PERGAMON

Pattern Recognition 34 (2001) 711–719

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Automatic video parsing using shot boundary detection and camera operation analysis☆

Mee-Sook Lee, Yun-Mo Yang[1], Seong-Whan Lee*

Center for Artificial Vision Research, Korea University, Anam-dong, Seongbuk-ku, Seoul 136-701, South Korea

Received 26 June 1999; received in revised form 16 December 1999; accepted 16 December 1999

Abstract

In this paper, we present an efficient video parsing method for automating content-based video indexing and retrieval using shot boundary detection and camera operation analysis techniques. In the shot boundary detection, the local color information is used in order to eliminate the false detection caused by an abrupt change of illumination such as camera flash or thunder. In order to reduce the computation time in the shot boundary detection, an adaptive time window is applied to this procedure. Local spatio-temporal images and multilayer perceptron are used for analyzing camera operations. The proposed method uses a learning algorithm with spatio-temporal information in the frame and does not process the entire video image to reduce the processing time. In order to verify the performance of the proposed automatic video parsing method, experiments have been carried out with a video database that includes news, documentary and movie. Experimental results demonstrate the efficiency of the proposed video parsing technique.

Keywords: Automatic video parsing; Content-based retrieval; Video indexing; Shot boundary detection; Camera operation analysis; Multilayer perceptron

1. Introduction

Advances in multimedia technologies have demonstrated that video data is a very important and common medium for education, broadcasting, publishing, and military intelligence. However, the effective use of video data is seriously limited by the lack of technologies that make the organization and retrieval of information from video data sources easy and effective. Moreover, it is difficult to represent and manage video data due to its time-dependent nature [1]. When we watch video data, it usually takes about 2 h. We sometimes skip the video data to get to an overview or search for a specific part. Therefore, a method for fast video data browsing that enables a viewer to get an idea of the video content without watching it entirely, is one of the most desirable functions. Content-based video indexing and retrieval is one method that finds and manage the essential information of video data [2]. It is desirable to enable viewers to get a summary of the sequence of the video data without seeing its whole sequence.

An automatic video parsing is necessary for the content-based video indexing and retrieval. Video parsing involves two tasks: a video segmentation and a video indexing. The video streams are segmented into elemental units such as shots and scenes at the video segmentation stage (see Fig. 1). The elemental units are labeled based on their content information at the video indexing stage. Generally, the video data consists of three basic units: frame, shot, and scene. A frame is an individual image. A video shot can be defined as consecutive frames. Finally, a scene is comprised of a collection of one or

☆This research was supported by National Creative Research Initiatives Program of the Korean Ministry of Science and Technology.

[1] The author is also affiliated with the Department of Electronics and Information Engineering, Korea University.

* Corresponding author.

0031-3203/01/$20.00
PII: S0031-3203(00)00007-8

more adjoining shots that focus on an object or several objects of interest. Shots are the basic unit for video manipulation. There are many different transitions between shots. The simplest transition is cut, but the other transitions include fade, dissolve, wipe, and so on. Some of examples are shown in Fig. 2.

Camera operation analysis is useful for partitioning and indexing the video data. There are six basic camera operations (Fig. 3).

In this paper, an efficient video parsing method using the shot boundary detection and the camera operation analysis is presented. For the shot boundary detection, local color information is used. In order to reduce the computation time, an adaptive time window is utilized. Local spatio-temporal images and multilayer perceptron (MLP) are used for analyzing the camera operations. This method is reliable and fast, because it utilizes the learning algorithm with spatio-temporal information in the frame and does not have to process the entire image.

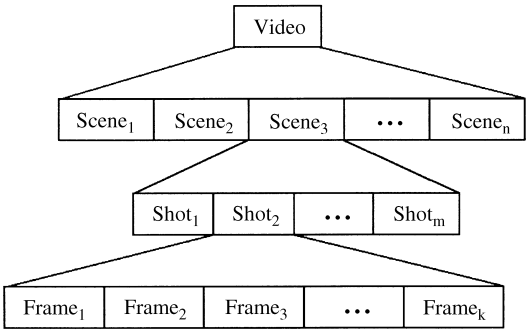

Fig. 1. The hierarchical structure of a video.

The rest of this paper is organized as follows: Section 2 describes the previous works; Section 3 proposes a shot boundary detection method; Section 4 describes a new camera operation analysis method; Section 5 describes the experimental results; and Section 6 concludes this paper.

2. Previous works

In this section, we review previous research on shot boundary detection and camera operation analysis. Early methods for shot boundary detection focused on cut detection, and more recent researches focus on gradual transition detection.

