CONTENT-BASED VIDEO COPY DETECTION USING MULTIMODAL ANALYSIS

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Onur Küçüktunç

July, 2009

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Özgür Ulusoy (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Uğur Güdükbay (Co-supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. A. Enis Çetin

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. İbrahim Körpeoğlu

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Selim Aksoy

Approved for the Institute of Engineering and Science:

Prof. Dr. Mehmet B. Baray
Director of the Institute

ABSTRACT

CONTENT-BASED VIDEO COPY DETECTION USING MULTIMODAL ANALYSIS

Onur Küçüktunç
M.S. in Computer Engineering
Supervisors: Prof. Dr. Özgür Ulusoy and Assoc. Prof. Dr. Uğur Güdükbay
July, 2009

Huge and increasing amount of videos broadcast through networks has raised the need of automatic video copy detection for copyright protection. Recent developments in multimedia technology introduced content-based copy detection (CBCD) as a new research field alternative to the watermarking approach for identification of video sequences.

This thesis presents a multimodal framework for matching video sequences using a three-step approach: First, a high-level face detector identifies facial frames/shots in a video clip. Matching faces with extended body regions gives the flexibility to discriminate the same person (e.g., an anchor man or a political leader) in different events or scenes. In the second step, a spatiotemporal sequence matching technique is employed to match video clips/segments that are similar in terms of activity. Finally the non-facial shots are matched using low-level visual features. In addition, we utilize fuzzy logic approach for extracting color histogram to detect shot boundaries of heavily manipulated video clips. Methods for detecting noise, frame-droppings, picture-in-picture transformation windows, and extracting mask for still regions are also proposed and evaluated.

The proposed method was tested on the query and reference dataset of CBCD task of TRECVID 2008. Our results were compared with the results of top-8 most successful techniques submitted to this task. Experimental results show that the proposed method performs better than most of the state-of-the-art techniques, in terms of both effectiveness and efficiency.

Keywords: copy detection, video processing, shot-boundary detection, video segmentation, subsequence matching, face detection.

ÖZET

ÇOK KİPLİ ANALİZ İLE İÇERİK TABANLI VİDEO KOPYA SEZİMİ

Onur Küçüktunç
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticileri: Prof. Dr. Özgür Ulusoy ve Doç. Dr. Uğur Güdükbay
Temmuz, 2009

Yüksek ve artan oranlarda videonun çeşitli ağlarda serbestçe yayımlanması, telif haklarının korunması için otomatik video kopya sezimi ihtiyacını beraberinde getirmiştir. Çoklu ortam teknolojilerindeki gelişmeler, filigram yaklaşımına alternatif olarak içerik tabanlı video kopya sezimi yöntemini ortaya sürmüştür.

Bu tez çalışmasında video kliplerini eşleştirmeyi sağlayan çok kipli bir sistem önerilmektedir. İlk olarak bir yüz detektörü kullanılarak yüz içeren video bölümleri belirlenir. Yüz ve bedenin yüze yakın belirli bir kısmını eşleştirmek, aynı kişiyi (örneğin, sunucu veya politik lider) farklı olaylarda veya sahnelerde ayırma esnekliği sağlar. İkinci olarak, harekete bağlı benzerliği olan video bölümlerini eşleştirmek için uzaysal ve zamansal dizileri eşleyen bir teknik kullanılır. Son olarak, yüz içermeyen video bölümleri düşük seviyeli görsel öznitelikler ile eşleştirilmektedir. Bunlara ek olarak, videoları bölütlemek için kullanılan renk diklemlerinde bulanık mantıktan yararlanılmaktadır. Gürültüleri, silinen film karelerini, iç içe geçmiş çerçeveleri tespit etmek ve durağan bölgeler için maske oluşturmak için de yöntemler önerilmiştir.

Tanıtılan sistem, TRECVID 2008 yarışmasında İçerik Tabanlı Kopya Sezimi görevi için hazırlanan sorgu ve referans videoları üzerinde test edilmiş, sonuçlarımız bu göreve katılan en iyi 8 teknikle karşılaştırılmıştır. Bu deneylerde sistemimizin diğer birçok modern teknikten daha verimli ve etkili çalıştığı gösterilmiştir.

Anahtar sözcükler: kopya sezimi, video işleme, video bölütleme, altdizi eşleştirme, yüz tanıma.

