to any keypoint. In addition, we assume that the camera or object motion is continuous between adjacent frames in a single shot, and hence we discard the match if the image locations of $\text{key}_{i,j}$ and $\text{key}_{i+1,k}$ are far apart (larger than 20 pixels). Fig. 2 shows some examples of feature matching results for adjacent frames.

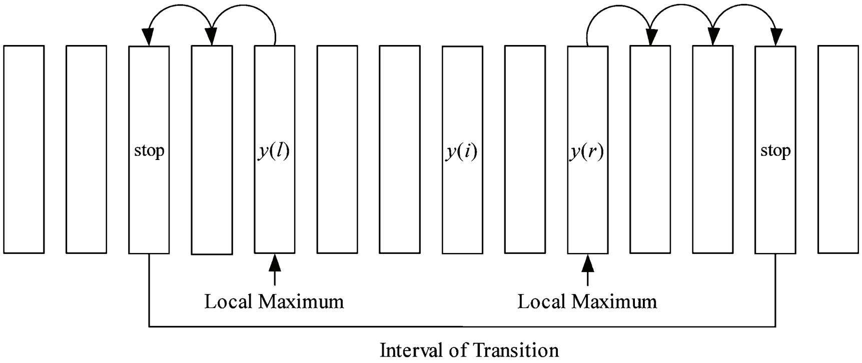


Fig. 3. Our algorithm for determining intervals.

Each frame F_i is related to a value $y(i)$, i.e., the number of matched points between F_i and F_{i+1}. If a shot change occurs in F_i, the keypoints in F_i should only have a few matched keypoints in F_{i+1}, since most of the objects that appear before the shot change should be replaced after the transition. Therefore, we assume that a shot change takes place when there is a salient local minimum in the values of $y(i)$, so the problem is reduced to finding the minima of $y(i)$.

The following is the formal formulation of our initial algorithm for finding the minima.

```
1. For each local minimum $y(t)$ that satisfies
       $y(t) < y(t-1)$ and $y(t) < y(t+1)$:
       1.1 Find the local maxima $y(l)$ and $y(r)$
       on the left and right of $y(t)$.
       Let $M$ be the maximum in $\{y(i) \,|\, l \leq i \leq r\}$.
       1.2 If $M - y(t) > T_r \times M$, where $T_r$ is a parameter
       selected within the range $[0, 1]$,
       then $y(t)$ is a candidate.
2. For each candidate found in Step 1:
       2.1 If $y(t) > T_{hi}$, the candidate is discarded.
       2.2 If $M < T_{lo}$, the candidate is discarded.
```

In the first step, the maxima $y(l)$ is the maxima found just to the left of $y(t)$, where $y(r)$ is the maxima found just to the right of $y(t)$. We only keep the minima that are less than a fraction $(T_r = 0.49)$ of the maximum M in $\{y(i) \,|\, l \leq i \leq r\}$. In the second step, we eliminate candidates with values larger than a threshold $T_{hi} = 185$ because there are still many matching keypoints; thus, the two adjacent frames are still considered similar. When M is too small (measured by $T_{lo} = 15$), it is not representative enough for comparison, so the corresponding minimum is also discarded.

C. Intervals of Transitions

The candidates found with local minima are only time instants, not intervals. However, since many shot changes are gradual transitions, it is necessary to find the intervals of such transitions. Our method for finding the intervals is also based on feature matching. Shot changes are likely to occur when the number of matched objects decreases; thus, there should not be any transitions when several objects in adjacent frames are matched. In our method, the local maxima to the left and right of the candidate transition are possible start and end frames of that transition. We add another condition: the video sequence before and after the shot change should also be "stable," resulting in stable numbers of matched keypoints. Hence, the search for start and end points begins with the two maxima and continues until the number of matched keypoints is stable.

For a given candidate $y(i)$, we first locate the nearest maxima $y(l)$ and $y(r)$ on its left and right respectively. To find the first frame of the transition, we perform keypoint matching between adjacent frames, starting with F_l and F_{l-1}, in the reverse order of the video, until the number of matched keypoints becomes stable. In other words, we perform matching between F_l and F_{l-1}, F_{l-1} and F_{l-2}, F_{l-2} and F_{l-3}, and so on to generate the numbers of matched keypoints, $y'(l), y'(l-1)$, $y'(l-2)$, etc. When the difference between $y'(i)$ and $y'(i{-}1)$ is small enough, the process stops, and F_i is considered as the first frame of the transition. The last frame of the transition is found in a similar manner by starting with the matching step between F_r and F_{r+1}. Fig. 3 illustrates our algorithm for finding intervals.

III. Reducing False Alarms by Matching Nonadjacent Frames

The above approach, which is based on local-minimum analysis, provides an efficient initial step for detecting transition candidates. However, since only correspondence between adjacent frames are employed, false detection may occur when the video is affected by certain changes, such as sudden lighting changes, occlusions, and fast object motions. In this section, we perform fine selection of shot changes by examining the intervals of transitions to remove cases of false detection.

Matching nonadjacent frames provides richer image correspondence information, but exhaustively matching a large number of pairs of frames within an interval is very time consuming. Since variations in a shot usually continue for a limited period of time, we match the frames before and after the intervals of candidate shot changes. If the number of matched CCH features between the first and last frames of an interval is relatively high, it indicates that the same objects remain visible; thus, the candidate transition detected initially is a false alarm and should be deleted.