2.1. Shot boundary detection

The major techniques that have been used for the shot boundary detection are pixel differences, edge difference, and histogram comparison. In these methods, if the difference between two adjacent frames exceeds a predefined threshold, then the latter frame is detected as a shot boundary.

Zhang et al. [1] proposed a method using pixel differences. This is the easiest way to detect shot boundary. In this method, the number of pixels was counted, if the pixels' gray values are greater than a threshold. This count is compared against a second threshold to determine if the shot boundary has been found. In this method, a 3×3 filter was used in order to eliminate sensitivity to camera movement and noise effects.

In the methods using edge difference, the number and position of edges in edge-detected images were compared to reduce the effects of camera motion [4]. The percentage of edges that enter or exit between the two adjacent

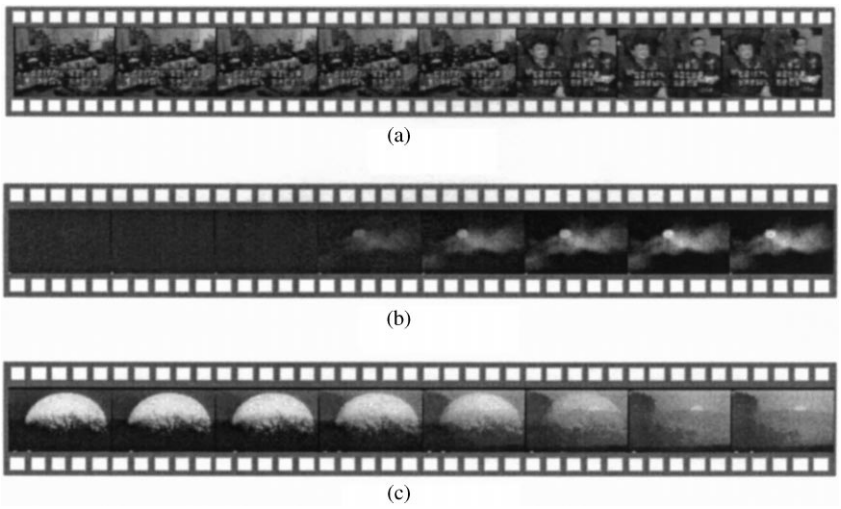

Fig. 2. Different types of transition between shots. (a) Cut, (b) fade, (c) dissolve.

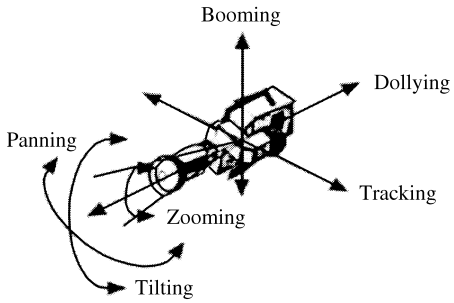

Fig. 3. The six basic camera operations [3].

frames was computed. Shot boundaries were detected if the percentage exceeded the predefined threshold. This method was more accurate at detecting cuts and much less sensitive than chromatic scaling, but it needed much more computing time than the other methods.

The histogram comparison is the most common method used to detect shot boundaries. The simplest method using the histogram computes gray levels or color histograms of the two adjacent frames. In these methods, if bin-wise difference between the two histograms of two adjacent frames exceeds a threshold, the shot boundary is detected. Zhang et al. [1] implemented the histogram comparison using gray-level histogram, demonstrating that it was very efficient for detecting shot boundaries. Swanberg et al. [5] used gray-level histogram differences in regions weighted by how likely the region was to change in the video sequence. However, the generalization ability of this method is very poor because it needs the special models of the video scenes.

The above methods for the detection of shot boundaries have significant problems caused by camera motion and object movement, noise, and gradual transitions. One way to resolve these problems is to analyze the various camera operations.

2.2. Camera operation analysis

The method of camera operation analysis was addressed first through motion vector field analysis [3], by matching motion vector fields with predefined models in Hough space for various camera operations. This method was computationally expensive because it used optical flow and block matching algorithm to analyze the direction of motions in the video data. Moreover, it was greatly affected by noise (such as the vibrations of a camera caused by unintentional movements of the cameramen's hands) and the frames in a sequence that were spoiled by flashlights.

Zhang et al. [6] proposed a camera operation analysis method using motion vectors in MPEG-type video data. In this method, the predetermined prototypes of motion vectors caused by the camera operations were used.

