1996

Scene Change Detection for Video Database Management Systems-A Survey

Haitao Jiang

Abdelsalam (Sumi) Helal

Ahmed K. Elmagarmid
Purdue University, ake@cs.purdue.edu

Anupam Joshi

Report Number:

96-058

SCENE CHANGE DETECTION FOR VIDEO DATABASE MANAGEMENT SYSTEMS-A SURVEY

Haitao Jiang
Abdelsalam (Sumi) Helal
Ahmed K. Elmagarmid
Anupam Joshi

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

CSD-TR 96-058
October 1996

Contents

1 Introduction **3**

2 Background **5**

2.1 Basic Notations 5

2.2 MPEG Standard: Different Frame Types 5

2.3 DC Images, DC Sequences and Their Extraction 6

2.4 Basic Measurements of Inter-frame Difference 7

2.4.1 Template Matching 7

2.4.2 Color Histogram 8

2.4.3 χ^2 Histogram 8

3 Full Image Video Scene Change Detection **11**

3.1 Detecting Abrupt Scene Changes 11

3.2 Detecting Gradual Scene Changes 12

4 Scene Change Detection on the Compressed Video Data **15**

4.1 DC Image Sequence Based Approach 16

4.2 DC Coefficients Based Approach 17

5 Model–based Video Scene Change Detection **19**

6 Evaluation Criteria for the Performance of SCD Algorithms **21**

7 Conclusion **23**

Scene Change Detection for Video Database Management Systems—A Survey

Haitao Jiang
Abdelsalam (Sumi) Helal
Ahmed K. Elmagarmid
Anupam Joshi
Department of Computer Science
Purdue University
West Lafayette, IN 47907
{jiang,helal,ake,joshi}@cs.purdue.edu

Abstract

Scene change detection (SCD) is one of several fundamental problems in the design of a video database management system (VDBMS). It is the first step towards the automatic segmentation, annotation, and indexing of video data. Scene change detection is also used in other aspects of VDBMS, e. g. hierarchical representation and efficient browsing of the video data. In this paper, we provide a taxonomy that classifies existing SCD algorithms into three categories: full video image based, compressed video based, and model based algorithms. The capabilities and limitations of the SCD algorithms are discussed in detail. The paper also proposes a set of criteria for measuring and comparing the performance of various SCD algorithms. We conclude by discussing some important research directions.

keywords: Scene Change Detection, Video Segmentation, Video Databases, Survey

1 Introduction

A *video database management system* is a software that manages a collection of video data and provides content–based access to users [10]. There are four basic problems that need to be addressed in a video database management system. These are video data modeling, video data insertion, video data storage organization and management, and video data retrieval. One fundamental aspect that has a great impact on all basic problems is the content–based temporal sampling of video data [24]. The purpose of the content–based temporal sampling is to identify significant video frames to achieve better representation, indexing, storage, and retrieval of the video data. Automatic content–based temporal sampling is very difficult due to the fact that the sampling criteria are not well defined, i. e. whether a video frame is important or not is usually subjective. Moreover, it is usually highly application–dependent and requires high level, semantic interpretation of the video content. This requires the combination of very sophisticated techniques from computer vision and AI. The state of the art in those fields, however, has not advanced to the point where semantic interpretations would be possible.

However, researchers usually can get satisfying results by analyzing the visual content of the video and partitioning it into a set of basic units called *shots*. This process is also referred to as *video data segmentation*. Content–based sampling thus can be approximated by selecting one representing frame from each shot since a shot is defined as a continuous sequence of video frames which have no significant inter-frame difference in terms of their visual contents.[1] A single shot usually results from a single continuous camera operation. This partitioning is usually achieved by sequentially measuring inter-frame differences and studying their variances, e. g. detecting sharp peaks. This process is often called *scene change detection* (SCD).

Scene change in a video sequence can either be abrupt or gradual. *Abrupt*

[1]There are many definitions in the literature from different points of views. This definition seems to be the most agreed upon one

Figure 1: An example of an abrupt scene change

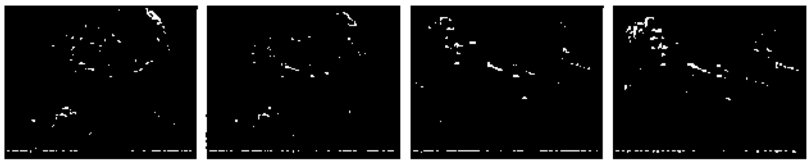

Figure 2: An example of an gradual scene change

scene changes results from editing "cuts"and detecting them is called *cut detection* [11]. Gradual scene changes results from *chromatic edits, spatial edits and combined edits* [11]. Gradual scene changes include special effects like zoom, camera pan, dissolve and fade in/out etc. An example of abrupt scene change and gradual scene change is shown in Figure 2. Scene change detection is usually based on some measurements of the image frame, which can be computed from the information contained in the images. Those information can be color, spatial correlation, object shape, motion contained in the video image, or DC coefficients in the case of compressed video data. In general, gradual scene changes are more difficult to detect than the abrupt scene changes, and may cause lots of scene detection algorithms to fail under certain circumstances.

Existing scene change detection (SCD) algorithms can be classified in many ways according to, among others, the video features they use and the video objects they can be applied on. In this paper, we discuss SCD algorithms in three main categories: (1) approaches that work on uncompressed full image sequences; (2) algorithms that aim at working directly on the compressed video; and (3) approaches that are based on explicit models. The latter are also called *top-down approaches* [10], whereas the first two categories are called *bottom-up approaches.* This paper is organized as followings. Section 2 briefly presents some background information about the SCD problem. Then, three categories of existing work are summarized in sections 3, 4, and 5, respectively. Their performance, advantages, and drawbacks are also discussed. Section 6 presents some criteria for evaluating the performance of SCD algorithms. Section 7 discusses some possible future research directions.

