Video Shot Cut Detection Using Adaptive Thresholding

Y. Yusoff, W. Christmas and J. Kittler
Centre for Vision, Speech and Signal Processing
University of Surrey
Guildford GU2 7XH
e-mail: {Y.Yusoff,W.Christmas}@eim.surrey.ac.uk

Abstract

The performance of shot detection methods in video sequences can be improved by the use of a threshold that adapts itself to the sequence statistics. In this paper we present some new techniques for adapting the threshold. We then compare the new techniques with an existing one, leading to an improved shot detection method.

Keywords: shot cut detection, video databases

1 Introduction

With the rapid rise of interest in analysis of audio-visual material, there is a corresponding growth in the need for methods to reliably detect shot boundaries within the video sequence. There are several approaches to the problem [1, 4, 12]. In many of the methods, the detection decision is based on a hard threshold of some dissimilarity measure, whose value is determined by experimentation. The optimal value depends on the requirements of the application and will be a trade-off between the number of false positives detected and the number of undetected true positives.

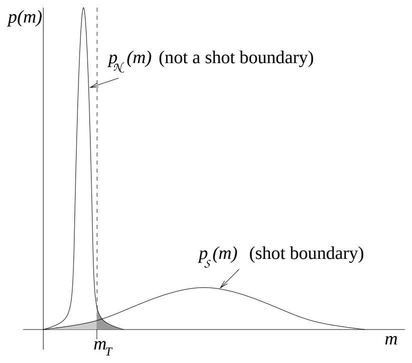
There are various possibilities for improving on the basic methods. The variety of basic methods opens up the possibility of combining several of them into a multiple expert framework, explored in [8, 9, 13]. Also, one can use an adaptive threshold setting, by using statistics of the dissimilarity measure within a sliding window [2, 11, 14]. In this work we present some new methods for implementing an adaptive threshold and compare them with existing ones. We also provide a thorough examination of the improvement obtained over the basic techniques.

In the next section we examine the assumptions made on the results of the dissimilarity measures and discuss how to improve the methods by using an adaptive thresholding strategy. In Section 3 we present a description of our experimental data and an outline of our baseline shot detection algorithms. We then describe the strategies we employ in Section 4. Experimental results are detailed in Section 5 and conclusions in Section 6.

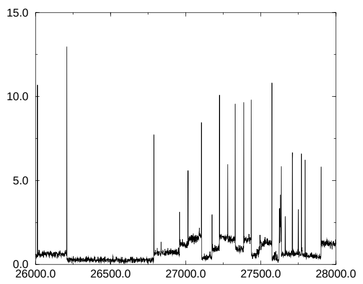
2 Modelling the dissimilarity measure statistics

In [12] we described a set of algorithms for detecting shot cuts. In each of these methods, a single statistic m is generated for each pair of frames to quantify the degree of dissimilarity between the two frames. We make the assumption that these dissimilarity measures

$\{m\}$ come from one of two distributions: one for shot boundaries ($\mathcal{S}$) and one for "not-a-shot-boundary" ($\mathcal{N}$). In general, $\mathcal{S}$ has a considerably larger mean and standard deviation than $\mathcal{N}$ (Fig. 1(a)). If, for example, we assume that the costs of false positives and undetected true positives are the same, and that the distribution statistics are stationary, the standard classification methods would indicate that the decision threshold m_T should be fixed so that the tails of the two density functions $p_{\mathcal{S}}$ and $p_{\mathcal{N}}$ (shown shaded in Fig. 1(a)) have an equal area. Because of the difference between the widths of the two distributions, we can see that this threshold is fairly close to the mean $\mu_{\mathcal{N}}$ of $\mathcal{N}$, and hence that it is important that the position and width of $\mathcal{N}$ are accurately determined.


p(m)
$p_{\mathcal{N}}(m)$ (not a shot boundary)
$p_{\mathcal{S}}(m)$ (shot boundary)
m_T
m

(a) Distributions of dissimilarity measure m



(b) Plot of dissimilarity measure against frame number

Figure 1: Representation of dissimilarity measures.