Acknowledgement

I would like to express my sincere gratitude to my supervisors Prof. Dr. Özgür Ulusoy and Assoc. Prof. Dr. Uğur Güdükbay for their instructive comments, suggestions, support and encouragement during this thesis work.

I am grateful to my jury members, Prof. Dr. A. Enis Çetin, Asst. Prof. Dr. İbrahim Körpeoğlu, and Asst. Prof. Dr. Selim Aksoy for reading and reviewing this thesis.

Special thanks to my family for their support during my education and their never lasting efforts on me.

I would also like to thank to my friends Daniya Zamalieva, Sare Gül Sevil, and Fırat Kalaycılar for their caring friendship and motivation. Especially, thanks to Muhammet Başţan for his support and collaboration during all stages of this study.

Finally, I would like to thank to TÜBİTAK-BİDEB for their financial support during my graduate education.

Sound and Vision video is copyrighted. The Sound and Vision video used in this work is provided solely for research purposes through the TREC Video Information Retrieval Evaluation Project Collection.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

A recent study released by IDC [6] found that the total volume of digital content added to digital universe was 487 billion gigabytes in 2008, meaning that there are more than 90 GBs of data per person on Earth. This number was 281 billion gigabytes in 2007 [11].

In this huge digital universe, each industry has its own share: manufacturing and transportation industry is rapidly deploying digital surveillance cameras, retail industry stores customer activities, governments process and store PBs of satellite images, healthcare sector depends on medical imaging databases and records, and so on. Among these industries, broadcasting, media, and entertainment industry already generates 50% of the digital data today. When all TV channels broadcast digitally and all the movies become digital within the next 10 years, the share of communication/media/entertainment industries will be higher than today's.

YouTube [14], which is the #1 largest video sharing site, contributes most of the video streaming, sharing, and storage on the Internet today. Chad Hurley (CEO and co-founder of YouTube) claims that online video broadcasting will be

the most accessible form of communication in the next decade [12]. In September 2008, 13 hours of video were being uploaded to YouTube every minute, and as Hudley states, the volume will continue to grow exponentially. Today, 20 hours of video are being uploaded to YouTube every single minute [16].

Even for the world's lead video sharing website, detection of copyrighted materials is one of the biggest issues. Copyright holders constantly issue takedown notices for the unauthorized clips of TV shows, movies and music videos. After some organizations have issued lawsuits against YouTube, the website has introduced a system called Video ID to copyright holders to check uploaded videos against a database of copyrighted content for reducing violations [2, 15]. However, the solution is not as simple as detecting watermarks; therefore, the problem of video copy detection still remains an open-research area.

For identification of copyrighted materials, content-based copy detection (CBCD) was introduced as a new research field alternative to the watermarking approach. In addition to copyright protection issues, there are other applications of video copy detection. For instance, it allows the tracking of news stories across different sources [81, 39], measuring the novelty [75], tracking of known or repeated sequences [24], and identification of commercials [31]. Video copy detection techniques also enhance the indexing, searching, and retrieval capabilities of a multimedia database.

In this thesis, our aim is to propose a complete content-based video copy detection framework that uses multimodal analysis based on facial detection, spatio-temporal activity matching, and low-level visual feature similarity.

1.2 Challenges

Video copy detection is a challenging problem in computer vision due to some reasons. First of all, the problem domain is exceptionally wide. Depending on the purpose of a video copy detection system, different solutions can be applied. For example, a simple frame-based color histogram similarity approach could be

enough for detecting exact duplicates of video segments or identifying commercial breaks. On the other hand, matching news stories across different channels (camera viewpoints) is a totally different problem, and will probably require interest point-matching techniques. Therefore, no general solution can be proposed to video copy detection problem.

Secondly, the problem space is extremely large, which often requires real-time solutions. For the case of YouTube, the system needs to process *20 hours of video content per second* to find an exact or near-duplicate segment of a copyrighted material. Suppose that Warner Bros. Entertainment, Inc. wants to find unauthenticated clips of their movies on YouTube. Warner Bros. owns more than 593 movies that are produced, co-produced, and/or distributed by the production company [8]. If we take a movie about 2 hours, we can find that we need to compare *20 hours of video with ∼1200 hours of copyrighted movies per second.*

More importantly, a video copy detection system is expected to mimic human vision, understanding, and logic. However, the techniques, such as face detection, recognition, visual feature extraction, dimensionality reduction, similarity calculation, etc. are still being developed today. This is, in fact, a limitation of all copy detection and image/video retrieval systems.