2 Background

We now introduce some basic notations used in this paper, followed by the notions of DC (Discrete Cosine) images, DC sequences and how they can be extracted from compressed video. Several most often used image measurements are also briefly described in terms of their use in measuring the inter-frame difference. It should be noted that they may not work well for scene detection when used separately, thus they usually are combined in the scene change detection algorithms. For example, Swanberg et al. [28] use a combination of template and histogram matching to measure the video frames.

2.1 Basic Notations

Following notations are used throughout this paper. A sequence of video images whether they are fully uncompressed or spatially reduced, are denoted as $I_i, 0 \leq i < N$, N is the length or the number of frames of the video data. $I_i(x, y)$ denotes the value of the pixel at position (x, y) for the ith frame. H_i refers to the histogram of the image I_i. The inter-frame difference between images I_i, I_j according to some measurement is represented as $d(I_i, I_j)$

2.2 MPEG Standard: Different Frame Types

According to the International Standard ISO/IEC 11172 [8], a MPEG-I compressed video stream can have one or more of following types of frames:

- I (Intra-coded) frames are coded without reference to other frames. They are coded using spatil redundnacy reduction which is a lossy block–based coding involving DCT, quantization, run length encoding, and entropy coding.
- P (Predicitive-coded) frames are coded using motion compensation predication from the last I or P frame.
- B (Bidirectionally-predictive coded) frames are coded using motion compensation with reference to both previous and next I or P frames.
- D (DC-coded) frames are coded using DC coefficients of blocks, thus only contains low frequency information. D frames are not allowed to co-exist with I/P/B frames and rarely used in practice.

Obviously, any MPEG compressed video stream must have at least I frames. The data size ratios between frames suggested by the standard are: 3:1 for I:P and 5:2 to 2:1 for P:B. In other words, B frames has the highest degree of compression and I frame has the least one. More details about MPEG video streams can be found in [8].

2.3 DC Images, DC Sequences and Their Extraction

A *DC (Discrete Cosine) image* [34, 33, 32, 31] is a spatially reduced version of a given image. It can be obtained by first dividing the original image into blocks of $n \times n$ pixels each, then computing the average value of pixels in each block which correspondes to one pixel in the DC image. For the compressed video data, e.g. MPEG video, a sequence of DC images can be constructed directly from the compressed video sequence, which is called a *DC sequence*. Figure 3 is an example of a video frame image and its DC image.

There are several advantages of using DC images and DC sequences in the SCD for the compressed video.

- DC images retain most of the essential global information for image processing. Thus, lots of analysis done on the full image can be done on its

DC image instead.

- DC images are considerably smaller than the full image frames which makes the analysis on DC images much more efficient.
- Partial decoding of compressed video save computation time than full-frame decompression.

Extracting DC images from an MPEG video stream has been described in Yeo and Liu [34, 33, 32, 31]. Extracting DC image of an I frame is trivial since it is given by its DCT (Discrete Cosine Transform) coefficients. Extracting DC images from P-frames and B-frames need use inter–frame motion information. This may result in many multiplication operations. To speed up the computation, two approximations are proposed: *zero-order* and *first-order*. The authors claim that the reduced images formed from DC coefficients, whether they are precisely or approximately computed, retain the "global features" which can be used for video data segmentation, SCD, matching and other image analysis.

2.4 Basic Measurements of Inter-frame Difference

2.4.1 Template Matching

Template matching is to compare the pixels of two images across the same location and can be formulated as

$$d(I_i, I_j) = \sum_{x=0,y=0}^{x<M,y<N} |I_i(x,y) - I_j(x,y)| \tag{1}$$

where image size is of $M \times N$. Template matching is very sensitive to noise and object movements since it is strictly tied to pixel locations. This can cause false SCD and can be overcome to some degree by partitioning the image into several subregions. Figure 4 is an example of inter–frame difference sequence based on template matching. The input video is the one that contains the first image sequence in the Figure 2.

2.4.2 Color Histogram

The color histogram of an image can be computed by dividing a color space, e.g. RGB, into discrete image colors called *bins* and counting the number of pixels fall into each bin [27]. The difference between two images I_i and I_j based on their color histograms H_i and H_j can be formulated as:

$$d(I_i, I_j) = \sum_{k=1}^{n} |H_{ik} - H_{jk}| \tag{2}$$

Which denote difference in the number of pixels of two image that fall into same bin. In the RGB color space, above formula can be written as

$$d_{RGB}(I_i, I_j) = \sum_{k}^{n}(|H_i^r(k) - H_j^r(k)| + |H_i^g(k) - H_j^g(k)| + |H_i^b(k) - H_j^b(k)|) \tag{3}$$

Using only simple color histogram may not detect the scene changes very well since two images can be very different in structure and yet have similar pixel values. Figure 5 is the inter-frame difference sequence of the same video data as in Figure 2 with color histogram measurement.

2.4.3 χ^2 Histogram

The χ^2 histogram computes the distance measure between two image frames as:

$$d(I_i, I_j) = \sum_{k=1}^{n} \frac{(H_{i(k)} - H_{j(k)})^2}{H_{j(k)}} \tag{4}$$

Several researchers [19, 37, 38] have used χ^2 histogram in their SCD algorithms and they report that it generates better results compared to other intensity-based measurements, e. g. color histogram and template matching. Figure 6 is the inter-frame difference sequence of the same video data as in Figure 2, but computed using χ^2 histogram.