Tonomura and Akutsu [7] proposed a new camera operation analysis method using X-ray images obtained by computing the average of each line and each column in successive frames. The corresponding method operates on groups of x-t and y-t spatio-temporal images. An x-t (respectively y-t) spatio-temporal image is obtained by fixing the variable y (respectively x) constant in the moving image box. Up/down motions are represented in the y-t image and left/right motions are represented in the x-t image. An edge detection is first performed for all the spatio-temporal images; the intensity and the gradients of edges are calculated. Then a horizontal (respectively vertical) X-ray image is obtained by taking weighted integral for the edge images of x-t images (respectively y-t images), weighting on the intensity of edges.

Maeda [8] proposed a modified method, based on the method of Tonomura and Akutsu [7], that would to reduce the computing time. In this method, the spatio-temporal images made in sub-blocks are used to analyze camera operations.

However, some difficult problems remain in the camera operation analysis. All methods often fail when very large object motion cannot be discriminated from background motion; even in cases in which the semantics processing of a human viewer succeeds in distinguishing between the two objects. Also, some methods are too sensitive to noise.

3. Shot boundary detection

The method of histogram comparison is the most common method for detecting shot boundaries from a raw video data. This method is quite simple, but ignores spatial information in the frame. Also an error rate caused by an abrupt luminance change is very high. In order to solve this problem, color histogram comparison is proposed. However, this method needs an additional computation time to convert one color model into other. In this paper, the mean of color values in local sub-blocks is used. This feature contains the spatial information of a frame and is robust when the abrupt luminance change occurs.

3.1. Local color histogram comparison

In this paper, we propose a new method for efficient shot boundary detection, using the local color information modifying the net comparison method [9]. In this proposed method, the mean of color values in local sub-blocks is used to more fully utilize the spatial information in a frame, and to eliminate a false detection caused by abrupt luminance changes. The basic notions used in this paper for detecting the shot boundaries are given as follows.

Assume that each pixel in an image has the same probability to change. The similarity SIM used for

detecting the shot boundaries is defined as follows:

$$SIM(n,n+k) = 1 - \frac{N_C(n,n+k)}{N}, \tag{1}$$

where N is the total number of sub-blocks in one frame, N_C is the number of sub-blocks that are changed between two frames, and k is the size of window,

$$N_C(n,n+k) = \sum_{i=0}^{N-1} \varphi_{n,n+k}(i). \tag{2}$$

In Eq. (2), φ is the function defined as follows:

$$\varphi_{n,n+k}(i) = \begin{cases} 1 & \text{if } D^H_{n,n+k} > T_{sub} \text{ or } D^S_{n,n+k} > T_{sub}, \\ 0 & \text{otherwise}, \end{cases} \tag{3}$$

where D^H and D^S represent the difference of mean values of hue(H) and saturation(S), between two sub-blocks, respectively.

$$D^H_{n,n+k}(i) = |E_H(n,i) - E_N(n+k,i)|, \tag{4}$$

$$D^S_{n,n+k}(i) = |E_S(n,i) - E_S(n+k,i)|. \tag{5}$$

In Eqs. (4) and (5), $E_H(n,i) = \frac{1}{M}\sum_{k=0}^{M-1} H^n_i(k)$ and $E_S = \frac{1}{M}\sum_{k=0}^{M-1} S^n_i(k)$. $H^n_i(k)$ and $S^n_i(k)$ are the hue and saturation value of the kth pixel located at the ith sub-block in the nth frame, respectively.

The proposed algorithm for the shot boundary detection described above is summarized as in Algorithm 1.

Algorithm 1. Proposed algorithm for the shot boundary detection

1. Determine the number and the position of sub-blocks.
2. Convert RGB color into HSV color.
3. Compute the mean values of hue(H) and saturation(S) at each sub-block.
4. Compare the corresponding sub-blocks in the adjacent frames by computing the difference D^H and D^S.
5. If one of the D^H and D^S is greater than the predefined threshold T_{sub}, we determine that this sub-block has been changed: otherwise it has not been changed.
6. If the ratio of the changed sub-blocks to the total number of sub-blocks is greater than the threshold T_{frame}, a shot boundary is declared.

3.2. Adaptive time window

It is time consuming to apply a shot boundary detection method to every frame in a video sequence, because most all video shots comprise of dozens of frames, and the frames within a shot are very similar in color and content. In order to reduce the computation time, we use a time sliding window which represents the size of a shot. All the frames within the time sliding window are similar, so the process to compare the difference between them can be skipped. However, there may be a high error rate in a method using the fixed size time window when the sampling rate is higher than the size of the time window. So, we used an adaptive time window whose size can be changed. An algorithm for the adaptive time window is given in Algorithm 2.