1.2.1 TRECVID Copy Detection task

TREC Video Retrieval Evaluation (TRECVID) [71] is an organization focusing on content-based analsis of video. Every year, participants of TRECVID test their systems on spesific tasks, including automatic segmentation, indexing, and content-based retrieval of video.

Beginning in 2008, TRECVID introduced content-based copy detection (CBCD) as a new task to evaluate. Aim of the task is to determine the place of each query video in the test collection accompanied with a decision score. A copy is defined as a segment of video derived from another video, usually by means of various transformations such as addition, deletion, modification (of aspect, color,

contrast, encoding), camcording, etc.; so the queries are constructed according to this definition.

1.2.2 Video Transformations

Each query video in TRECVID CBCD task is constructed by taking a segment from the test collection, transforming and/or embedding it into some other video segment, and finally applying one or more transformations to the entire query segment [5]. Since the query set prepared for CBCD task is used for evaluation purposes, we focus on the transformations in Table 1.1 [1]. These transformations cover most of the video modifications in daily life (cf. Figure 1.1).

Table 1.1: The list of transformations used in the CBCD task.

#	**Transformation details**
T1	Camcording
T2	Picture-in-picture Type 1
T3	Insertion of patterns (15 different patterns)
T4	Strong re-encoding (change of resolution, bitrate)
T5	Change of gamma
T6	Combination of 3 transformations amongst: blur, gamma, frame dropping, contrast, compression, ratio, noise (A)
T7	Combination of 5 transformations amongst (A)
T8	Combination of 3 transformations amongst: crop, shift, contrast, caption, flip, insertion of pattern, picture-in-picture Type 2 (original video is behind) (B)
T9	Combination of 5 transformations amongst (B)
T10	Combination of 5 transformations amongst all the transformations from 1 to 9

1.3 Summary of Contributions

We propose a complete content-based video copy detection framework that uses multimodal analysis based on facial shot detection, spatio-temporal activity matching, and low-level visual feature similarity. The overview of the CBCD system is shown in Figure 1.2.

(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 1.1: Transformations: (a) original frame, (b) picture-in-picture type 1, (c) insertion of pattern, (d) strong re-encoding, (e) change of gamma, (f) letter-box, (g) white noise, (h) crop, (i) shift, (j) caption/text insertion, (k) flip, and (l) picture-in-picture type 2.

The contributions of this thesis can be summarized as folows:

- Fuzzy color histogram method that is robust to illumination changes;
- Fuzzy color histogram based shot-boundary detection method for the videos where heavy transformations (such as cam-cording, insertions of patterns, strong re-encoding) occur;
- Mask extraction, picture-in-picture window detection, and noise detection methods for content-based copy detection systems;
- An effective facial shot detection and matching method for detecting copy shots;
- A variable-weighted visual similarity calculation technique; and

