MVA2007 IAPR Conference on Machine Vision Applications, May 16-18, 2007, Tokyo, JAPAN

A Local Keypoint Matching Technique for Transition Detection

Chun-Rong Huang
Institute of Information Science
Academia Sinica
Nankang, Taipei, Taiwan

nckuos@iis.sinica.edu.tw

Huai-Ping Lee
Department of Computer Science
University of North Carolina
at Chapel Hill
Campus Box 3175, Sitterson Hall

lhp@cs.unc.edu

Chu-Song Chen
Institute of Information Science
Academia Sinica
Nankang, Taipei, Taiwan

song@iis.sinica.edu.tw

Abstract

Shot change detection is an essential step in video content analysis. However, automatic shot change detection often suffers high false detection rates when there are camera or object movements. In this paper, we propose an approach to solve this problem based on local keypoint matching of the video frames. Experimental results show that the proposed algorithm is effective for all kinds of shot changes.

1 Introduction

The progress of storage and multimedia technologies have made videos to be easily retrieved and processed. Temporal segmentation is a fundamental step in video processing, and shot change detection is the most basic way to achieve it. Many studies on shot change detection such as [1] focus on finding low-level visual features, such as color histogram and edges, and then locate the spots of changes of these features. Motion estimation techniques such as optical flow are also used to find transitions, since shot changes imply motion changes. Bouthemy et al. [2] measured the number of pixels that belong to the part undergoing dominant motion to predict shot changes. Lienhart [3] used pattern recognition techniques to train a model to predict dissolve transitions. Some surveys for early approaches can be found in [1] and [4]. These kinds of 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. [5] proposed a video segmentation method from a different view. They used spatio-temporal slices to recognize camera motion, zooming, hard cuts, and so on. Boccignone et al. [6] proposed an interesting method based on the observation of human eyes when making comparisons between two pictures. Human eyes focus on a certain FOA (Focus of Attention) in a particular order to compare pictures. They tried to find out these FOA sequences by calculating *saliency maps* for each frame, and then detect the change of the FOA sequences. Cernekova et al. [7] used mutual information (MI) to measure information transported from one frame to another. Abrupt transitions and fades between two shots lead to a low MI. To distinguish fades from the abrupt transitions, they further exploited joint entropy as the inter-frame information. This approach shows an impressive performance on shot change detection, but can only be used for the cases of abrupt transitions and fades.

We propose a new unified approach to detect all kinds of shot changes. Our method is not based on change of some low-level features but based on the recognition of objects in the scene. If we can match many objects between two adjacent frames, there should not be transitions between them. With object tracking techniques, we can minimize the influence of object and camera motion, and therefore we can detect not only abrupt transitions but also gradual transitions. In the following sections, we present the feature matching method adopted, as well as our algorithm to find the shot changes.

2 Feature Matching for Image Correspondence Finding

The goal of feature matching is to match points on the same object from multiple images. In order to reduce ambiguities of matching, points to be matched must have some distinctive features that no other points would have; these features must be invariant to transformations such as translation, rotation, and scaling, so that we can still detect the object after it moves. Recently, Lowe [8] introduced a scale invariant feature transformation (SIFT) descriptor that is invariant to both scale and rotation. In [9], it has been shown that SIFT is one of the most effective approaches when scale and viewpoint changes occur. Instead of using edge orientations, contrast context histogram (CCH) [10], which is more efficient to compute, is proposed to find image correspondence. We are going to use CCH to detect the shot changes because CCH has comparable matching accuracy to that of SIFT, but spends much less computation time.

2.1 Contrast context histogram

The main issue in developing invariant local descriptors is how to represent a region effectively and discriminatively. The color histogram is an 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 there are already many salient keypoints (salient corners) extracted from an image I. For each keypoint $\boldsymbol{p}_c$ located at the center of an $n \times n$ local region $\boldsymbol{R}$, we compute the contrast $C(\boldsymbol{p})$ of a point $\boldsymbol{p}$ 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. We separate $\boldsymbol{R}$ into several non-overlapping regions, $\boldsymbol{R}_1$, $\boldsymbol{R}_2$, ..., $\boldsymbol{R}_t$. Without lost of generality, we use a log-polar coordinate system (r, θ), which is more sensitive to the positions of points close to the center than to those of points farther away, to perform the division. The direction of $\theta = 0$ in the log-polar coordinate system is set to coincide with the edge orientation of $\boldsymbol{p}_c$ to ensure that the descriptor is invariant to image rotations.

To increase the discriminative ability of the descriptor, we introduce the positive and negative histogram bins of the contrast values for each sub-region. For the region $\boldsymbol{R}_i$, we define the positive contrast histogram bin with 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 composing 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:

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

2.2 Locate transitions by matching adjacent frames

The first part of our algorithm is to locate the time instants at which shot changes take place. We observed that objects or scenes are being replaced during transitions, while they may be moving or rotating within a shot. Most previous works in shot change detection produce many false alarms when objects or cameras move because they only detect the change of some overall features in the video, which not only changes dramatically during transitions but also changes when something moves in a single shot. The advantage of feature matching is that it is invariant to affine transformations, and so we can match objects even after they moved. An additional advantage is that we do not have to design a detector for each kind of transitions. No matter how the shot changes, it is a change of objects in the scene, therefore all kinds of shot change can be detected in a unified manner.