Next, we describe the steps of our algorithm for fine selection of shot changes. After finding the intervals of transitions in the initial step, discussed in Section II, the first and the last frames of each interval are matched again. If there are still many matching keypoints, the two frames are considered similar because the detected transition is probably a false alarm. Specifically, let $\pi = [t_{\text{start}}, \ldots, t_{\text{end}}]$ be a time interval of a transition. F_{start} and F_{end} are the start and end frames of this transition,

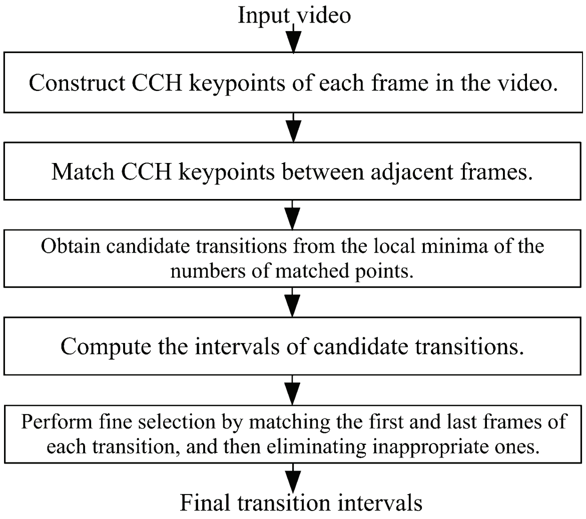
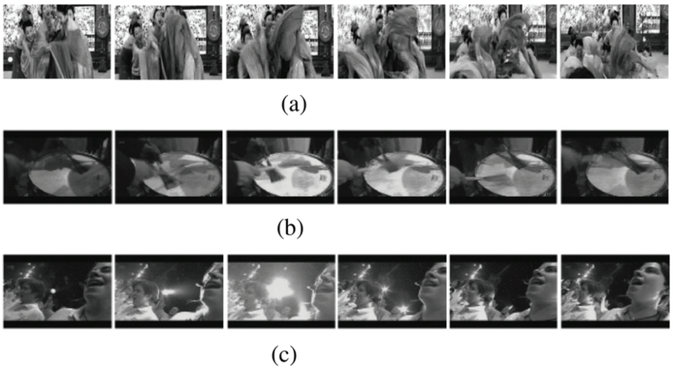

Input video
Construct CCH keypoints of each frame in the video.
Match CCH keypoints between adjacent frames.
Obtain candidate transitions from the local minima of the numbers of matched points.
Compute the intervals of candidate transitions.
Perform fine selection by matching the first and last frames of each transition, and then eliminating inappropriate ones.
Final transition intervals

Fig. 4. Complete algorithm of our approach.


(a)

(b)

(c)

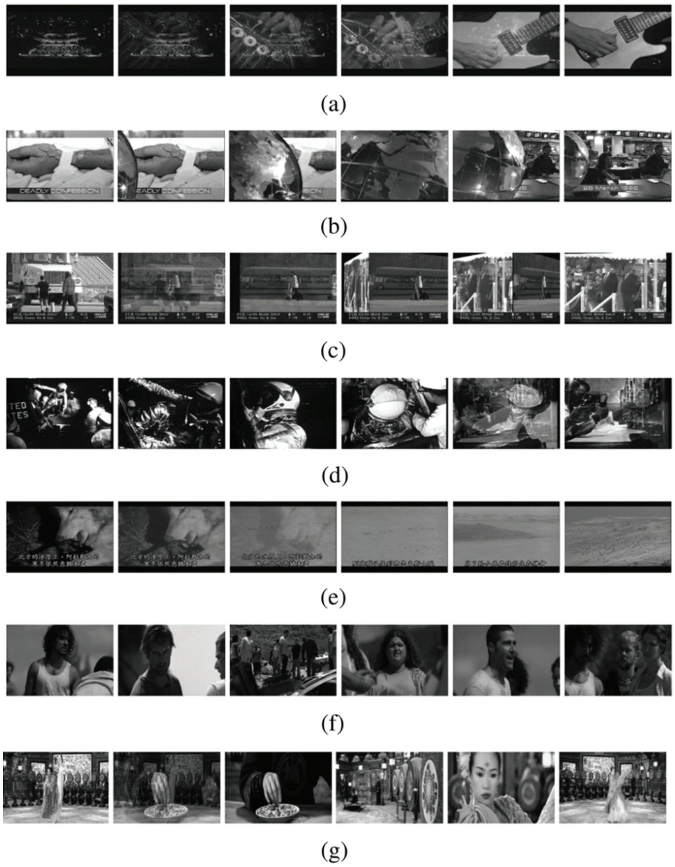
Fig. 5. Sample frames of some easily mis-detected shots. (a) Fast multiple object motions, (b) sudden lighting changes, and (c) spot lights.

respectively. We compute the number of matched keypoints N_π between F_{start} and F_{end}, and let μ_π be the average number of matched keypoints between adjacent frames,

$$\mu_\pi = \frac{\sum_{t_{\text{start}}}^{t_{\text{end}}} y(t)}{t_{\text{end}} - t_{\text{start}}}. \tag{6}$$

If $N_\pi > T_e \times \mu_\pi$, where T_e is a fractional threshold (0.11 in our experiments), the transition π is removed. In the last step, if two adjacent transitions are too close to each other (only one or two frames apart), they are merged because it would be unreasonable to change to a new scene for only one or two frames. The steps of our algorithm are presented in Fig. 4.

With the initial adjacent-frame matching and subsequent examination of transition intervals, our algorithm can deal with difficult cases that involve high content motions or variations. For example, Fig. 5(a) shows a shot in which some women are tracked as they fall on the floor. The motions of the camera and the women result in multiple blurred moving objects, but our algorithm still successfully identifies the associated frames as being within a single shot. Figs. 5(b) and (c) show two cases that are affected by lighting and color changes; sudden light in a dark environment will change the visibility of objects. In addition, spot lights in a bright scene will cause over-saturation and some objects will be occluded by brilliant rays. These cases are apt to be wrongly characterized as transitions by existing methods, but our algorithm can discard them successfully.


(a)

(b)

(c)

(d)

(e)

(f)

(g)

Fig. 6. Sample frames in our test set. (a) Dissolve, (b) News 1 (19980328_ABC), (c) News 2 (19980326_CNN), (d) Documentary 1 (ANNI005), (e) Documentary 2 (Return of the Caribou), (f) TV Serial (Lost), and (g) Movie (House of Flying Daggers).