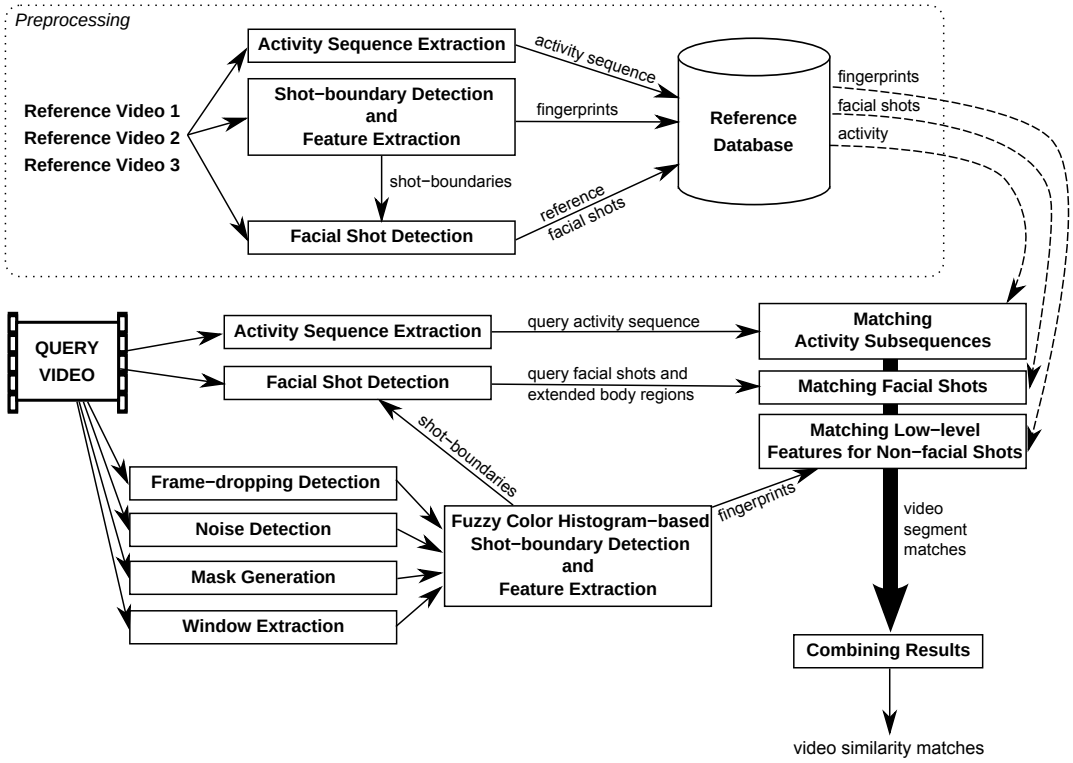


Figure 1.2: Overview of our CBCD system.

- Subsequence matching of activity time-series that is robust to many video manipulations, such as flip, picture-in-picture, re-encoding, and so on.

1.4 Outline of the Thesis

The thesis is organized as follows

- Chapter 2 describes the video segmentation of query and reference videos along with the detection of various transformations; i.e., frame-dropping, noise addition, text/logo insertion, and picture-in-picture transformation.
- Chapter 3 presents our CBCD framework with three steps multimodal analysis: first, facial shots are identified and matched; in the second step, similarities between activities of the query and reference video sequences are

detected; and finally, similar non-facial shots are matched with low-level MPEG-7 visual descriptors.

- Chapter 4 includes the evaluation of each proposed method, and the comparison of our CBCD framework with the state-of-the-art methods used in TRECVID'08 CBCD task.

- Chapter 5 gives the conclusions and future research directions.

Chapter 2

Video Segmentation

The detection of shots, as in many video indexing and retrieval applications, is the first step of video analysis. Video is segmented (or frames are merged) into temporal pieces as a result of shot-boundary detection (see Figure 2.1).

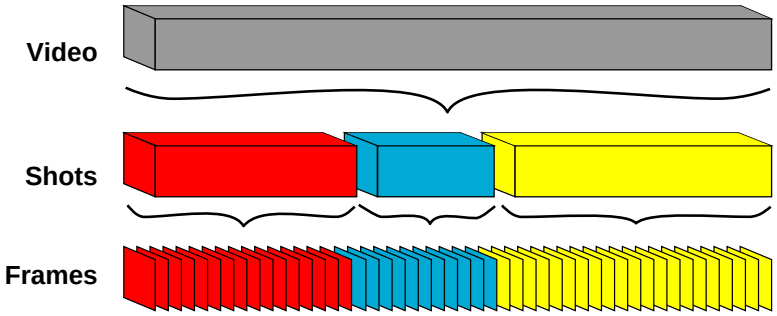


Figure 2.1: Temporal segmentation of a video.

A shot is defined as a series of related consecutive frames representing a continuous action in time and space taken by a single camera [21]. A video is composed of several shots combined with abrupt or gradual transitions (see Figure 2.2). An abrupt transition, also known as hard-cut, is the most common and easy to detect transition type. On the other hand, gradual transitions (fades, dissolves and wipes) are spread over a number of frames, thus they are harder to detect.