In a previous work [13] it was implicitly assumed that the distributions were indeed stationary, and thus that a single decision threshold could be used. To find this threshold, we experimented with a range of thresholds until the best value was found. In practice however we found, not surprisingly, that the stationarity assumption for $\mathcal{N}$ does not hold up well. In particular, we realised that $p_{\mathcal{N}}$ often varies gradually within a shot, and abruptly at shot boundaries. This can be seen from Fig. 1(b), which shows an example of the dissimilarity measure (in this case the mean absolute pixel difference) as a function of frame number, for a sequence of off-air news material. (The sharp peaks correspond to shot boundaries.) In this sequence we can see that the mean level in particular appears to change gradually if at all within a shot, but jumps significantly in value at the shot boundaries. Furthermore, experiments demonstrated that this single decision threshold can be consistently grossly over- or underestimated when applied to video material with distinctive characteristics, such as sports events or cartoons.

This suggests that it may be possible to improve the detection performance by estimating $p_{\mathcal{N}}$ dynamically, using the dissimilarity measures from the previous and next few frames, and using the result to adaptively set the detection threshold. In practice, we estimate the mean $\mu_{\mathcal{N}}$ and possibly the variance $\sigma_{\mathcal{N}}$ in this way. We then set the threshold m_T to be some function of these statistics, e.g. some fixed distance from $\mu_{\mathcal{N}}$, or some multiple of $\sqrt{\sigma_{\mathcal{N}}}$ from $\mu_{\mathcal{N}}$. The method therefore uses a sliding window of a predetermined size where only the samples within this window are considered for estimating $p_{\mathcal{N}}$.

Since frame pairs that include a shot boundary are relatively rare occurrences compared with pairs that do not, we have to assume that p_S is stationary (Fig. 1(a)).

Investigation of other work in this area has produced some examples which did not fully exploit this possibility. In [14], for example, the authors used a sliding window but a fixed threshold. In [11], the threshold is represented as a multiple of the second highest sample within the window. It is in [2] that a structured method to adaptively set the detection threshold is presented. This is described further in Section 4. However, no quantifiable results were presented in these works. Also, there are other ways in which the adaptive approach can be implemented. Thus we set out to compare the performance of adapted versus non-adapted algorithms and contrast between several methods which we have developed for the adaptation process and that proposed by Dugad *et. al* in [2].

3 The experimental data and baseline algorithms

In our main data set, which consists of several off air sequences consisting of news programs, documentaries, children's shows, daytime soaps, etc., we aimed at capturing a broad variety of material. We also used two other sequences: a collection of various cartoons, and a rugby league match. These two sequences represent two different extremes in terms of content, with different statistical properties from the main set. Table 1 details the composition of our test data.

Table 1: Video Sequences used in the experiments.

Format	QCIF – 176 × 144 pixels YUV 4:2:0		
Frame rate	25fps		
Name	No. of frames	Time (mins)	No. of shot cuts
GENERAL	161928	108	1160
CARTOON	41750	27.8	256
RUGBY	40490	27	257

In this work, four separate algorithms are used to detect shot changes. These algorithms calculate different features of the video data, and can be used by themselves as stand-alone shot boundary detection systems. These methods are:

1. Average Intensity Measurement (AIM) [3]

   The algorithm computes the average of the intensity values for each component (YUV, RGB, etc.) in the current frame and compares it with that for the following frame.

2. Histogram Comparison (HC) [7, 15, 6]

   Histogram comparison methods are quite popular because they are fast and motion-insensitive. Our implementation is similar to that detailed in [15]. However, we extended it by including colour components as well.

3. Likelihood Ratio (LH) [5]

   Each region is represented by second order statistics under the assumption that this property remains constant over the region. We divide the frames into blocks, and carry out the likelihood ratio calculation over the blocks.

4. Motion Estimation / Prediction Error (ME)

   We estimate the next frame in a video sequence based on the motion information in the current frame and reconstruct the next frame using the motion vectors. We used the block-based n-step search algorithm for a $\pm 2^n$ search window as described in [10]. To obtain the dissimilarity measure, the mean absolute difference between the reconstructed frame and the original frame is calculated.

4 The adaptive thresholding scheme

In [11], the authors applied a local thresholding method whereby the frame differences of successive m frames is examined. They then declare a shot change when two conditions are simultaneously satisfied:

1. the difference is the maximum within a symmetric sliding windows of size $2m - 1$
2. the difference is n times the second largest maximum in the sliding window.

Expanding from this work, a method was proposed in which the means and standard deviations from either side of the middle sample in the window is calculated [2]. The middle sample represents a shot change if the conditions below are simultaneously satisfied:

1. The middle sample is the maximum in the window
2. The middle sample is greater than $\max(\mu_{left} + T_d\sqrt{\sigma_{left}},\ \mu_{right} + T_d\sqrt{\sigma_{right}})$ where T_d is given a value of 5.