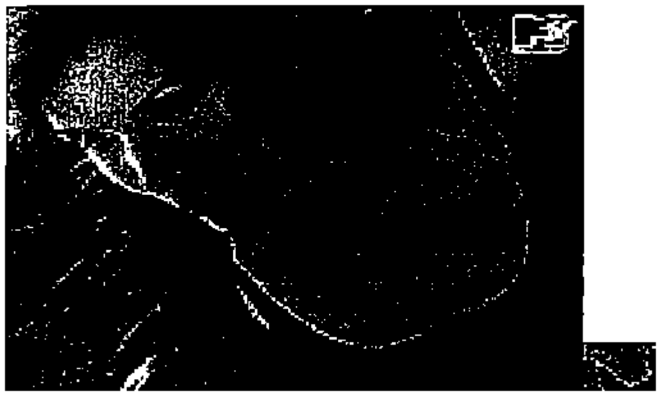

Figure 3: An Example of a full image and its DC Image

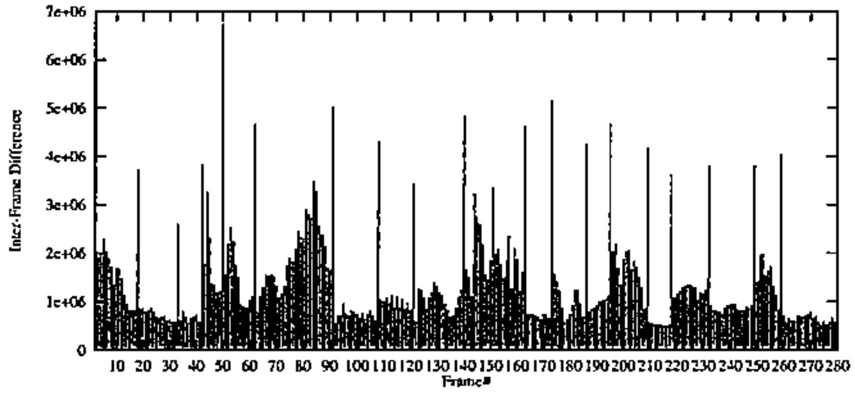

Figure 4: Template matching

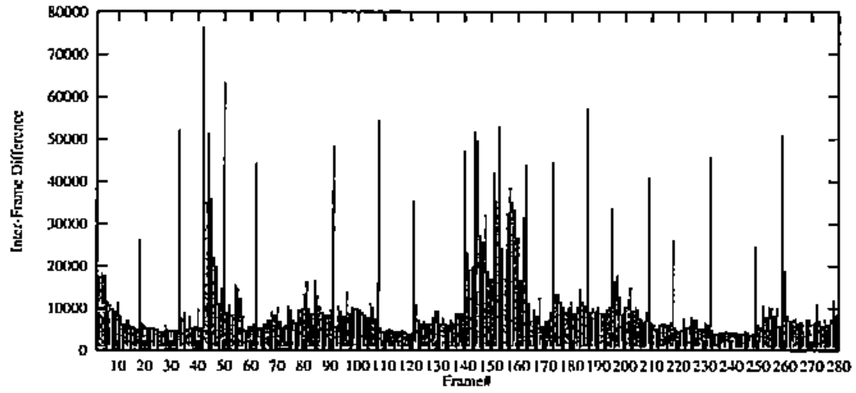

Figure 5: Color Histogram

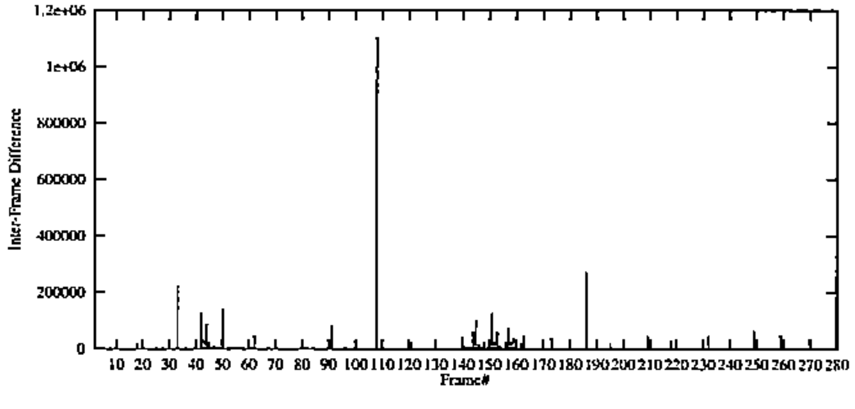

Figure 6: χ^2 Histogram

3 Full Image Video Scene Change Detection

Most of the existing work on SCD is based on full image video analysis. The differences between the various SCD approaches are the measurement function used, features chosen, and the subdivision of the frame images. Many use either intensity feature [19, 21, 20, 30, 37, 38] or motion information [2, 13, 24] of the video data to compute the inter-frame difference sequence. The problem with intensity based approaches is that they may fail when there is a peak introduced by object or camera motion. Motion based algorithms also has the drawback of being computationally expensive since they usually need to match the image blocks across the frames. After the inter-frame differences are computed, some approaches use a global threshold to decide a scene change. This is clearly insufficient since a large global difference does not necessarily implies there is a scene change as reported, for example, by Yeo [34, 32]. In fact, scene changes with globally low peaks is one of the situations that often causes the failure of the algorithms. Scene changes, either abrupt or gradual, is a localized process, and should be checked accordingly.