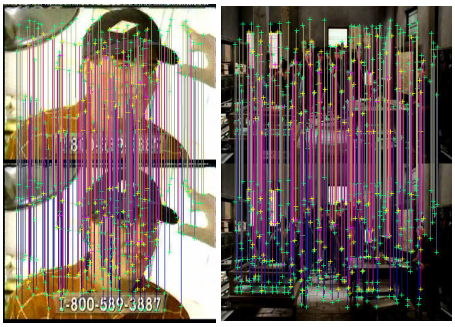


Figure 1: Some feature matching results between adjacent frames.

In our algorithm, each frame is preprocessed by keypoint detectors. The keypoints are extracted by detecting Harris corners [11] on each level of a multi-scale Laplacian pyramid [12]. A salient keypoint is selected by detecting the local maxima in a 7×7 region. Then, a local region $\boldsymbol{R}$ surrounding the keypoint is divided into several sub-regions 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:

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

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

We produce lists of keypoints and their local descriptor values for each frame. Then, we perform keypoint matching for each pair of adjacent frames, resulting in a 1D signal of numbers of the matched points. Formally, for the i-th frame F_i, there is a list of keypoints, key_i, consisting 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 $key_{i,j}$ in F_i is matched to the keypoint $key_{i+1,k}$ in F_{i+1} that has the shortest distance to $key_{i,j}$, but if the shortest distance is larger than a predefined threshold, the keypoint $key_{i,j}$ is not matched to any keypoint. Figure 1 shows some examples of feature matching results of adjacent frames.

Now each frame F_i is related to a value $y(i)$, the number of matched points between F_i and F_{i+1}. If a shot change occurs at F_i, the keypoints in F_i should have only few matched

keypoints in F_{i+1}, since most of the object appearing before the shot change should be replaced after the transition. Therefore, we assume that shot change takes place when there is a salient local minimum in the values $y(i)$, and the problem is reduced to finding the minima of $y(i)$.

Here is the formal formulation of our initial algorithm for finding the minima:

1. For each local minimum $y(t)$ satisfying that $y(t) < y(t-1)$ and $y(t) < y(t+1)$:
   1.1 Find the local maxima $y(l)$ and $y(r)$ on its left and right.
   Let M be the maximum in $\{y(i) \mid l \le i \le r\}$.
   1.2 If $M - y(t) > T_r \times M$, then $y(t)$ is a candidate, where T_r is a threshold selected within [0,1].
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 $y(t)$, and $y(r)$ is the maxima found just to the right of $y(t)$. We only keep the minimum that are less than a fraction ($T_r = 0.49$) of the maximum M in $\{y(i) \mid l \le i \le r\}$. In the second step, we eliminate the candidates with values larger than a threshold $T_{hi} = 185$ because there are still many matched keypoints, and thus the two adjacent frames are still considered to be 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.

2.3 Intervals of transitions

The candidates found with minimum finding are only time instants, not intervals. However, many shot changes are gradual transitions, and so finding the intervals of transitions is required. Shot changes are likely to occur when the number of matched objects decrease, and there should not be transitions when many objects are matched between adjacent frames. Therefore, the two local maxima to the left and right of the candidate are possible starting and ending points of the transition. We add an additional condition: the video sequence before and after the shot change should also be "stable," resulting in stable numbers of matched keypoints. So the search for starting and ending point begins at the two maxima, until the number of matched keypoints are stable.

For a given candidate $y(i)$, we first locate the nearest maxima $y(l)$ and $y(r)$ on its left and right. To find the starting 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. That is to say, 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, generating 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 starting frame of the transition. The ending frame of the transition is found in a similar manner by starting with matching between F_r and F_{r+1}.

3 Reducing False Alarms by Matching Non-adjacent Frames

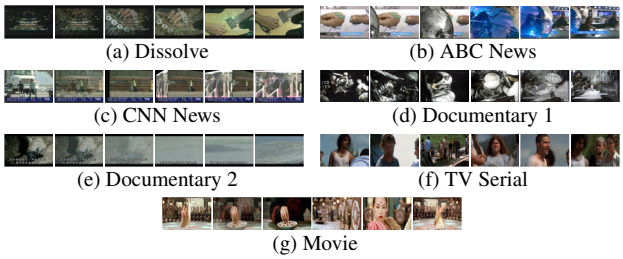
The above approach based on local-minimum analysis provides an efficient initial step to detect transition candidates. However, since only correspondences between adjacent frames are employed, false detection may occur when the video suffers from some changes such as sudden lighting changes, occlusions, and fast object motions. Matching non-adjacent 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 among a shot usually keeps up in a limited period of time and then the original shot will keep on going, we consider to match frames before and after the intervals of candidate shot changes. If there are still many matched keypoints, the two frames are considered similar to each other, and the transition candidate is likely to be a false detection. In detail, let $\pi = [t_{start}, ..., t_{end}]$ be a time interval of transition. F_{start} and F_{end} are the starting and ending frames of this transition, 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,

$$\mu_\pi = \frac{\sum_{t_{start}}^{t_{end}-1} y(t)}{t_{end} - t_{start}}. \quad (6)$$