Algorithm 2. An algorithm for the adaptive time window

```
WindowSize = MaxWindowSize;
for(i = 0; i<TotalNumberOfFrames; i++){
  NextFrameNo = CurrentFrameNo + WindowSize;
  if(IsSimilar(F(CurrentFrameNo), F(NextFrameNo))){
    CurrentFrameNo = NextFrameNo;}
  else {
    if(WindowSize == 1){
      CurrentFrameNo = NextFrameNo;
      WindowSize = MaxWindowSize;}
    else {
      WindowSize = WindowSize/ScaleFactor; }}}
```

In Algorithm 2, if the size of a shot is larger than the window, there is not a shot boundary within the window, and the time sliding window moves to the next. Otherwise, the size of the window is reduced by the scale factor to find the exact shot boundary.

4. Camera operation analysis

In this section, we propose a method for analyze the camera operations using 2D spatio-temporal images and a multilayer perceptron. The entire procedure of the proposed method is shown in Fig. 4.

First, the 2D spatio-temporal images are generated for a given period. Then, the 2D discrete fast Fourier transform and the power spectrums are applied in order to analyzing the texture of the spatio-temporal images. Finally, the multilayer perceptron is used to analyze the camera operations.

4.1. Spatio-temporal image

A spatio-temporal image is the representation of a path that a pixel moves for a given time. They have a special texture if pixels move in the same direction, so we can extract the direction of pixels by analyzing the texture of the spatio-temporal image. The spatio-temporal image is formed by stacking up the corresponding segments in a time sequence. Fig. 5 shows how spatio-temporal images are generated [8].

As shown in the Fig. 5, a segment is a horizontal or a vertical line in a frame. So, one spatio-temporal image is sufficient for only one direction, horizonal or vertical.

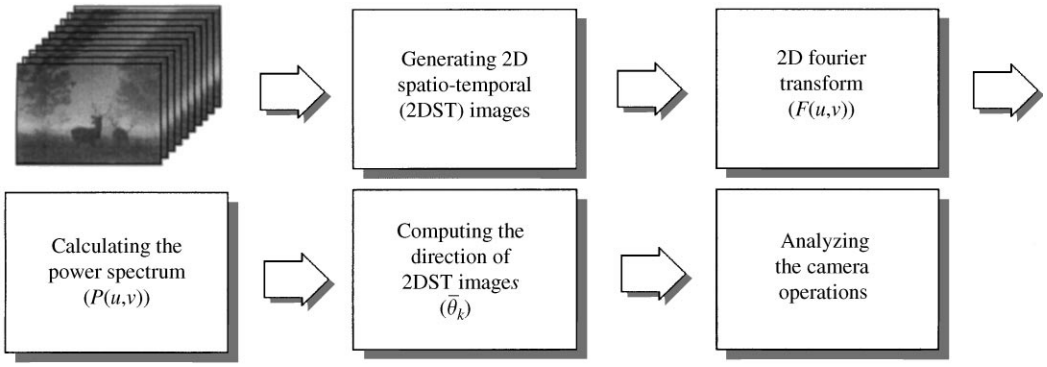

Fig. 4. The procedure for analyzing the camera operations.

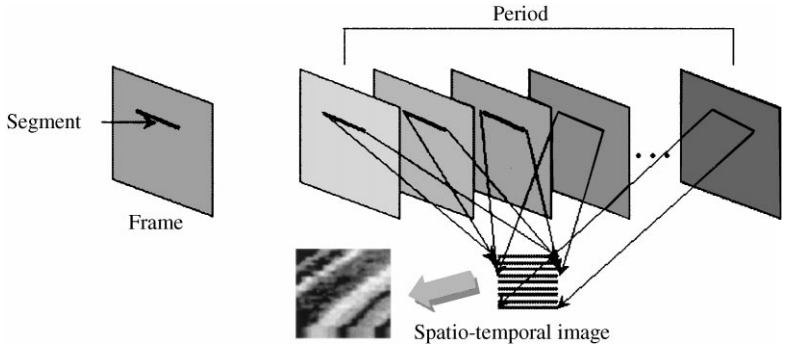

Fig. 5. A process to generate the spatio-temporal image.

The positions and directions of segments depend on the kind of camera operations which are analyzed. In this paper, two directions — a horizontal and a vertical axis — are used, and an appropriate number of segments are placed in a frame to get the exact camera movement.