3.1 Detecting Abrupt Scene Changes

Algorithms for detecting abrupt scene changes have been proposed by Nagasaka et al. [19], Hsu [13], Otsuji [21, 20], and Akutsu [2]. Over 90% accuracy rate has been achieved. However, these approaches do not take into account gradual scene changes.

Nagasaka and Tanaka [19] presented an approach that partitions the video frames into 4×4 equal-sized windows and compares the corresponding windows from the two frames. Every pair of windows are compared and the largest difference is discard. The difference values left are used to make the final decision. The purpose of the subdivision is to make the algorithm more tolerant to object movement, camera movement, and zooms. Six different type of measurement functions, namely difference of gray level sums, template matching, difference of gray level histograms, color template matching, difference of color histogram, and a χ^2 comparison of the color histograms, have been tested. The experimental results indicate that a combination of image subdivision and χ^2 color histogram approach provides the best results of detecting scene changes. The disadvantage of this approach is that it may miss gradual scene transition such as fading.

Otsuji [21, 20] computed both the histogram and the pixel-bases inter-frame difference based on brightness information to detect scene changes. Projection detection filter is also proposed for more reliable scene detection. The gradual scene changes are not taken into consideration.

Akutsu et al. [2] used both the average inter–frame correlation coefficient and the ratio of velocity to motion in each frame of the video to detect scene change. Their assumptions were that (1) the inter–frame correlation between frames from the same scene should be high, and (2) the ratio of velocity to motion across a cut should be high also. The approach doesn't address gradual scene changes and is computationally expensive since computing motion vectors requires the matching of image blocks across frames. Also, how to combine above two measurements to achieve better result is not clear from the paper.

Hsu et al. [13] treated the scene changes and activities in the video stream as a set of motion discontinuities which change the shape of the spatiotemporal surfaces. The sign of the Gaussian and mean curvature of the spatiotemporal surfaces is used to characterize the activities. Scene changes are detected using a empirically chosen global threshold. Clustering and split-and-merge approach are then taken to segment the video. The experimental results in the paper are not sufficient to make any judgment on the approach and no comparison results with other existing algorithms is available.

3.2 Detecting Gradual Scene Changes

Recently, more and more researchers have studied the methods for detecting both abrupt and gradual scene changes [30, 37, 38, 24]. Robust gradual SCD is more challenging than its abrupt counterpart, especially when there is a lot of motion involved. Unlike abrupt scene changes, a gradual scene change does not usually manifest itself by sharp peaks in the inter-frame difference sequence, and can be easily confused with object or camera motion. Gradual scene changes are usually determined by observing the behavior of the inter-frame differences over a certain period of time.

Tonomura et al., [30] used a comparison of an extended set of frames before and after the current frame to determine if the current frame is a cut. They also proposed to detect gradual scene changes by checking if the inter-frame differences over extended periods of time exceed a threshold value. However, lack of sufficient details and experimental results make it very difficult to judge the algorithm.

Zhang et al. [37, 38] evaluated four scene change detection approaches: template matching, the likelihood ratio between two images, histogram comparison, and χ^2 histogram comparison. They conclude that histogram comparison performs better in terms of computation cost. In their approach, gradual transitions are detected using so called *twin comparison* technique. Two thresholds $T_b, T_s, T_s < T_b$ are set for camera breaks and gradual transition respectively. If the histogram value difference $d(I_i, I_{i+1})$ between consecutive frames with difference values satisfies $T_s < d(I_i, I_{i+1}) < T_b$, they are considered potential start frames for the gradual transition. For every potential frame detected, an accumulated comparison $A_c(i) = D(I_i, I_{i+1})$ is computed till $A_c(i) > T_b$ and $d(I_i, I_{i+1}) < T_s$. The end of the gradual transition is declared when this condition is satisfied. To distinguish gradual transition from other camera operations like pans and zooms, the approach uses image flow computations. Gradual transition result in a null optical flow where there are other camera operations result in particular types of flows. Their approach achieves good results. Failures are either due to similarity of color histograms across the shots when color contents are very similar, or sharp changes in lighting such as flashes and flickering object.

Shahraray [24] detected abrupt and gradual scene changes based on motion-controlled temporal filtering of the disparity between consecutive frames. Each image frame is subdivided and image block matching is done based on image intensity values. A nonlinear order statistical filter [17] is used to combine the image matching values of different image blocks, i. e. the weight of an image match value in the total sum dependents on its order in the image match value list. The author claims that this match measure of two images is more consistent with the human's judgement.

Abrupt scene is detected by a thresholding process used by many existing algorithms that are discussed in this paper. Gradual transition is detected by identification of sustained low-level increase in image matching values. False detection due to the camera and object motions are supressed by both image block matching and temporal filtering of the image matching value sequence.

Shahraray [24] also mentioned a simple and interesting idea of verifying the scene detection results which he called *scene verification.* The idea is to measure inter-frame difference of representing frames result from the SCD algorithm, and high similarity would likely to indicate a false detection. It is reported that this algorithm is capable of processing 160×120 pixels video in real time on a Pentium PC, and has been extensively tested on a variety of TV boardcasts for more than one year. However, no statistical data about the accuracy of SCD is given in this paper.