IV. Experiment Results

A. *Test Set and Results*

To evaluate the proposed method, we use seven video sequences in our experiments (Fig. 6). The numbers of frames and transitions in the videos are summarized in Table I. The video “Dissolve,” from [22], consists of clips from a concert and several TV commercials; thus, it includes a lot of dissolve transitions. The two news clips are from the ABC and CNN, respectively. We tested them because wipe transitions are often seen in news previews, but they are seldom seen in other video genres. The movie “House of Flying Daggers” includes many complex dance and fight scenes, so it is good material for testing object recognition and the effect of motion blur. The TV serial “Lost” also has many scenes with fast motion and blur. The other two test videos are documentaries from the Open Video Project and

TABLE I
TEST SEQUENCES

Name	Length (frames)	Hard Cut	Dissolve	Fade	Wipe	Total Transitions
Dissolve [22]	25262	140	276	11	0	427
News 1 (19980328_ABC)	23642	116	31	9	10	166
News 2 (19980326_CNN)	10789	17	13	1	7	38
Documentary 1 (ANNI005)	11321	37	29	0	0	66
Documentary 2 (Return of the Caribou)	41358	83	122	2	0	207
TV Serial (Lost)	30706	297	0	1	0	298
Movie (House of Flying Daggers)	31236	14	372	0	0	386
Total	174314	704	843	24	17	1588

Discovery Channel, respectively. They have more static scenes, but there is still a lot of motion when animals are being tracked.

To evaluate the performance of the proposed algorithm, we use the recall and precision metrics, which are defined as follows:

$$\text{Recall} = \frac{H}{H+M} \tag{7}$$

$$\text{Precision} = \frac{H}{H+F} \tag{8}$$

where H, M, and F are the numbers of hits, miss detects, and false alarms, respectively. For example, the recall and precision of the Documentary 2 (Return of the Caribou) video listed in Table I are $(H)/(207)$ and (H/T), respectively, where H is the number of shot changes successfully detected among the 207 transitions, and $T(= H+F)$ is the total number of shot changes detected.

Another measure, the Q value [34], is used to evaluate the combination of recall and precision:

$$Q = \text{Recall} \times \text{Precision}. \tag{9}$$

If $Q = 1$, the recall and precision values are both 1. Since our algorithm can also predict intervals of transitions, we need another measure to evaluate the correctness of each predicted interval of a gradual transition. In our experiment, we use the weighted overlap coefficient (WOC) proposed by Nam and Tewfik [21]. Suppose $\text{Int}_{\text{act}} = [t_{\text{start}}, \ldots, t_{\text{end}}]$ is the actual time interval of a transition, and t_{mid} is the midpoint, i.e., $t_{\text{mid}} = (t_{\text{start}} + t_{\text{end}})/2$. Let $\text{Int}_{\text{pre}}[t'_{\text{start}}, \ldots, t'_{\text{end}}]$ be the predicted corresponding time interval. The WOC, which depends on the length of the overlapping region, gives a weight value $z(t)$ to each frame in the actual interval. The frames near the midpoint of the interval are usually more important than those near the beginning and ending points. Therefore, $z(t)$ is symmetric; it peaks at t_{mid} and decreases linearly as t moves away from t_{mid}. The sum of $z(t)$ in the interval is 1. Also, note that the slope of $z(t)$ equals 1 in the first half of the transition and -1 in the second half. The weighted overlap coefficient is then calculated as follows (see Fig. 7):

$$\text{WOC} = \sum_{t=t'_{\text{start}}}^{t'_{\text{end}}} z(t). \tag{10}$$

The coefficient considers the length of the overlapping intervals relative to the actual length of the transition. However, in a long transition, a long overlapping period does not necessarily yield a high WOC value. Overlapping in the middle of the transition is also important.

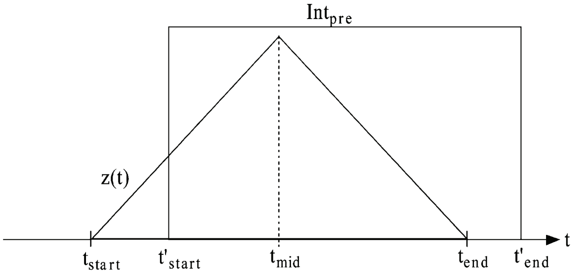
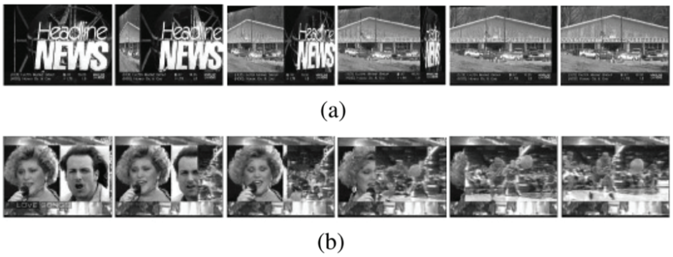


Fig. 7. Weighting function $z(t)$.


(a)

(b)

Fig. 8. Some wipe examples. (a) Cubic wipe and (b) wipe with motions.

TABLE II
EXPERIMENT RESULTS

Name	Recall	Precision	Q	WOC
Dissolve	95.08%	98.54%	93.70%	97.23%
News 1	96.99%	95.27%	92.40%	99.57%
News 2	97.37%	100.00%	97.37%	90.87%
Documentary 1	86.36%	100.00%	86.36%	99.37%
Documentary 2	94.20%	78.95%	74.37%	99.18%
TV Serial	95.64%	92.53%	88.50%	100.00%
Movie	97.41%	95.67%	93.20%	99.73%
Average	95.53%	93.47%	89.29%	98.69%

The average recall and precision in our experiments are 95.53% and 93.47%, respectively, and the average Q value is 89.29%. The results for all video clips are reported in Table II. From the table, we observe that our method is effective for many kinds of shot changes. Even for wipe, which is a very difficult case, our method still achieves over eighty-percent accuracy (Tables VI and VII). Fig. 8 shows some wipe examples that our method detected successfully; Fig. 8(a) shows a cubic wipe and Fig. 8(b) is a wipe containing multi split screens and fast object motions.