Various shot-boundary detection algorithms have been proposed [21, 40, 35,

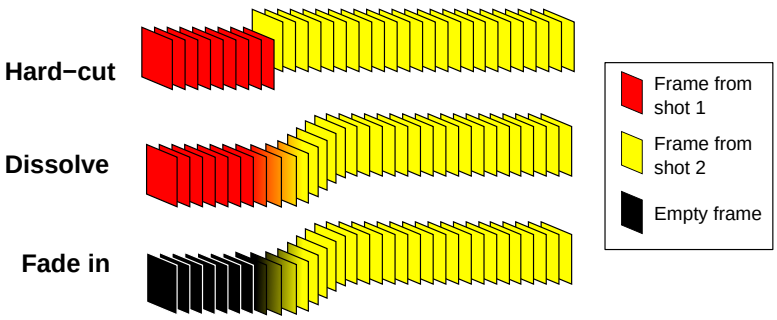


Figure 2.2: Types of transitions between shots.

23, 79, 72] and compared [26, 58, 27, 22]; however, to the best of our knowledge, no shot-boundary detection algorithm specialized for CBCD is found in the literature. Our aim is to propose an automatic shot-boundary detection algorithm for the videos on which various transformations are applied. In contrast to most of the existing methods, we utilize fuzzy logic approach for extracting color histogram to detect shot boundaries.

2.1 Related Work

2.1.1 Shot-boundary Detection

Studies on shot-boundary detection are typically based on extracting visual features (color, edge, motion, and interest points) and comparing them among successive frames. Truong *et al.* [72] propose techniques for cut, fade, and dissolve detections. An adaptive thresholding technique to detect peaks in the color histogram difference curve is presented for detecting hard cuts. Locating monochrome frames and considering luminance mean and variance are the steps for fade and dissolve detection. Danisman and Alpkocak [28] apply a method based on color histogram differences in RGB color space and thresholding for cut detection. They present skip frame interval technique, which reduces the computation time with a slight decrease in the precision. Dailianas [27], Boreczky and Rowe [22] compare early shot-boundary detection algorithms. Lienhart [58] extends this comparison by taking newer algorithms into account, and by

measuring their ability to detect the type and temporal extent of the transitions. Cotsaces *et al.* [26] give an up-to-date review.

In recent years, researchers focus on detecting gradual transitions effectively and avoiding the false alarms caused by flashlight and the motion of large objects in the scene, since the recognition of hard-cuts is very reliable for most of the methods. Huang *et al.* [40] propose an approach based on local keypoint matching of video frames to detect abrupt and gradual transitions. By matching the same objects and scenes using contrast context histogram (CCH) in two adjacent frames, the method decides that there is no shot change. Corana *et al.* [35] propose a two-step iterative algorithm, unique for both cuts and gradual transitions detection, in the presence of fast object motion and camera operations. Boccignone *et al.* [21] use a consistency measure of the fixation sequences generated by an ideal observer looking at the video for determining shot changes. A scene-break detection approach based on linear prediction model is proposed in [23]. Shot-boundaries are detected using Bayesian cost functions, by comparing original frame with the predicted frame, estimated using within video shot linear prediction model (WLPM) and dissolve linear prediction model (DLPM). Yuan *et al.* present a unified shot boundary detection system based on graph partitioning model [79]. The representation of the visual content, the construction of the continuity signal, and the classification of continuity values are handled in this work. The evaluations show that the SVM-based active learning outperforms both thresholding and nonactive learning.

2.1.2 Fuzzy Logic and Systems

Fuzzy logic introduced by Zadeh [80] is being used in many applications related to image processing. Konstantinidis *et al.* [51] and Han and Ma [38] utilize fuzzy logic for creating color histograms to be used in content-based image retrieval systems. Chung and Fung [25] introduce fuzzy color quantization to color histogram construction, and evaluate its performance in video scene detection with a very limited video dataset. Fang *et al.* [33] propose a fuzzy logic approach for temporal segmentation of videos, where color histogram intersection, motion

compensation, texture change and edge variances are integrated for cut detection. In [41], histogram differences of consecutive frames are characterized as fuzzy terms, such as small, significant and large, and fuzzy rules for detecting abrupt and gradual transitions are formulated in a fuzzy-logic-based framework for segmentation of video sequences. Das *et al.* [29] define a unified interval type-2 fuzzy rule based model using fuzzy histogram and fuzzy co-occurrence matrix to detect cuts and various types of gradual transitions.