To improve the result of detecting fades, dissolves, and wipes which most existing algorithms have difficulties with, Zabih et al. [36] proposed an algorithm based on edge changing fraction. They observed that new intensity edges appear (enter the scene) far from the locations of old edges during a scene change, and that old edges disappear (exit the scene) far from the locations of old edges. Abrupt scene changes, fades, and dissolves are detected by studying the peak values in a fixed window of frames. Wipes can be identified by the distribution of the entering and exiting edge pixels. A global computation is used to guard the algorithm from the camera or object motion. The algorithm has been tested on a data set available on the internet MPEG movie archive, and experimental results indicate that the algorithm is robust against the parameter variances, compression loss, and sub-sampling of the frame images. The algorithm performs well in detecting the fades, dissolves, and wipes but may fail in case of very rapid changes in lighting and fast moving object. It may also have difficulties being applied to video that is very dim where no edge can be detected. Initial implementation of the algorithm is about two frames/second on a SUN workstation.

4 Scene Change Detection on the Compressed Video Data

To efficiently transmit and store video data, several video compression schemes (MPEG, DVI, motion JPEG etc.) have been proposed and standardized. To detect scene changes on those video streams, two approaches can be taken:

- Fully decompress the video data into a sequence of image frames and then perform the video scene analysis on full images, i.e. use the algorithms discussed in the last section. However, fully decompressing the compressed video data can be computationally intensive. For example, it involves Hoffmann code decoding, inverse DPCM, inverse quantization, inverse DCT, and motion compensation steps in the case of MPEG compressed data.
- To speed up the scene analysis, some researchers developed SCD algorithms for the compressed video data without full decompression step. Those approaches are introduced in this section. They have been shown [31, 34, 32, 35] to be capable of producing similar results as the full image based approach, but much more efficient.

Most of work has been on the DCT based standard compressed video, e. g. MPEG [9]. Therefore, all SCD algorithms in this category are based on the DCT related information which can be extracted from the compressed video. Some algorithms operate on the corresponding DC image sequences of the compressed video [31, 34, 32], while some use DC coefficients and motion vectors instead [7, 23, 39, 18, 15]. They all only need partial decompression of the video as compared to those algorithms described in the Section 3.

4.1 DC Image Sequence Based Approach

Yeo and Liu [31, 34, 32] proposed to detect scene changes on the DC image sequence of the compressed video data. They [34] discussed following measurements: successive pixels difference (template matching) and global color statistic comparison (RGB color histogram). Template matching is sensitive to the camera and object motion, and may not produce good result as in the full frame image case. However, this measurement is more suitable for DC sequences because DC sequences are smoothed version of the corresponding full images and thus less sensitive to the camera and object movements. Based on comparison experiment results, global color statistic comparison is found to be less sensitive to the motion, but more expensive to be computed. Template matching is usually sufficient in most cases and used in their algorithm.

Abrupt scene changes are detected by first computing the inter-frame difference sequence, and then applying a slide window of size m. A scene change is found if

- The difference between two frames is the maximum within a symmetric window of size $2m - 1$.
- The difference is also n times of the second largest difference in the window. This criteria is for the purpose of guarding false SCD because of fast panning, zooming, or camera flashing.

The window size m is set to be smaller than the minimum number of frames between any scene change. The selection of parameters n and m relates to the balance of the trade off between missed detection rate and false detection rate, typical values can be $n = 3$ and $m = 10$. The sensitivity of these parameters are also experimented and studied. The selection of Gradual scene change may escape from above method.

Gradual scene changes can also be captured by computing and studying the difference of every frame with the previous kth frame, i. e. checking if a "plateau" appears in the difference sequence. They also discuss the detection of flashing-light scenes which may indicate the happening of important events or appearance of important person. Flashing-light scenes can be located by noticing two consecutive sharp peaks in a difference sequence, i.e. in a slide window of the difference sequence:

- the maximum and second largest difference values are very close.
- the two largest difference values are much larger than the average value of the rest.

Detecting scenes with captions is also studied. Their experimental results indicate that over 99% of abrupt changes and 89.5% of gradual changes have been detected and the algorithm about 70 times faster than that on the full image sequence. This conforms to the fact that DC images for the MPEG video is only $\frac{1}{64}$ of their original size. Although there may exist situations that DC images are not sufficient to detect some features [31], this approach is nonetheless very promising and produces best results in the literature.

4.2 DC Coefficients Based Approach

Arman et al. [7] detected scene changes directly on Motion JPEG compressed video data using the DC coefficients. A frame in the compressed video sequence is represented by a subset of blocks. A subset of the AC coefficients of the 8×8 DCT blocks is chosen to form a vector. It is assumed that the inner product of the vectors from the same scene is small. A global thresholded is used to scene changes, and in case there is an uncertainty, a few neighboring frames are selected for further decompression, and color histograms are used on those decompressed frames to find the location of scene change. This approach is computationally efficient. However, it does not address the gradual transition like fade and dissolving, and the experimental evaluation of the technique is not very sufficient.

Sethi and Patel [23] used only the DC coefficients of I frames of a MPEG compressed video to detect scene changes based on luminance histogram. The basic idea is that if two video frames belong to the same scene, their statistical luminance distribution should derived from a single statistical distribution. If they are not, a scene change can be declared. Their algorithm works as following: first, I frames are extracted from the compressed video streams; second, the luminance histograms of the I frames are generated by using the first DC coefficient. In the third step, the luminance histograms of consecutive frames are compared using one of the three statistical tests (Yakimovsky's likelihood ratio test, the χ^2 histogram comparison test or the Kolmogrov-Smirnov test which compares the cumulative distributions of the two data sets). Different types of video data have been used to test the algorithm, and χ^2 histogram comparison seems yield better results.