The camera operations can be extracted by analyzing the texture of the spatio-temporal images because the spatio-temporal images have a unique pattern when special camera operations occur. The two-dimensional discrete Fourier transform (2DFFT) and the power spectrum are used to analyze the textures in the spatio-temporal images. The direction θ_k of the spatio-temporal image in each segment can be calculated from Eq. (6)

$$p(\theta_k) > p(\theta_i) \quad (i \neq k,\ i,k = 0, \ldots, \pi), \tag{6}$$

where $p(\theta)$ is defined as $p(\theta) = \sum_r P(r,\theta)$. $P(r,\theta)$ is the power spectrum in polar coordinates form.

4.2. Camera operation analysis using multilayer perceptron

In this paper, two MLPs are used. One is for horizontal direction, and the other is for vertical direction. The architecture of the two MLPs used to analyze the camera operations is the same. This MLP has three layers: one input layer, one hidden layer and one output layer. Fig. 6 shows the architecture of an MLP used for analyzing camera operations.

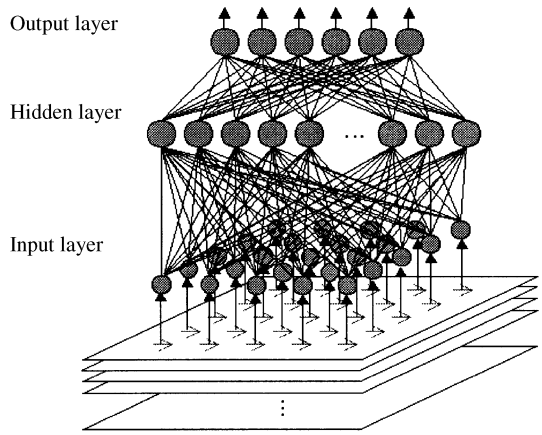

Fig. 6. An architecture of an MLP for analyzing the camera operations.

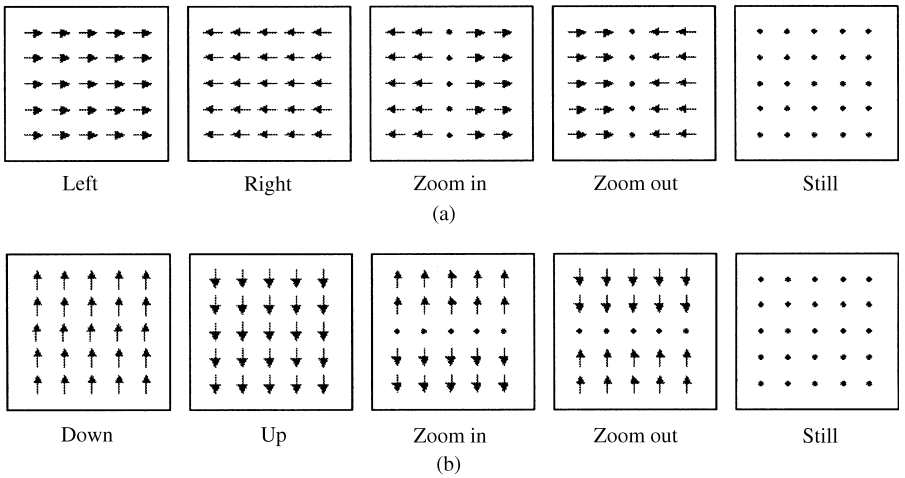

Fig. 7. The standard data for training the MLPs. (a) Standard training data for the horizontal camera movements. (b) Standard training data for the vertical camera movements.

The input layer has $k \times k$ nodes, and each node gets the direction extracted by analyzing each spatio-temporal image as input data. The number of nodes in an input layer depends on the size and the number of the spatio-temporal images in a frame. In this paper, a size of 32×32 and 25 segments (5×5) are selected through the preliminary experiments.

The hidden layer has 10 nodes to maximize the performance of the MLP. The output layer has 6 nodes corresponding each camera operation. For example, in case of the MLP for horizontal direction, each node in the output layer represents the still, right, left, zoom in, zoom out, or ambiguous case. The data created by adding the random noises into the standard training data are used to train the networks. The standard training data for each MLP are given in Fig. 7.

5. Experimental results

In order to verify the performance of the proposed video parsing method, experiments have been carried out with the various video data. The category, size and components of each video data are given in Table 1. All video data were digitized at a size of 320×240 pixels at 15 fps (frames per second).