With regard to the sensitivity of the parameters, there are two main thresholds, T_r and T_e; the former relates to the search of local minima and the later determines the transition intervals.

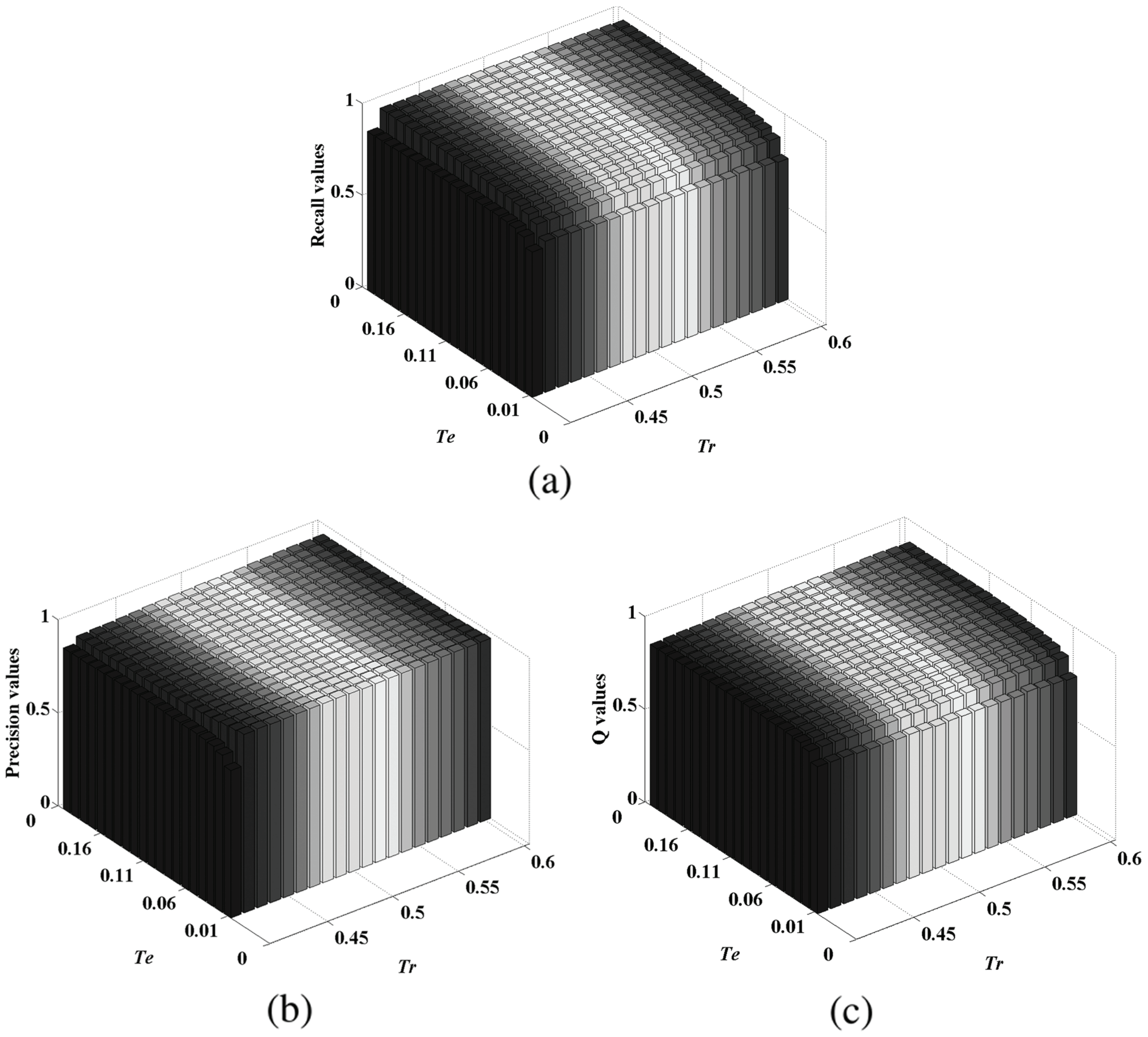


Fig. 9. Recall, Precision and Q values obtained by varying the thresholds T_r and T_e. (a) Recall values, (b) precision values, and (c) Q values.

TABLE III
AVERAGE COMPUTATION TIME FOR EACH STEP OF OUR ALGORITHM

Operation	Time (ms)
Constructing keypoints	14.23
Matching keypoints	10.63
Searching transition instants	0.01
Finding transition intervals	0.01
Examining transition intervals	2.19
Total computation time per frame	27.07

TABLE IV
AVERAGE CONSTRUCTION AND MATCHING TIME IN MS FOR KEYPOINTS IN EACH TEST VIDEO

Name (Resolutions)	Construction	Matching	Construction +Matching
Dissolve (352×240)	11.65	5.73	17.38
News 1 (352×264)	16.70	13.86	30.56
News 2 (352×264)	16.68	11.67	28.35
Documentary 1(320×240)	12.47	7.82	20.29
Documentary 2(352×240)	12.11	6.54	18.65
TV Serial (352×240)	10.39	5.19	15.58
Movie (352×240)	16.74	17.02	33.76
Average	14.23	10.63	24.86

We investigated the effects of the parameters by varying their values. The recall, precision, and Q values obtained by varying T_r and T_e are shown in Fig. 9. From these figures, we observe that the results are not sensitive to the values of the parameters, unless the values are very small.

In terms of computational speed, our method takes 0.027 s to process a frame on an Intel Pentium 4 computer with a 3.4G CPU and 768M memory. Table III shows the averaged computation time in milliseconds for each step of our algorithm, and Table IV shows the averaged construction and matching time for the keypoints of a frame. It can be seen from Table IV that the time varies with the content of the videos. Because the two news clips and the movie sequences contain relatively complex scenes, more keypoints are detected. As a result, they require longer construction and matching time.