Zhang et al. [39] used DCT blocks and vector information of the MPEG compressed video data to detect scene changes based on a count of non-zero motion vectors. Their observation is that the number of valid motion vectors in P or B frames tend to be low when such frames lie between two different shots. Those frames are then decompressed and full image analysis are done to detect scene changes. The weakness of this approach is that motion-compensation related information tend to be unreliable and unpredictable in the case of gradual transitions, which causes the approach to fail.

Meng et al. [18] used the variance of DC coefficients in I and P frames and motion vector information to characterize scene changes of MPEG-I and MPEG-II video streams. The basic idea of their approach is that frames tend to have very different motion vector ratios if they belong to different scene, and to have very similar motion vector ratios if they are within the same scene.
The scene detection algorithm works as following. First, a MPEG video is decoded just enough to obtain the motion vectors and DC coefficients, and then inverse motion compensation is applied only to the luminance micro-blocks of P frames to construct their DC coefficients. Then the suspected frames are marked:

- A I frame is marked if there is a peak inter-frame histogram difference and the immediate B frame before it has a peak value of the ratio between forward and backward motion vectors.

- A P frame is marked if there is a peak in its ratio of intra coded blocks and forward motion vectors.
- A B is marked if its backward and forward motion vector ratio has a peak value.

Final decisions are made by going through marked frames to check if they satisfy the local window threshold. The threshold is set according to the estimated minimal scene change distance. Dissolve effect is determined by noticing a parabolic variance curve.

As more and more video data are compressed and made available on the internet and World Wide Web, above scene changes detection certainly make themselves good choices in many cases. However, we should notice their limitations. First, current video compression standards like MPEG are optimized for data compression rather than for the representation of the visual content and they are lossy, e.g. they do not necessarily produce accurate motion vectors [36]. Second, motion vectors are not always readily obtainable from the compressed video data since large portion of the existing MPEG video has I frames only [36]. Moreover, some of the important image analysis, e.g. automatic caption extraction and recognition, may not possible on the compressed data.

5 Model–based Video Scene Change Detection

All the research works introduced so far are solely based on the image processing techniques. It is, however, possible to build an explicit model of the video data to help the SCD process [12, 10, 1]. These algorithms are sometimes referred to as top-down approaches [12, 10], whereas algorithms in Sections 3 and 4 are known as bottom-up approaches. The advantages of the model–based SCD is that a systematic procedure based on mathematical models can be developed, and certain domain specific constraints can be added to improve the effectiveness of the approaches [12]. The performance of such algorithms depends on the models they based on.

Hampapur et al. [12, 10] used a *production model–based classification* for video segmentation. Based on a study of the video production process and different constraints abstracted from it, a *video edit model* was proposed which captures the process of video editing and assembly. The model include three components: *edit decision model, assembly model,* and *edit effect model.* The edit effect model contains both abrupt scene change (cut) and gradual scene changes (translate, fade, dissolve and morphing etc.). Template matching and χ^2 histogram measurements are used. Gradual scene changes (they called chromatic edits) like fade and dissolve are modeled as chromatic scaling operations. Fade is modeled as a chromatic scaling operation with positive and negative fade rate, and dissolve is modeled as a simultaneous chromatics scaling operations of two images. The first step of their algorithm is to identify the features which correspond to each of the edit classes to be detected, then classify the video frames based on these features. Feature vectors extracted from the video data are used together with the mathematical models to classify the video frames and to detect any edit boundaries. Their approach has been tested using the cable TV program video with cut, fade, dissolve and spatial edits, with an overall 88% accurate rate being reported [10].

Aigrain and Joly [1] proposed an algorithm based on a differential model of the distribution of pixel value differences in a motion picture which includes:

- a small amplitude additive zero-centered Gaussian noise which models camera, film and other noises.
- a intrashot change model for pixel change probablity distribution results from object, camera motion, or angle change, focus or light change at a given time and in a given shot. The mode can be expressed as:

  $$P(s) = k\left[\frac{a-|s|}{a^2}\right] + (1-k)\alpha e^{-\alpha|s|}$$

  where a is the number of grey levels, k is the proportion of auto–correlated

Type	Model	Notes
Cut	$Q(s) = \frac{2(a-\|s\|)}{a(a-1)}$	a is the number of grey level d is the duration of change
Wipe	$Q(s) = \frac{da(a-\|s\|)}{a(a-1)}$	same as above
Dissolve	$Q(s) = \frac{2d(a-d\|s\|)}{a(a-1)}$ for $d\|s\| \leq a$ $Q(s) = 0$ for $d\|s\| > a$	same as above
Fade (to/from white, to/from black)	$Q(s) = \frac{d}{a}$ for $d\|s\| \leq a$ $Q(s) = 0$ for $d\|s\| > a$	$s > 0$ for fade from black or to white $s < 0$ for fade from white or to black

Table 1: Pixel Difference Distribution Models for Scene Changes

pixels, α and s are variables.

- a set of shot transition model for different kinds of abrupt and gradual scene changes which are assumed to be linear. They can be summerized in the Table 5.

The first step of their SCD algorithm is to reduce the resolution of frame images by undersampling. Its purpose is to overcome the effects of the camera and object motion, as well as to make the computation more efficient in the following steps. Second, compute the histogram of pixel difference values, and count the number of pixels whose change of value is within a certain range determined by studying above models. Different scene changes are then detected by checking the resulting integer sequence. Experiments show that their algorithm can achieve 94%–100% detecion rate for abrupt scene changes, and around 80% for gradual scene changes.