5.1. Results of shot boundary detection

We compare the proposed method with previous methods. The selected previous methods are pixel-wise comparison (PWC), gray-level histogram comparison (GHC), local block gray-level histogram comparison (LGHC), histogram comparison of difference image (HCOD), and edge image comparison (EIC). The difference between frames of each of the two methods is defined as follows:

Table 1
Video data used for experiments

Video category	Video size (minutes)	The number of shot boundary	The number of camera operations
News	20	218	19
Documentary	10.35	92	31
Movie	10.48	71	17

- *Pixel-wise comparison (PWC):*

$$PWC(n, n+k) = \frac{\sum_{i=0}^{N-1} \Phi_{n,n+k}(i)}{N}, \tag{7}$$

where $\Phi_{n,n+k}(i)$ is the function that returns 1 if the difference between two pixels of two frames is greater than the predefined threshold, T_{pix}, otherwise returns 0. $\Phi_{n,n+k}(i)$ is defined as Eq. (8). In Eq. (8), $D^G_{n,n+k} = |I_n(i) - I_{n+k}(i)|$. $I_n(i)$ is the gray value of the ith pixel in the nth frame is

$$\Phi_{n,n+k}(i) = \begin{cases} 1 & \text{if } D^G_{n,n+k}(i) > T_{pix}, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

- *Gray-level histogram comparison (GHC):*

$$GHC(n, n+k) = \sum_{i=0}^{Q-1} |G_n(i) - G_{n+k}(i)|, \tag{9}$$

where $G_n(i)$ is the ith gray-level histogram value of the nth frame, and Q is the size of a histogram bin.

● *Local block gray-level histogram comparison (LGHC):*

$$LGHC(n, n+k) = \sum_{b=0}^{N_b-1} \sum_{i=0}^{Q-1} |G_n(b,i) - G_{n+k}(b,i)|, \quad (10)$$

where N_b is the number of local blocks in a frame, and $G_n(b,i)$ is the ith gray-level histogram value of the bth local block in the nth frame.

● *Histogram comparison of difference image (HCOD):*

$$HCOD(n, n+k) = \frac{\sum_{i \notin [-T_{pix}, T_{pix}]} hod(i)}{\sum_{i=-Q+1}^{Q-1} hod(i)} = \frac{\sum_{i \notin [-T_{pix}, T_{pix}]} hod(i)}{N}, \quad (11)$$

where $hod(i)$ is the ith histogram value of the difference image between two frames, and $hod(i)$ is defined as follows (in Eq. (12), f_n is the nth frame):

$$hod(i) = G(f_n - f_{n+k}, i). \quad (12)$$

● *Edge image comparison (EIC):*

$$EIC(n, n+k) = \max(\rho_{in}, \rho_{out}), \quad (13)$$

where ρ_{in} and ρ_{out} are the number of entering edge pixels and exiting edge pixels in frames, respectively. ρ_{in} and ρ_{out} are defined as follows:

$$\rho_{in} = 1 - \frac{\sum_{x,y} \bar{E}_n[x+\delta x, y+\delta y] E_{n+k}[x,y]}{\sum_{x,y} E_n[x+\delta x, y+\delta y]}, \quad (14)$$

$$\rho_{out} = 1 - \frac{\sum_{x,y} E_n[x+\delta x, y+\delta y] \bar{E}_{n+k}[x,y]}{\sum_{x,y} E_n[x,y]}, \quad (15)$$

where E_n is the edge image of the nth frame, and $\bar{E}_n$ is the dilated image of E_n. $(\delta x, \delta y)$ is a representative motion vector between two frames.

Table 2 compares the results of the shot boundary detection of the previous methods and the proposed method. All of methods including the proposed method were designed to examine every frame of the test video data (N_C, N_{FN} and N_{FP} represent the number of frames correctly detected, the number of false negatives and the number of false positives, respectively).

The edge difference comparison shows the best performance in detecting the shot boundaries, which the local block gray-level histogram comparison and the proposed method show the good performance as compared with other methods. However, the number of false positives detected by the local block histogram comparison was too large.

Table 3 shows the computing time of each method. According to Table 3, the pixelwise comparison method and the proposed method are faster than the others. The edge image comparison method shows the worst performance in computing time because it needs very complex procedures.