In our work, we experimented with three different methods of estimating the decision threshold. However they all follow the same principle. We describe the overall scheme next, and then describe the individual thresholding methods in Section 4.2.

4.1 The general method

The general method is as follows. We estimate the mean $\mu_{\mathcal{N}}$ (and variance $\sigma_{\mathcal{N}}$ if required) of $\mathcal{N}$ dynamically, from the similarity measures m of M neighbouring frames. The decision threshold m_T is recalculated for each new frame, using one of the methods discussed in Section 4.2, and a decision made. However, after a shot cut is detected, no new decisions are made until $M/2$ frames have elapsed.

4.2 Computing the decision threshold

We next describe the three different models for setting the threshold.

Constant variance model Here we assume that:

(a) the distributions are unimodal,

(b) $\mu_{\mathcal{N}}$ varies over a small enough range and $p_{\mathcal{S}}$ is sufficiently broad that changes in the value of $p_{\mathcal{S}}$ in the region of the intersection of the density functions can be ignored,

(c) apart from $\mu_{\mathcal{N}}$, the distributions are stationary.

These assumptions suggest that the threshold could be set at some fixed positive offset from $\mu_{\mathcal{N}}$:

$$m_T = \mu_N + T_c \tag{1}$$

Thus T_c reflects the width of $\mathcal{N}$ in some way. The best value of T_c is determined by experimenting with a range of values on a training set of video material for which ground truth information is available.

Proportional variance model If on the other hand the variance, $\sigma_{\mathcal{N}}$, is assumed to vary with ${\mu_{\mathcal{N}}}^2$, we should set the threshold at some multiple of $\mu_{\mathcal{N}}$:

$$m_T = T_p \mu_N \tag{2}$$

In this case, the width of $\mathcal{N}$ is reflected in the value of $T_p\,\mu_{\mathcal{N}}$. The value of T_p is also determined from experimentation.

The *Dugad* model As we described earlier, this is an implementation of the model proposed by Dugad, *et. al* [2], as given below:

$$m_T = \mu_N + T_d \sqrt{\sigma_N} \tag{3}$$

As explained earlier, the authors calculate the means and standard deviations on the left and right of the centre sample and use the maximum of the two. They applied this method on a histogram comparison algorithm and their experimentation gave them an optimum value for T_d as 5. Since we have four base methods, and our histogram comparison methods may not be exactly equivalent, we carried out our experiments, as with the other models, over a range of values to get an optimum T_d.

5 Experimental Results

5.1 Organisation

The three models we have from Section 4.2 are:

1. Constant Variance Model – the threshold is Added to the mean of the samples. We label this as A.
2. Proportional Variance Model – the threshold is Multiplied by the mean of the samples. We label this M.
3. The *Dugad, et. al* Model – the threshold is multiplied by the standard Deviation of the samples and added to the mean. We label this D.

Also, for each model, we adopted two different strategies as follows, leading to six different methods:

1. In the method shown in [2], the window is split into two halves on either side of the centre sample. This method is prefixed with the letter D (Dual windows). Thus a split window using the Dugad model would be labelled *DD*.
2. Another strategy is to include all the samples, including the centre sample, into the calculation. This method is prefixed with the letter S (Single window). Thus a single window strategy using the Multiplicative method would be labelled *SM*.

Table 2 indicates the optimal window sizes for each method, and for each of the algorithms described in Section 3, as explained in Section 5.3.

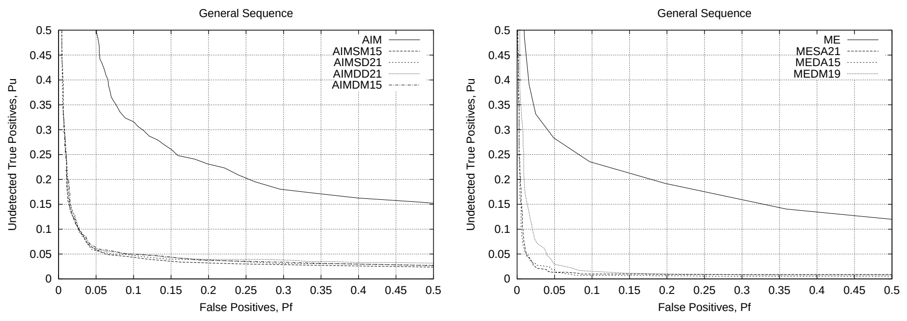
We then apply a range of thresholds for each of the models and construct a Receiver Operating Curve (ROC). In the ROC graphs, p_u denotes the proportion of false positives, and p_f denotes the proportion of undetected true positives. For comparison purposes, we consider the equal error rate as our performance criteria. Regardless of this, some applications may have different requirements such as the minimal number of undetected true positives, at the expense of a higher false positive rate if need be.