6 Evaluation Criteria for the Performance of SCD Algorithms

It is difficult to evaluate and compare existing SCD algorithms due to the lack of objective performance measurements. This is mainly attributed to the diversity of factors involved in the video data. However, various video resources can be used to test and compare algorithms against some user and application independent evaluation criteria, and give us indications of their effectiveness. Unfortunately, there is no widely accepted test video data set currently available and lots of researchers use MPEG movies available on a few WWW archive sites[2] as inputs to their SCD algorithms. Such video data may not be a good benchmark to use in testing SCD algorithms. There are several reasons for this. First, the purpose of these movies was not for benchmarking SCD algorithms. So, although some of them may take half an hour to download and may occupy very large disk space (a 1 minute MPEG video can easily takes up over 5MB storage space depending on the encoding method), they may not be a good representive data set for all the possible scene change types. Second, the qualities of those movies varies greatly since they come from different sources and are encoded using various coding algorithms. For example, many MPEG movies have only I frames only which may cause problems to some SCD algorithms on compressed video data. Third, there is no widely accepted "correct" SCD results available for any of those MPEG data sets. Thus, an effort towards building a public accessible library of SCD test video data sets will be very useful. Such test data set should include video data from various applications which covers different types of scene change, along with the analysis results made and agreed upon by researchers.

We argue that performance measurements of SCD algorithms should include one or more of following:

- CPU time spent for a given video benchmark, e. g. the number of frames processed by SCD algorithm per time unit.
- average success rate or failure rate for SCD over various video benchmarks. The failure includes both false detection and missed detection, for example, 100% scene change capture rate does not necessarily imply that the algorithm is good since it may have very high false change alarms. The results of a SCD algorithm can be compared to human SCD results

[2]For example, http://w3.eeb.ele.tue.nl/mpeg/index.html

which can be assumed to be correct.

- SCD granularity. Can the algorithm decide between which frames a scene change occurs? Furthermore, can it also report the type of the scene change, i. e. whether it is fade in or dissolve?
- Stability, i. e. its sensitivity to the noise in the video stream. Very often, flashing of the scene and background noises can trigger the false detection.
- Types of the scene changes and special effects that it can handle.
- Generality. Can it be applied to various applications? i. e. What are the different kind of video data resources can it handle?
- Formats of the video it can accept (full image sequence, MPEG-I, MPEG-II, or AVI video etc.).

7 Conclusion

In this paper, a taxnomy of existing SCD techniques for the video database system is presented and discussed. Criteria of benchmarking SCD algorithms are also proposed. Existing SCD algorithms have achieved above 90% success rate for abrupt scene changes, and above 80% success rate for gradual scene changes. These numbers are, in general, fairly acceptable in certain applications. However, there is an obvious need for further improvement.

There are several possible ways to achieve this:

1. Use additional visual as well as audio information, rather than relying only on the color or intensity information most existing algorithm rely on. Other visual information includes such as captions, motion of the objects and camera and object shapes. The problem of how to use audio signals and other information contained in the video data in the scene change detection and video segmentation has not been carefully addressed in the literature so far, although some initial efforts [25, 26] have been made for video skimming and broswing support.

2. Develop adaptive SCD algorithms which can combine several SCD techniques and can self–adjust to various parameters. They would choose best criteria optimized for a different given video data, i. e. a video sequence with frequent object movements (action movies) v. s. the one with very few motions (lecture video).

3. Use a combination of various scene change models. Developing scene change models can be a very difficult task due to the complicated nature of video production and editinge. However, different aspects of the video editing and production process can be individually modeled and used in developing detectors for certain scene changes.

Another idea is to develop new video coding and decoding schemes that include more information about scene content. As pointed out by Bove [3], current motion–compensated video codec standards like MPEG, complicate the scene analysis task by partitioning the scene into arbitrary tiles, resulting in a compressed bitstream which is not physically or semantically related to the scene structure. For a complete solution to the problem, however, a better understanding of the human capabilities and techniques for SCD is needed. This would involve using information available from psychopysics [5, 6, 14, 22], and also understanding the neural circuitry of the visual pathway [16]. Techniques developed in computer vision for detecting motion or objects [4, 29, 7, 5] can also be incorprated into SCD algorithms.

References

[1] P. Aigrain and P. Joly, *Automatic Real-time Analysis of Film Editing and Transition Effects and its Applications*, Computer and Graphics **18** (1994), no. 1, 93–103.

[2] Hideo Hashimoto Akihito Akutsu, Yoshinobu Tonomura and Yuji Ohba, *Video Indexing Using Motion Vectors*, Proceedings of SPIE: Visual Communications and Image Processing 92, November 1992.

[3] V. M. Bove, *What's Wrong with Today's Video Coding*, TV Technology (1995).

[4] Claudette Cedras and Mubarak Shah, *Motion-based Recognition: A Survey*, Image and Vision Computing **13** (1995), no. 2, 129–155.

[5] R. Chellappa, Charles L. Wilson, and Saad Sirohey, *Human and Machine Recognition of Faces: A Survey*, Proceedings of the IEEE **83** (1995), no. 5, 705–740.

[6] M.R.W. Dawson, *The How and Why of What Went Where in Apparent Motion*, Psychological Review **98** (1991), 569–603.

[7] Farshid Arman, Arding Hsu and Ming-Yee Chiu, *Image Processing on Compressed Data for Large Video Database*, Proceedings of the ACM Multimedia (California, USA), June 1993, Association of Computing Machinery, pp. 267–272.

[8] International Organization for Standardization, *ISO/IEC 11172 (MPEG)*.

[9] D. Le. Gall, *MPEG: A Video Compression Standard for Multimedia Applications*, Communication of ACM **34** (1991), no. 4, 46–58.

[10] Arun Hampapur, *Design Video Data Management Systems*, Ph.D. thesis, The University of Michigan, 1995.