Table 2
Performance of the shot boundary detection methods

Method	News			Documentary			Movie		
	N_C	N_{FN}	N_{FP}	N_C	N_{FN}	N_{FP}	N_C	N_{FN}	N_{FP}
PWC	164	54	344	82	10	103	53	18	150
GHC	194	24	135	78	14	89	61	10	153
LGHC	202	16	303	87	5	104	63	8	128
HCOD	190	28	135	85	7	89	62	9	97
EIC	208	10	104	88	4	67	65	6	116
Proposed method	202	16	181	87	5	89	63	8	111

Table 3
Computing time (frames/s)

Method	Video categories		
	News	Documentary	Movie
PWC	11.2	10.7	11.5
GHC	6.0	6.6	6.4
LGHC	7.2	6.9	7.0
HCOD	8.4	8.6	8.4
EIC	3.5	3.0	3.5
Proposed method	10.8	10.2	10.5

The edge image comparison seems to be the best method when computing time is not important. The pixelwise comparison and the gray-level histogram comparison can be recommended for applications where the ratio of false positive is not important. Therefore, the proposed method seems to be the best algorithm in view of performance and computing time.

5.2. Results of camera operation analysis

We classify the basic camera operations in three classes: horizontal movement, vertical movement, and zoom in and out. The examples of the video data used in these experiments are shown in Fig. 8.

Table 4 shows the results of camera operation analysis of the proposed method. In Table 4, the composition type represents the data in which two or more simple camera operations are mixed together. The examples of this composition are panning with zooming in, tilting with zooming out, panning with tilting and zooming in, etc.

The documentary video data in Table 1 is used for these experiments. The proposed method detected correctly simple camera operations, but performed poorly in detecting composite camera operations. In the case of

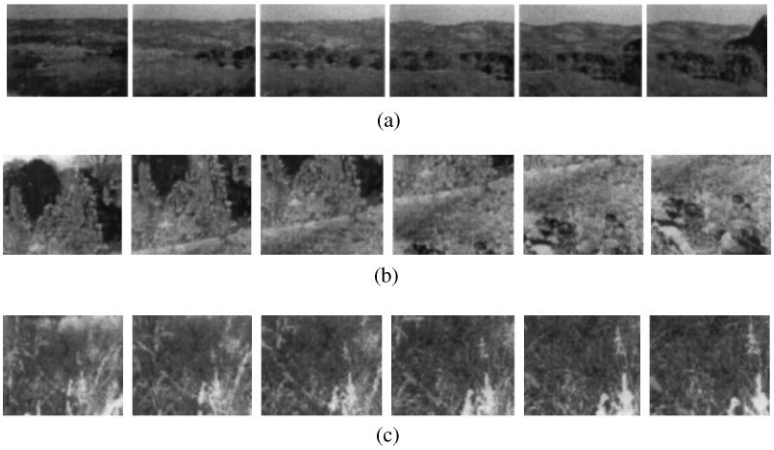

Fig. 8. The examples of the video data used in the camera operation analysis. (a) Horizontal movement, (b) Vertical movement, (c) Zoom in.

Table 4
Results of the camera operation analysis

The type of camera operations	The number of camera operations	N_C	N_{FN}	N_{FP}
Horizontal movement	25	19	6	75
Vertical movement	9	6	3	0
Zoom in/zoom out	10	7	3	21
Composition	23	11	12	57

horizontal movement detection, an error rate caused by moving objects is high, so the number of false positives is rather large.

6. Conclusions

In this paper, we have proposed an efficient video parsing method for detecting shot boundaries and for analyzing camera operations. In the proposed method, since local sub-blocks can remove the influence of some factors such as light changing different irradiance, local color information excluding value information was used for detecting shot boundaries to eliminate the false detection caused by an abrupt change of illumination such as camera flash or thunder.

In order to reduce the computation time in shot boundary detection, an adaptive time window has been applied to this procedure. In this method, since we can skip similar frames, the calculation time to detect a shot boundary can be dramatically reduced in the case of a long shot. This method is efficient for documents or movies, but is not good for dynamic videos such as sports or commercial advertisements.

Local spatio-temporal images are used for analyzing camera operations. We can increase the reliability of the proposed method using multilayer perceptron. The proposed method is fast because it does not have to process the entire video image.

Experimental results with various video data sources demonstrated the efficiency of the proposed video parsing method. For further study, the detection capabilities for shots caused by various gradual transitions should be developed. In addition, the position and direction of segments should be experimented with during the various camera operations.

Acknowledgements

Y.-M. Yang was supported by University Research Program supported by the Korean Ministry of Information Communication.

References

[1] H.J. Zhang, A. Kankanhalli, S.W. Smoliar, Automatic partitioning of full-motion video, Multimedia Systems 1 (1) (1993) 10–28.
[2] P. Aigrain, H.J. Zhang, D. Petkovic, Content-based representation and retrieval of visual media: a state-of-the-art review, Multimedia Tools Appl. 3 (1996) 179–202.
[3] A. Akutsu, Y. Tonomura, H. Hashimoto, Y. Ohba, Video indexing using motion vectors, Proceedings of SPIE — Visual Communications and Image Processing'92, Vol. 1818, San Jose, CA, 1992, pp. 1522–1530.
[4] R. Zabih, J. Miller, K. Mai, A feature-based algorithm for detecting and classifying scene breaks, Proceedings of ACM Multimedia'95, San Francisco, CA, 1995, pp. 189–200.