Table 2: Optimal window sizes

Method	SA	DA	SM	DM	SD	DD
AIM	25	25	15	15	21	21
HC	17	9	9	17	21	21
LH	9	9	9	11	9	21
ME	21	15	19	21	21	21

5.2 Comparison with non-adaptive results

We have discovered that the adaptive thresholding methods have shown significantly better results for all the algorithms concerned. This is evident for all the methods that we employed. This is demonstrated in Fig. 2 where the results for the AIM and ME algorithms and a selection of the methods are shown.



(a) AIM (b) ME

Figure 2: Comparison of non-adaptive measures and a selection of adaptive methods.

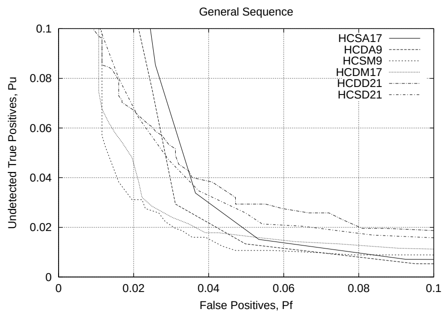
5.3 Effects of varying the window size

We initially used window sizes of 9, 11, 15, 21, 25 and 29. If the results show a simple trend, we test all of the window sizes between the two best performing ones and take the best of these as the optimum. Otherwise, we test all of the window sizes in the range 3 to 31.

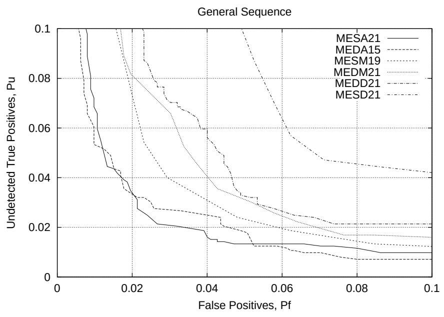
Initially, increasing the window size tends to increase the accuracy of the shot detection, and eventually it decreases. This is not a universal behaviour, however. The LH and HC algorithms are generally best with a small window size, getting worse as the size increases. For the LH algorithm using the DD method, the results are fairly constant until the window size is large, where it gets worse. On the other hand, the ME algorithm using DD or SF shows erratic behaviour, improving for larger window sizes.

5.4 Determining the best performing adaptive thresholding method

First, we plot the 6 adaptive methods for each algorithm and select the best one. Fig. 3 shows examples for HC and ME. Then we plot these “best of method” results against each other to come up with the best overall (Fig. 4).



(a) HC



(b) ME

Figure 3: ROC curves of the adaptive thresholding methods with their optimum window sizes.

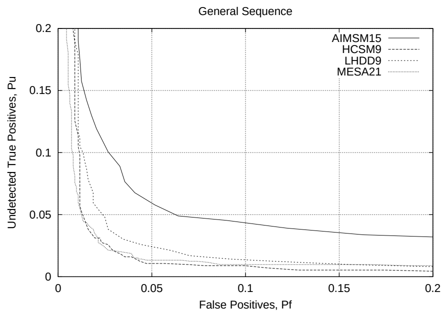


Figure 4: ROC curves of best performing algorithm and adaptive thresholding methods

For an equal error rate, the best performing methods are MESA21 and HCSM9, with MESA21 being marginally better (Fig. 4) due to a lower false detection rate. However, if we want to minimise the undetected true positive rate, then HCSM9 would be better. MESA21 is best with a large window size, which causes it to miss cuts which are within the window. Our experiments have shown that there are a few of these cuts in our experimental data, mainly during commercials.

In Fig. 5, we give an example of an instance where a true positive was detected by HCSM9 and not by MESA21. Note that there is some blurring due to excessive camera zoom on the shot left of the cut. This gave a high error rate for our motion prediction as shown in Fig. 6(a). For comparison, we show the dissimilarity measure for the HCSM9 case in Fig. 6(b).