TABLE V
AVERAGE RECALL AND PRECISION RATES BEFORE AND AFTER FINE SELECTION OF SHOT CHANGES

	Recall	Precision
Before	96.22%	79.62%
After	95.53%	93.47%

The average WOC value of 98.69% demonstrates that our algorithm for finding intervals is also reliable. We further investigate the results of the initial stage and the fine selection stage of the algorithm. Table V shows the recall and precision values of the shot changes before and after fine selection. The precision increases by about 14 percent, which demonstrates that the fine selection stage improves the overall performance.

B. Performance on Each Type of Transition

Having discussed the performance of our method for general shot change detection, we now evaluate its performance on each type of transition. Strictly speaking, since our method does not distinguish between the types of transitions, the recall of each type can be well formed; however, the precision can not be defined precisely. We thus formulate a measure that approximately estimates the specific type of precision of our approach. The evaluations of both metrics are presented below.

To evaluate the recall of a particular type of transition, we use the following equation:

$$\text{Recall}_{\text{type}} = \frac{H_{\text{type}}}{H_{\text{type}} + M_{\text{type}}} \tag{11}$$

where $\text{type} \in \{\text{hard} - \text{cut}, \text{ dissolve}, \text{ fade}, \text{ wipe}\}$, H_{type} is the number of hits and M_{type} is the number of miss detects for the particular type of transition. For example, in Table I, the recall of the hard-cut transitions of Documentary 2 (Return of the Caribou) is $(H_{\text{hard-cut}})/(83)$, where $H_{\text{hard}-\text{cut}}$ is the number of hard cuts successfully detected among 83 hard-cut transitions.

To evaluate the precision, we have to compute F_{type}, which is the number of false alarms for a particular type of transition, as follows:

$$\text{Precision}_{\text{type}} = \frac{H_{\text{type}}}{H_{\text{type}} + F_{\text{type}}} \tag{12}$$

where $\text{type} \in \{\text{hard} - \text{cut}, \text{ dissolve}, \text{ fade}, \text{ wipe}\}$.

In methods devoted to detecting a particular kind of transition, such as those compared in Section IV.C, the above precision evaluation is defined straightforwardly. However, since our approach is a general technique that can detect multiple kinds of transitions simultaneously, the hits of some transitions will be false alarms of other types of transitions. For example, if a transition detected by our method belongs to the hard-cut type, it will be treated as a false alarm of the dissolve type in the computation of F_{dissolve}.

TABLE VI
RECALL FOR EACH KIND OF SHOT TRANSITION

Name	Hard Cut	Dissolve	Fade	Wipe
Dissolve	98.57%	93.48%	90.91%	N/A
News 1	99.14%	93.55%	100.00%	80.00%
News 2	100.00%	100.00%	100.00%	85.71%
Documentary 1	94.59%	75.86%	N/A	N/A
Documentary 2	97.62%	91.74%	100.00%	N/A
TV Serial	95.62%	N/A	100.00%	N/A
Movie	100.00%	97.31%	N/A	N/A
Average	97.16%	94.42%	95.83%	82.35%

TABLE VII
PRECISION FOR EACH KIND OF SHOT TRANSITION

Name	Hard Cut	Dissolve	Fade	Wipe
Dissolve	98.59%	98.52%	98.48%	N/A
News 1	95.36%	95.10%	95.40%	94.32%
News 2	100.00%	100.00%	100.00%	100.00%
Documentary 1	100.00%	100.00%	N/A	N/A
Documentary 2	79.73%	78.36%	79.92%	N/A
TV Serial	92.53%	N/A	92.83%	N/A
Movie	95.78%	95.67%	N/A	N/A
Average	92.95%	93.82%	95.17%	96.67%

A better way to evaluate the precision of each type is to use F, instead of F_{type}, in (12), so that the hits of the other types of transitions are not counted in the estimation of false alarms, where F is the number of false alarms used in (8), generated by our approach for general transition detection. Nevertheless, this evaluation is still not quite even, because all the false alarms are due to the misdetection of a particular type of transition; hence the precision is generally underestimated. For a general approach like ours, which does not distinguish between transition types, we define the precision of each type as follows:

$$\text{Precision}{*}_{\text{type}} = \frac{H_{\text{type}}}{H_{\text{type}} + (\#\text{type}/\#\text{total})F} \tag{13}$$

where #type is the number of transitions of the specified type and #total is the total number of transitions in a video. For example, #disolve and #total are 31 and 166, respectively, for the video News 1 (19980328_ABC) in Table I. Equation (13) is then used as an approximate measure of the type-specific precision of our approach. The recall and precision values for different kinds of transitions using our approach are detailed in Tables VI and VII, respectively. The average recall values for the hard cut, dissolve and fade transitions are all higher than 94%, while the average precision values are all above 92%. The results demonstrate that our algorithm's performance is effective and independent of these types of transitions. Only the recall value for detecting wipe transitions is less than 90%. This is because a wipe transition sometimes combines two images separated by moving boundaries, where one of the images is from the previous shot, and the other is new. Such images will keep warping during a wipe transition, but they may not be altered very much between adjacent frames. The change in the number of matched keypoints is thus less obvious than that in the other three types of transitions. A possible way to further improve the wipe detection rate is to combine and utilize both keypoint and salient boundary information, because wipes usually contain moving boundaries. This will be a future work of our study.