If $N_\pi > T_e \times \mu_\pi$, where T_e is a fractional threshold (which is 0.07 in our experiments), transition π is removed. As a last step, when two adjacent transitions are too close to each other (only one or two frames apart), they are merged into one, because it is unreasonable to change into a scene for only a frame or two and then change into another.

4 Experimental Results

We tested our algorithm on several real videos, and their lengths and numbers of transitions are summarized in Table 1. The video "Dissolve" is from [3], including a lot of dissolve transitions. The video consists of clips from a concert and many TV commercials. The two news clips are from ABC and CNN. We tested them because wipe transitions are often seen in news preview but are seldom seen in other genres of videos. The movie "House of Flying Daggers" includes many complex dancing and fighting scenes and is a good material for test of object recognition and the effect of motion blur. The TV serial "Lost" also has many scenes with fierce motion and blur. Other test videos are two documentaries from the Open Video Project and from Discovery Channel. They have



(a) Dissolve (b) ABC News (c) CNN News (d) Documentary 1 (e) Documentary 2 (f) TV Serial (g) Movie

Figure 2: Sample frames in our test set.

Table 1: Test Sequences

Name	Frames	Shots	Recall	Precision
Dissolve	25262	427	92.04%	98.50%
News	23642	166	96.39%	96.39%
News	10789	38	94.74%	97.30%
Documentary	11321	66	84.85%	98.25%
Documentary	41358	207	93.72%	83.26%
TV Serial	30706	298	96.64%	88.89%
Movie	31236	386	97.41%	94.24%
Average			94.65%	93.07%

more static scenes, but still has a lot of motion when tracking animals.

We use recall and precision to measure the performance. They are defined as follows:

$$Recall = \frac{H}{H+M}, \quad Precision = \frac{H}{H+F}$$

where H, M, and F are the number of hits, miss detects, and false alarms, respectively.

The average recall and precision in our experiments are 94.65% and 93.07%, respectively. Our detailed results for all video clips are reported in Table 1. From this table, it can be seen that our method is effective for all kinds of shot changes. As for the computational speed, our method takes 0.0436 seconds per frame on an Intel Pentium 4 3.4G CPU computer with 768M memory.

5 Conclusion

We have proposed a new method for shot change detection. It 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. There are mainly two contributions in our work. Firstly, we solve the problem with object recognition techniques rather than with some overall features; shot changes can be distinguished with object or background motions in the scene. Secondly, we proposed a unified approach to detect all kinds of shot changes; there is no need to use a different algorithm for some kinds of transitions. Our method is easy to implement and can achieve nearly real-time processing. Since our method can detect most transitions, it can serve as a reliable initial detection of shot changes and be combined with other methods for further distinguishing the types of shot changes.

Acknowledgment

This research was supported in part by NSC95-2422-H-001-024, NSC 95-2422-H-001-007 and NSC 95-2752-E-002-007-PAE from the National Science Council, Taiwan.

References

[1] U. Gargi, R. Kasturi, and S. H. Strayer, "Performance characterization of video-shot-change detection methods," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 1, pp. 1–13, 2000.

[2] P. Bouthemy, M. Gelgon, and F. Ganansia, "A unified approach to shot change detection and camera motion characterization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 7, pp. 1030–1044, 1999.

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

[4] ——, "Reliable transition detection in videos: A survey and practitioner's guide," *International Journal of Image and Graphics*, vol. 1, no. 3, pp. 469–486, 2001.

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

[6] G. Boccignone, A. Chianese, V. Moscato, and A. Picariello, "Foveated shot detection for video segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 3, pp. 365–377, 2005.

[7] Z. Cernekova, I. Pitas, and C. Nikou, "Information theory-based shot cut/fade detection and video summarization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 1, pp. 82–91, 2006.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[9] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptorsl," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.

[10] C.-R. Huang, C.-S. Chen, and P.-C. Chung, "Contrast context histogram - a discriminating local descriptor for image matching," *In Proceedings of International Conference on Pattern Recognition*, vol. 4, pp. 53–56, 2006.

[11] C. Harris and M. Stephens, "A combined corner and edge detector," *In Proceedings of The Fourth Alvey Vision Conference*, pp. 147–151, 1988.

[12] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," *In Proceedings of International Conference on Computer Vision*, pp. 525–531, 2001.