[11] Arun Hampapur, Ramesh Jain, and Terry Weymouth, *Digital Video Indexing in Multimedia Systems*, Proceedings of the Workshop on Indexing and Reuse in Multimedia Systems, August 1994.

[12] ———, *Digital Video Segmentation*, Proceedings Second Annual ACM Multimedia Conference and Exposition, 1994.

[13] P. R. Hsu and H. Harashima, *Detecting Scene Changes and Activities in Video Databases*, ICASSP'94, vol. 5, April 1994, pp. 33–36.

[14] A. Joshi, *On Connectionism and the Problem of Correspondence in Computer Vision*, Ph.D. thesis, Department of Computer Science, Purdue University, 1993.

[15] H. C. Liu and G. L. Zick, *Scene Decomposition of MPEG Compressed Video*, Digital Video Compression: Algorithms and Technologies, vol. SPIE 2419, February 1995.

[16] M. Livingstone and D. O. Hubel, *Segregation of Form, Color, Movement and Depth: Anatomy, Physiology and Perception*, Science **240** (1988), 740–749.

[17] H. G. Longbotham and A. C. Bovic, *Theory of Order Statistic Filters and Their Relationship to Linear FIR Filters*, IEEE Transactions on Acoustics, Speech, and Signal Processing **ASSP-37** (1989), no. 2, 275–287.

[18] J. Meng, Y. Juan, and S. F. Chang, *Scene Change Detection in a MPEG Compressed Video Sequence*, Storage and Retrieval for Image and Video Database III, vol. SPIE 2420, February 1995.

[19] A. Nagasaka and Y. Tanaka, *Automatic Video Indexing and Full-video Search for Object Appearances*, Second Working Conference on Visual Database Systems (Budapest, Hungary), IFIP WG 2.6, October 1991, pp. 119–133.

[20] K. Otsuji and Y. Tonomura, *Projection Detecting Filter for Video Cut Detection*, Proceedings of First ACM International Conference on Multimedia, August 1993.

[21] K. Otsuji, Y. Tonomura, and Y. Ohba, *Video Browsing Using Brightness Data*, Visual Communications and Image Processing, vol. SPIE 1606, 1991, pp. 980–989.

[22] Z. Pizlo, A. Rosenfeld, and J. Epelboim, *An Exponential Pyramid Model of the Time Course of Size Processing*, Vision Research **35** (1995), 1089–1107.

[23] I. K. Sethi and N. Patel, *A Statistical Approach to Scene Change Detection*, Storage and Retrieval for Image and Video Database III, vol. SPIE-2420, February 1995, pp. 329–338.

[24] B. Shahraray, *Scene Change Detection and Content-Based Sampling of Video Sequences*, Digital Video Compression: Algorithms and Technologies, vol. SPIE 2419, February 1995, pp. 2–13.

[25] Michael A. Smith and Michael G. Christel, *Automating the Creation of a Digital Video Library*, ACM Multimedia (1995).

[26] Michael A. Smith and Alexander Hauptmann, *Text, Speech, and Vision for Video Segmentation: The Informedia Project*, AAAI Fall 1995 Symposium on Computational Models for Integrating Language and Vision, 1995.

[27] M. J. Swain and D. H. Ballard, *Color Indexing*, International Journal of Computer Vision (1991), 11–32.

[28] Deborah Swanberg, Chiao-Fe Shu, and Ramesh Jain, *Knowledge Guided Parsing in Video Database*, Electronic Imaging: Science and Technology, San Jose, California, February 1993, IST/SPIE.

[29] Mruthyunjaya S. Telagi and Athamaram H. Soni, *3-D object recognition techniques: A Survey*, Proceedings of the 1994 ASME Design Technical Conferences, vol. 73, September 1994.

[30] Yoshinobu Tonomura, Akihito Akutsu, Yukinobu Taniguchi, and Gen Suzuk, *Structured Video Computing*, IEEE Multimedia 1 (1994), no. 3, 34–43.

[31] Boon-Lock Yeo, *Efficient Processing of Compressed Images and Video*, Ph.D. thesis, Princeton University, January 1996.

[32] Boon-Lock Yeo and Bede Liu, *A Unified Approach to Temporal Segmentation of Motion JPEG and MPEG Compressed Video*, Second International Conference on Multimedia Computing and Systems, May 1995.

[33] ———, *On The Extraction of DC Sequence From MPEG Compressed Video*, The International Conference on Image Processing, October 1995.

[34] ———, *Rapid Scene Analysis and Compressed Video*, IEEE Transactions on Circuit and System For Video Technology **5** (1995), no. 6, 533–544.

[35] Minerva M. Yeung and Bede Liu, *Efficient Matching and Clustering of Video Shots*, International Conference on Image Processing, vol. I, October 1995, pp. 338–343.

[36] Ramin Zabih, Justin Miller, and Kevin Mai, *Feature-Based Algorithms for Detecting and Classifying Scene Breaks*, Fourth ACM Conference on Multimedia (San Francisco, California), November 1995.

[37] H. J. Zhang, A. Kankanhalli, and S. W. Smoliar, *Automatic Partition of Animate Video*, Tech. report, Institute of System Science, National University of Singapore, 1992.

[38] ———, *Automatic Parsing of Full-Motion Video*, Multimedia Systems **1** (1993), 10–28.

[39] H. J. Zhang, C. Y. Low, Y. Gong, and S. W. Smoliar, *Video Parsing Using Compressed Data*, Image and Video Processing II, vol. SPIE 2182, 1994, pp. 142–149.