C. Comparisons With Specifical Transition Detection Methods

In addition to the results described above, we compared our method with other noteworthy approaches. We selected three state-of-the-art algorithms, each of which was designed for a particular kind of transition, and compared them with the proposed general approach. The first algorithm was designed for hard-cut detection [4]. In their research, the authors compared

TABLE VIII
COMPARISON OF HARD CUT DETECTION ALGORITHMS

Name	# of hard cuts	Our algorithm		Gargi et al.'s method [4]	
		$Recall_{hard\text{-}cut}$	$Precision*_{hard\text{-}cut}$	$Recall_{hard\text{-}cut}$	$Precision_{hard\text{-}cut}$
Dissolve	140	98.57%	98.59%	83.57%	68.42%
News 1	116	99.14%	95.36%	86.21%	73.53%
News 2	17	100.00%	100.00%	82.35%	58.33%
Documentary 1	37	94.59%	100.00%	89.19%	66.00%
Documentary 2	83	97.62%	79.73%	62.65%	36.11%
TV Serial	297	95.62%	92.53%	75.00%	58.47%
Movie	14	100.00%	95.78%	64.29%	2.56%
Average		97.16%	92.95%	78.96%	39.64%

TABLE IX
COMPARISON OF DISSOLVE DETECTION ALGORITHMS

Name	# of dissolves	Our algorithm		Su et al.'s method [23]	
		$Recall_{dissolve}$	$Precision*_{dissolve}$	$Recall_{dissolve}$	$Precision_{dissolve}$
Dissolve	276	93.48%	98.52%	60.28%	88.72%
News 1	31	93.55%	95.10%	80.00%	91.43%
News 2	13	100.00%	100.00%	50.00%	100.0%
Documentary 1	29	75.86%	100.00%	55.17%	66.67%
Documentary 2	122	91.74%	78.36%	77.46%	66.34%
Movie	372	97.31%	95.67%	4.84%	11.54%
Average		94.42%	93.82%	41.53%	61.39%

many color histogram-based methods, and concluded that the one using the technique based on histogram intersection [5] achieved the best results. Hence, we compared the histogram intersection method in [4] with our approach. Given a pair of histograms of the current frame and the incoming frame, $\mathrm{His}(F_i)$ and $\mathrm{His}(F_{i+1})$, each containing n bins, the histogram intersection of them is the summation of the minimum values of each pair of the corresponding bins. Then, a hard-cut is detected if the intersection value is small.

The second and the third algorithms were designed for dissolve [23] and wipe [19], [20] detection, respectively. In [23], Su *et al.* proposed a motion tolerant dissolve detection algorithm, which detected dissolves by considering the monotonicity of changes in the intensity of pixels within an observation window. They classified the pixels in the window into three different categories: proponents, fence sitters and opponents. The probability of a pixel belonging to each category was computed and a binomial distribution model was used to distinguish between dissolves and motions. The method proposed by Fernando *et al.* [19], [20] detected a wipe transition by employing its geometric property. They first analyzed and modeled common wipe patterns, and found that wipes can be detected by salient moving boundaries. Hence, in their approach, salient lines between adjacent frames were detected by Hough transformation. If the slopes of the intersection lines remain equal for a time period, a wipe transition is identified. Line and complex pattern analysis of wipe transitions were also introduced in [20].

For all three methods, we experimentally choose the parameters that yield the best Q values for comparison. As each method can only detect one kind of transition, we compute the recall for the particular transition by using (11). The precision for the type of transition is computed by using (12), but the hits of the other types of transitions are not counted in the evaluation of F_{type}. Since the type-specific precision defined in (13) is only an approximation for overall reference, we suggest that readers focus on the recall values for comparison.

The performance comparison of our approach and the histogram intersection approach [4] for hard-cut detection is presented in Table VIII. Generally, our approach performs better because the histogram-based approach finds it relatively difficult to distinguish between moving objects and hard cuts. In contrast, our approach based on matching salient keypoint descriptors is more suitable to distinguish between them.

The performance comparisons of the dissolve and wipe cases are shown in Tables IX and X, respectively. Our method also outperforms Su *et al.*'s method [23] in most situations for the dissolve cases. Since their method is pixel-based, it is sensitive to image noise or the ghost effects in a video. However, we employ descriptors computed in salient local regions, which are more robust against noise.

Among the videos evaluated, the performance of the methods in [4] and [23] is not satisfactory for the video Movie (House of Flying Daggers), as shown in Tables VIII and IX, respectively. This is because that the video contains complex dance and fight scenes that involve very rapid movements. In contrast, our method can still provide satisfactory results in this situation.

Surprisingly, we found that the performance of Fernando *et al.*'s method [19], [20] is not as good as expected for wipe detection in our test videos, as shown in Table X. This is because many frames in the sequences contain multiple split-screens or twin pictures. The edges of the split-screens generate lines with similar slopes in adjacent frames, and thus result in serious false alarms when the method is used. Since our approach considerably avoids such false alarms by reliably matching and counting corresponding points, it is thus more accurate for wipe detection. The above comparisons demonstrate that our approach based on local keypoint matching is generally effective for shot change detection.

TABLE X
COMPARISON OF WIPE DETECTION ALGORITHMS

Name	# of wipes	Our algorithm		Fernando et al.'s method [19][20]	
		$Recall_{wipe}$	$Precision*_{wipe}$	$Recall_{wipe}$	$Precision_{wipe}$
News 1	10	80.00%	94.32%	60.00%	5.17%
News 2	7	85.71%	100.00%	71.43%	38.46%
Average		82.35%	96.67%	64.71%	8.53%

TABLE XI
COMPARISON WITH THE LOCAL KEYPOINT BASED METHOD IN HARD-CUTS

Name	# of hard cuts	Our algorithm		Park et al.'s method [13]	
		$Recall_{hard\text{-}cut}$	$Precision*_{hard\text{-}cut}$	$Recall_{hard\text{-}cut}$	$Precision_{hard\text{-}cut}$
Dissolve [Lienhart 2001]	140	98.57%	98.59%	85.71%	68.97%
News 1 (19980328_ABC)	116	99.14%	95.36%	93.10%	95.58%
News 2 (19980326_CNN)	17	100.00%	100.00%	94.12%	55.17%
Documentary 1 (ANNI005)	37	94.59%	100.00%	83.78%	93.94%
Documentary 2 (Return of the Caribou)	83	97.62%	79.73%	49.40%	66.13%
TV Serial (Lost)	297	95.62%	92.53%	89.56%	83.65%
Movie (House of Flying Daggers)	14	100.00%	95.78%	100.00%	9.33%
Average		97.16%	92.95%	84.66%	67.80%