5.5 Sequences that cause problems

We tested our baseline methods (Section 3) on the animation sequence and the sports program and discovered that the results were very poor as shown in Fig. 7. It was then interesting to see if the performance of the adaptive thresholding models would be as good

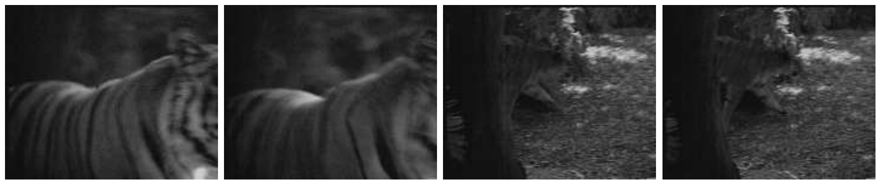
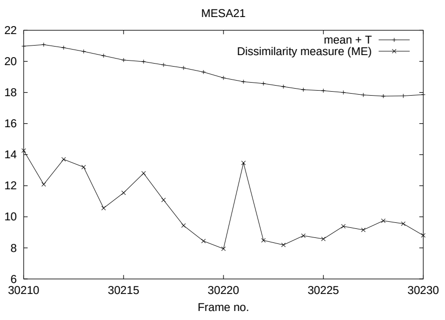
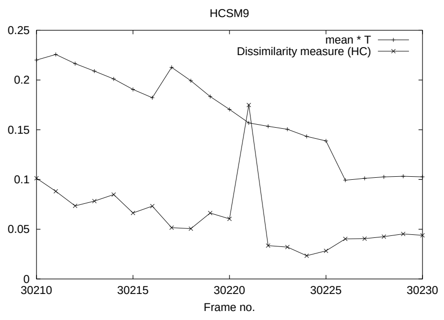


Figure 5: MESA21 Undetected True Positive



(a) MESA21, using the model of Eq 1



(b) HCSM9, using the model of Eq. 2

Figure 6: Graphs showing how the adaptive threshold varies with the dissimilarity measure for MESA21 and HCSM9. In each case the bottom trace is the dissimilarity measure, and the top trace is the adaptive threshold.

as that of our main data set, the *General Sequence*.

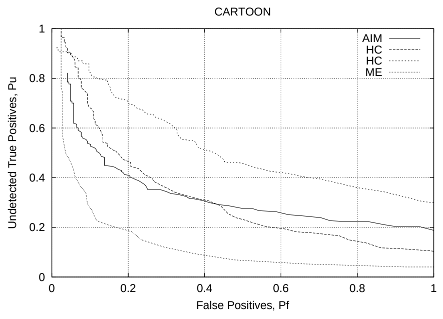
We went through the same procedure as for the main data set to determine the best performing models. In the case of the animation sequence, *Cartoon*, the best baseline method is ME and for the sport sequence, *Rugby*, HC. As shown in Fig. 8, the adaptive methods give a considerable improvement in the detection rate for both these sequences.

These two sequences differ from our main data set. In the animation sequence, the content is synthetic and the maximum number of distinct colours was only 256, which we believe to be the norm for television cartoons. We have also noticed that on occasions where there is relatively little action over the course of a shot, each frame is repeated (giving an effective $12\frac{1}{2}$ frames per second rate).

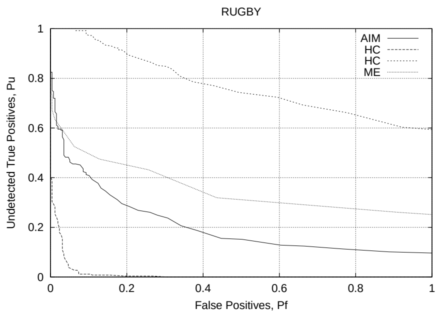
For the sports sequence, the main feature was the presence of busy camera movements and high speed action. Also, there were numerous shots of objects moving across the field of view whilst the camera was focused on action further ahead in the distance.

6 Conclusion

We have shown that adaptive thresholding considerably improves the rate of detection for shot cuts regardless of the method used. In some cases, the improvements can be

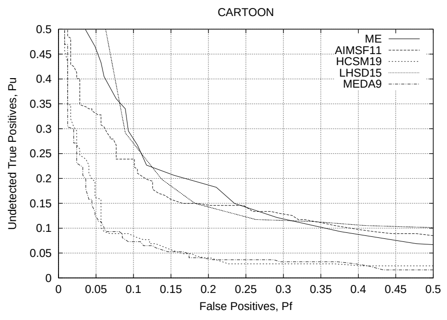


(a) Cartoon

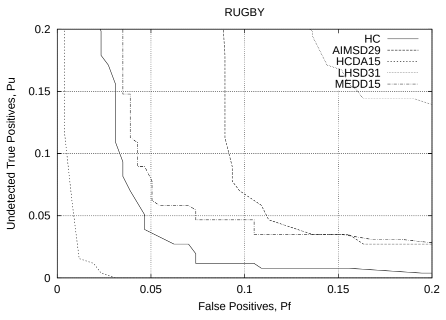


(b) Rugby

Figure 7: ROCs for the baseline algorithms.