[5] D. Swanberg, C.F. Shu, R. Jain, Knowledge guided parsing in video database, Proceedings of SPIE — Storage and Retrieval for Image and Video Databases, Vol. 1908, San Jose, CA, 1993, pp. 13–24.
[6] H.J. Zhang, C.Y. Low, Y. Gong, S.W. Smoliar, Video parsing using compressed data, Proceedings of SPIE — Imaged Video Parsing II, Vol. 2182, San Jose, CA, 1994, pp. 142–149.
[7] Y. Tonomura, A. Akutsu, Y. Taniguchi, G. Suzuki, Structured video computing, IEEE Multimedia Mag. (1994) pp. 34–43.
[8] J. Maeda, Method for extracting camera operations to describe sub-scenes in video sequences, Proceedings of SPIE — Digital Video Compression on Personal Computers: Algorithms and Technologies, Vol. 2187, San Jose, CA, 1994, pp. 56–67.
[9] W. Xiong, J.C.M. Lee, M-C. Ip, Net comparison: a fast and effective method for classifying image sequences, Proceedings of SPIE — Storage and Retrieval for Image and Video Databases II, Vol. 2420, San Jose, CA, 1995, pp. 318–328.

About the Author—MEE-SOOK LEE was born in Seoul, Korea, in 1973. She received the B.S. and M.S. degree in computer science and engineering from Korea University, Seoul, Korea, in 1996 and 1998, respectively.

She is currently working as a Researcher at James Martin + Co, Seoul, Korea. Her research interests include image processing, content-based video retrieval, and pattern recognition.

About the Author—YUN-MO YANG received his B.S. degree in Electronics Engineering from Korea University in 1979, and a M.S. and Ph.D. degrees in Information Engineering from Tohoku University in Japan in 1984 and 1988, respectively. He has been working as a professor in the Department of Electronics and Information Engineering at the Korea University. His research interests include character recognition, image processing and digital video analysis.

About the Author—SEONG-WHAN LEE received his B.S. degree in Computer Science and Statistics from Seoul National University, Seoul, Korea, in 1984; and M.S. and Ph.D. degrees in Computer Science from KAIST in 1986 and 1989, respectively.

From February 1989 to February 1995, he was an Assistant Professor in the Department of Computer Science at Chungbuk National University, Cheongju, Korea. In March 1995, he joined the faculty of the Department of Computer Science and Engineering at Korea University, Seoul, Korea, as an Associate Professor. Currently, he is the director of National Creative Research Initiative Center for Artificial Vision Research (CAVR) supported by the Korean Ministry of Science and Technology.

He was the winner of the Annual Best Paper Award of the Korea Information Science Society in 1989. He obtained the First Outstanding Young Researcher Award at the 2nd International Conference on Document Analysis and Recognition in 1993, and the First Distinguished Research Professor Award from Chungbuk National University in 1994. He also obtained the Outstanding Research Award from the Korea Information Science Society in 1996.

He has been the Co-Editor-in-Chief of the International Journal on Document Analysis and Recognition since 1998 and the Associate Editor of the Pattern Recognition Journal, the International Journal of Pattern Recognition and Artificial Intelligence, and the International Journal of Computer Processing of Oriental Languages since 1997.

He was the Program Co-Chair of the 6th International Workshop on Frontiers in Handwriting Recognition, the 2nd International Conference on Multimodal Interface and the 17th International Conference on Computer Processing of Oriental Languages, the 5th International Conference on Document Analysis and Recognition, and the Workshop Co-Chair of the 3rd International Workshop on Document Analysis Systems.

At present, he is the Workshop Co-Chair of the IEEE International Workshop on Biologically Motivated Computer Vision in 2000 and the Program Co-Chair of the 8th International Workshop on Frontiers in Handwrighting Recognition in 2002. He served on the program committees of several well-known international conferences.

He is a fellow of International Association for Pattern Recognition, a senior member of the IEEE Computer Society and a life member of the Korea Information Science Society, the International Neural Network Society, and the Oriental Languages Computer Society.

His research interests include pattern recognition, computer vision and neural networks. He has more than 150 publications on these areas in international journals and conference proceedings, authored two books and edited four books.