TABLE XII
COMPARISON WITH THE LOCAL KEYPOINT BASED METHOD IN GRADUAL TRANSITIONS

Name	# of gradual transitions	Our algorithm		Park et al.'s method [13]	
		$Recall_{gradual}$	$Precision*_{gradual}$	$Recall_{gradual}$	$Precision_{gradual}$
Dissolve [Lienhart 2001]	287	93.38%	98.52%	72.13%	28.40%
News 1 (19980328_ABC)	50	92.00%	95.02%	74.00%	6.10%
News 2 (19980326_CNN)	21	95.24%	100.00%	42.86%	24.32%
Documentary 1 (ANNI005)	29	75.86%	100.00%	75.86%	11.17%
Documentary 2 (Return of the Caribou)	124	91.87%	78.39%	42.74%	6.56%
TV Serial (Lost)	1	100.00%	92.83%	100.00%	0.15%
Movie (House of Flying Daggers)	372	97.31%	95.67%	50.00%	15.56%
Average		94.22%	93.90%	58.26%	12.09%

D. Comparison With the Local Keypoint Based Method

We also compared our method with the local keypoint based method proposed by Park *et al.* [13]. In their method, a "frame distance" l is selected in advance. Each frame F_i is matched with the frame F_{i+l}, and if the number of matched points is larger than a fixed threshold Th_2, a shot boundary is declared. They first applied $l = 1$ to detect abrupt changes, and then applied a larger l to detect gradual transitions. The SIFT [14] was used to find image correspondence between frames. In the comparison, we exhaustively investigated the combination of l and Th_2 that yields the best Q values in the experiments, and the values of l turned out to be $l = 1$ and $l = 22$ for the hard-cut and the gradual transition, respectively.

The performance comparison of our approach and Park *et al.*'s approach [13] for hard-cut and gradual transition detection are presented in Tables XI and XII, respectively. The number of gradual transitions is the summation of the numbers of dissolves, fades and wipes. Our approach consistently performed better as shown in the tables, and the method in [13] broke down for several videos. We owe the reasons to the following. Applying a fixed interval between frames (eg., setting $l = 1$ and $l = 22$) cannot handle temporal variability of the video lengths of transitions in many situations. In our approach, the transition interval is first estimated by an outer extension of the interval delimited by the neighboring local maxima, and then we match the nonadjacent frames of the interval boundaries to refine the results. In addition, unlike the method in [13] that fixed the threshold of number of matched points for boundary detection, we proposed a method that uses local minima and maxima of the match numbers for threshold determination in our initial transition-candidate detection step. Our method can thus handle transition situations varying with the video contents better. In terms of the computation speed, our method is about ten times faster than Park *et al.*'s [13] approach. It is because that the CCH descriptor is employed in our approach, which is computationally more efficient than SIFT.

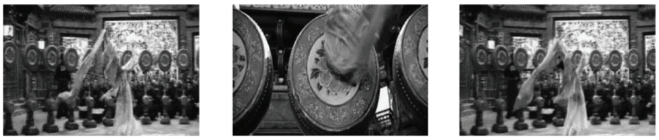


Fig. 10. Example of misdetection caused by scene changes twice in a very short period (ten frames).

E. Discussion

Although our method is effective for most cases, there are still some instances of mis-detection and false alarms. We now discuss some interesting cases encountered in our experiments.

Fig. 10 is a case of misdetection (a shot change that was not identified). The scene in the middle (the drums) only lasts for ten

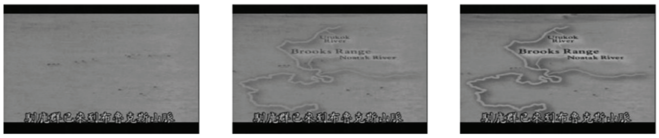


Fig. 11. False alarm caused by the map appearing while the background remains the same.

frames, but it involves a lot of motion. Consequently, the interval found by our algorithm spans the scenes (with the dancer) before and after the drums. Due to the high matching performance of the CCH descriptor, the two frames belonging to the same scene, but separated by another scene, are almost perfectly matched; therefore, the transition interval is discarded.

Fig. 11 shows an interesting example of a false alarm. The foreground is a map that fades in gradually, but the background does not change. We recognize the map as a new object that dominates the scene, so these frames indicate a false alarm. However, to some humans, this situation appears to be a shot change in the video.

V. Conclusion and Future Work

We have proposed a new method for shot change detection that is less sensitive to object or camera motion due to the robustness of the feature tracking algorithm. A method for finding intervals of transitions is also proposed. The contribution of our work is twofold. First, we solve the detection problem by using object recognition techniques, rather than some overall features, so that shot changes can be distinguished from object or background motions in a scene. Second, we propose a unified approach for detecting most kinds of shot changes; hence, there is no need to use different algorithms for different kinds of transitions. Our method is easy to implement and can achieve real-time processing. Since it can detect most transitions, the proposed approach can serve as a reliable initial detection technique for shot changes. It can also be combined with other methods to further distinguish between different types of shot changes.

References

[1] R. Zabih, J. Miller, and K. Mai, "A feature-based algorithm for detecting and classifying scene breaks," in *Proc. ACM Multimedia*, 1995, pp. 189–200.

[2] B. Shahraray, "Scene change detection and content-based sampling of video sequences," *SPIE Digital Video Compression, Algorithm and Technologies*, vol. 2419, pp. 2–13, 1995.

[3] 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, no. 7, pp. 1030–1044, 1999.

[4] U. Gargi, R. Kasturi, and S. H. Strayer, "Performance characterization of video-shot-change detection methods," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 1, pp. 1–13, Jan. 2000.

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

[6] K. Shen and E. J. Delp, "A fast algorithm for video parsing using mpeg compressed sequences," in *Proc. Int. Conf. on Image Processing*, 1995, vol. 2, pp. 252–255.

[7] R. Lienhart, "Reliable transition detection in videos: A survey and practitioner's guide," *Int. J. Image Graph.*, vol. 1, no. 3, pp. 469–486, 2001.

[8] C.-W. Ngo, T.-C. Pong, and H.-J. Zhang, "Motion analysis and segmentation through spatio-temporal slices processing," *IEEE Trans. Image Process.*, vol. 12, no. 3, pp. 341–355, Mar. 2003.