(a) Cartoon



(b) Rugby

Figure 8: Comparison between the best baseline algorithm against the adaptively thresholded methods.

dramatic.

Our methods also show a marked improvement over the Dugad method for all the data sets which we tested. This is true for all of the baseline algorithms, not just the histogram comparison method used in [2].

It is found that the adaptively thresholded versions of ME and HC are the best performing algorithms. The ME based model achieved a marginally better equal error rate for the general sequence but is limited in its true positive detection rate due to the large window size required to achieve this. In this respect, the HC based model is better.

Interestingly, for the *Cartoon* and *Rugby* sequences, in each case there was one algorithm which was clearly better than the others, which was ME for *Cartoon* and HC for *Rugby*. This meant that the adaptive thresholding versions of these algorithms were also the best performers for the respective sequences. It also indicates that certain algorithms are better suited for some sequences than others.

Acknowledgements

The work presented in this paper has been carried out in the context of the EC 5th Framework Project IST-13082 (ASSAVID).

References

[1] J. S. Boreczky and L. A. Rowe. Comparison of video shot boundary detection techniques. *Journal of Electronic Imaging*, 5(2):122–128, April 1996.

[2] R. Dugad, K. Ratakonda, and N. Ahuja. Robust video shot change detection. *IEEE Workshop on Multimedia Signal Processing*, Dec 1998.

[3] A. Hampapur, R. Jain, and T. Weymouth. Digital video segmentation. In *Proc. ACM Multimedia '94*, pages 357–364. ACM Press, 1994.

[4] H. Jiang, A. S. Helal, A. K. Elmagarmid, and A. Joshi. Scene change detection techniques for video database systems. *ACM Multimedia Systems*, 6(3):186–195, 1998.

[5] R. Kasturi and R. Jain, editors. *Computer Vision: Principles*. IEEE Computer Society Press, 1991.

[6] R. Lienhart. Comparison of automatic shot boundary detection algorithms. In *Proc. IS&T/SPIE Conf. on Storage and Retrieval for Image and Video Databases VII*, volume 3656, pages 290–301, San Jose, CA, January 1999.

[7] A. Nagasaka and Y. Tanaka. Automatic video indexing and full-video search for object appearances. *Visual Database Systems II*, pages 113–127, 1992.

[8] M. R. Naphade, R. Mehrotra, A. M. Ferman, J. Warnick, T. S. Huang, and A. M. Tekalp. A high-performance shot boundary detection algorithm using multiple cues. In *Proc. IEEE Int. Conf. on Image Proc.*, volume 2, pages 884–887, Oct 1998.

[9] C. Taşkiran and E. J. Delp. Video scene change detection using the generalized sequence trace. *Proc. IEEE Int. Conf. on Image Proc.*, pages 2961–2964, 1998.

[10] A. Murat Tekalp. *Digital Video Processing*. Prentice-Hall, 1995.

[11] B. L. Yeo and B. Liu. Rapid scene analysis on compressed video. *IEEE Trans. on Circuits and Systems for Video Technology*, 5(6):533–544, Dec 1995.

[12] Y. Yusoff, W. Christmas, and J. Kittler. A study on automatic shot change detection. In D. Hutchison and Ralf Schafer, editors, *Proc 3rd. Eur. Conf. on Multimedia Apps, Services and Techniques (ECMAST)*, pages 177–189. Springer, May 1998.

[13] Y. Yusoff, J. Kittler, and W. Christmas. Combining multiple experts for classifying shot changes in video sequences. In *Proc. 6th Int. Conf. on Multimedia Comp. and Systems (ICMCS)*, volume 2, pages 700–704, Florence, Italy, June 1999. IEEE.

[14] R. Zabih, J. Miller, and K. Mai. A feature-based algorithm for detecting and classifying production effects. *ACM Multimedia Systems*, 7(2):119–128, 1999.

[15] H. Zhang, A. Kankanhalli, and S. W. Smoliar. Automatic partitioning of full-motion video. In *Multimedia Systems*, volume 1, pages 10–28. Springer-Verlag, 1993.