[9] J. Bescós, "Real-time shot change detection over online mpeg-2 video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 4, pp. 475–484, Apr. 2004.

[10] G. Boccignone, A. Chianese, V. Moscato, and A. Picariello, "Foveated shot detection for video segmentation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 3, pp. 365–377, Mar. 2005.

[11] B.-L. Yeo and B. Liu, "Rapid scene analysis on compressed video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, no. 6, pp. 533–544, 1995.

[12] Z. Cernekova, I. Pitas, and C. Nikou, "Information theory-based shot cut/fade detection and video summarization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 1, pp. 82–91, Jan. 2006.

[13] M.-H. Park, R.-H. Park, and S. W. Lee, "Shot boundary detection using scale invariant feature matching," in *Proc. SPIE Visual Communications and Image Processing*, 2006, vol. 6077, pp. 569–577.

[14] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[15] M. Wu, W. Wolf, and B. Liu, "An algorithm for wipe detection," in *Proc. Int. Conf. Image Processing*, 1998, vol. 1, pp. 893–897.

[16] S.-C. Pei and Y.-Z. Chou, "Efficient mpeg compressed video analysis using macroblock typeinformation," *IEEE Trans. Multimedia*, vol. 1, no. 4, pp. 321–333, Dec. 1999.

[17] S.-C. Pei and Y.-Z. Chou, "Effective wipe detection in mpeg compressed video using macro block type information," *IEEE Trans. Multimedia*, vol. 4, no. 3, pp. 309–319, 2002.

[18] W. A. C. Fernando, C. N. Canagarajah, and D. R. Bull, "Fade and dissolve detection in uncompressed and compressed video sequences," in *Proc. Int. Conf. on Image Processing*, 1999, vol. 3, pp. 299–303.

[19] W. A. C. Fernando, C. N. Canagarajah, and D. R. Bull, "Wipe scene change detection in video sequences," in *Proc. Int. Conf. on Image Processing*, 1999, vol. 3, pp. 294–298.

[20] W. A. C. Fernando and C. N. Canagarajah, "Wipe scene change detection and classification in video sequences," *J. Electron. Imag.*, vol. 13, no. 2, pp. 362–375, 2004.

[21] J. Nam and A. H. Tewfik, "Detection of gradual transitions in video sequences using b-spline interpolation," *IEEE Trans. Multimedia*, vol. 7, no. 4, pp. 667–679, 2005.

[22] R. Lienhart, "Reliable dissolve detection," *Storage and Retrieval for Media Databases, SPIE 4315*, pp. 219–230, 2001.

[23] 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.

[24] Z. Zhang, R. Deriche, O. Faugeras, and Q. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," *Artif. Intell.*, vol. 78, pp. 87–119, 1995.

[25] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. Fourth Alvey Vision Conf.*, 1988, pp. 147–151.

[26] C. Schmid and R. Mohr, "Local grayvalue invariants for image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 5, pp. 530–534, 1997.

[27] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, 2005.

[28] Y. Ke and R. Sukthankar, "Pca-sift: A more distinctive representation for local image descriptors," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2004, vol. 2, pp. 506–513.

[29] C.-R. Huang, C.-S. Chen, and P.-C. Chung, "Contrast context histogram—A discriminating local descriptor for image matching," in *Proc. Int. Conf. on Pattern Recognition*, Hong Kong, 2006, vol. 4, pp. 53–56.

[30] C.-R. Huang, C.-S. Chen, and P.-C. Chung, "Contrast context histogram—An efficient discriminating local descriptor for object recognition and image matching," *Pattern Recognit.*, vol. 41, no. 10, pp. 3071–3077, 2008 [Online]. Available: http://imp.iis.sinica.edu.tw/CCH/CCH.htm

[31] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2000, vol. 2, pp. 142–149.

[32] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 4, pp. 509–522, 2002.

[33] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," in *Proc. Int. Conf. on Computer Vision*, 2001, vol. 1, pp. 525–531.

[34] R. A. Joyce and B. Liu, "Temporal segmentation of video using frame and histogram space," *IEEE Trans. Multimedia*, vol. 8, no. 1, pp. 130–140, Feb. 2006.

Chun-Rong Huang (S'04–M'05) received the B.S. and Ph.D. degrees in the electrical engineering from National Cheng Kung University, Taiwan, R.O.C., in 1999 and 2005, respectively.

In 2005, he joined the Institute of Information Science, Academia Sinica, Taipei, Taiwan, where, since 2005, he has been a Postdoctoral Fellow. His research interests include computer vision, computer graphic, multimedia signal processing, image processing, and medical image processing.

Dr. Huang is a member of the IEEE Circuits and Systems Society and the Phi Tau Phi honor society.

Huai-Ping Lee received the B.S. degree in computer science from National Taiwan University, Taipei, Taiwan, R.O.C., in 2004. He is currently pursuing the Ph.D. degree at the University of North Carolina, Chapel Hill.

He was a Research Assistant with the Institute of Information Science, Academia Sinica, Taipei, in 2006. His research interests include image analysis and computer graphics.

Chu-Song Chen (S'94–M'97) received the B.S. degree in control engineering from National Chiao-Tung University, Hsinchu, Taiwan, R.O.C., in 1989. He received an M.S. degree in 1991 and the Ph.D. degree in 1996, respectively, both from the Department of Computer Science and Information Engineering, National Taiwan University (NTU), Taipei.

He is now an Associate Research Fellow with the Institute of Information Science, Academia Sinica, Taipei, and also an Adjunct Associate Professor with the Graduate Institute of Networking and Multimedia, NTU. His research interest includes pattern recognition, computer vision, signal/image processing, and multimedia. He has published more than 70 technical papers.

Dr. Chen serves as the Secretary-General of the Image Processing and Pattern Recognition (IPPR) Society, Taiwan, which is one of the societies of the International Association of Pattern Recognition (IAPR), since 2007. He received the outstanding paper awards of IPPR in 1997, 2001, 2005, and 2008.