2.1.3 Methods used by Copy Detection Systems

In the field of CBCD, representing video with a set of keyframes (one or more representative frame for each shot) is a common approach. Some of the recent studies on CBCD task of TRECVID 2008 employ the following techniques. Llorente *et al.* [60] use an approach based on color histogram and thresholding, extended by [68] for detection of gradual transitions. Douze *et al.* prefer extracting 2.5 frames per second for query videos, and extracting only a few representative keyframes for the dataset [30]. We also preferred extracting a fixed number of frames per time interval in our earlier CBCD system [52]. Studies in video copy detection domain, therefore, do not necessarily use a shot-boundary detection method.

2.2 Effects of Video Transformations on Shot-boundary Detection

The negative effects of video transformations, and possible corrective actions taken by our method are discussed in this section:

1. *Frame dropping:* Dropped frames should be ignored or estimated; otherwise the shot-boundary detection algorithm decides each blank frame as a cut. Such frames have the mean of intensity values near to zero.

2. *Picture-in-picture:* Regardless of which type is applied to the video segment, detecting the window of picture-in-picture transformation (boundaries of the inner video) is crucial for feature extraction step of the CBCD system [66]. With the extracted window, foreground and background frames can be handled separately.

3. *Insertion of patterns, caption:* Although the insertion of a pattern or text does not affect the shot-boundary detection process strongly, a mask for still regions, which includes the inserted pattern or text, will increase the effectiveness of a CBCD system. Unmasked patterns and captions introduce new edges and regions of interests, and cause changes on color information.

4. *Camcording, crop, shift:* These transformations generally produce black framings on one or more sides of the video segment. Since the framings are also still regions, we can ignore these areas during the feature extraction.

5. *Strong re-encoding, blur, change of gamma, contrast, compression:* It is important to use a keypoint detector invariant to these changes. These changes have nearly no effect on shot-boundary detection because they are applied on the whole video with the same parameter values.

6. *Noise:* Since the detection of windows for picture-in-picture transformation depends on edge detection, noisy shots should be discovered and handled before further processing.

2.3 Shot-boundary Detection for Reference Videos

Detecting shot-boundaries of reference videos is important, since we need to extract visual features from reference keyframes in video copy detection part. Compared to shot-boundary detection of query videos, the video segmentation problem here is simpler (because the videos are not altered with heavy transformations), and yet can be solved with an easier approach.

We used a conventional color quantization method in RGB color space, and Chi-Square distance as the histogram comparison method:

$$D_{SqChi}(H_1, H_2) = \sum_{i=1}^{b} \frac{(H_1(i) - H_2(i))^2}{H_1(i) + H_2(i)} \tag{2.1}$$

Shot-boundary detection of reference videos was implemented using OpenCV [10], and the system works with ∼100fps rate.

2.4 Method for Query Videos

We present a fuzzy color histogram-based shot-boundary detection algorithm specialized for content-based copy detection applications. The proposed method aims to detect both cuts and gradual transitions (fade, dissolve) effectively in videos where heavy transformations (such as cam-cording, insertions of patterns, strong re-encoding) occur. Along with the color histogram generated with the fuzzy linking method on $L^*a^*b^*$ color space, the system extracts a mask for still regions and the window of picture-in-picture transformation for each detected shot, which will be useful in a content-based copy detection system. Experimental results show that our method effectively detects shot boundaries and reduces false alarms as compared to the state-of-the-art shot-boundary detection algorithms.

The parameters of the system are given in Table 2.1, and the overview of the proposed algorithm is presented in Figure 2.3.

2.4.1 Detection of Frame-dropping Transformation

Handling frame-dropping transformation is one of the key features of a shot-boundary detection system specialized for CBCD applications; since most of the proposed algorithms consider missing frames as hard-cuts. A dropped frame is either exactly or nearly a blank frame, which has a small overall intensity (less than $th_{bf} = 0.0039$). We define a binary function fd for a given video frame I_n

Table 2.1: The parameters of the algorithm.

Parameters	Description
θ_c	Threshold for cut detection
θ_g	Threshold for gradual transition detection
τ	Timescale for central moving average filter
th_{bf}	Intensity threshold for blank frame detection
th_n	Average intensity-change threshold for noisy image
th_{sr}	Threshold for still regions
s_{mf}	Size of median filter used in noise detection

as:

$$fd(I_n) = \begin{cases} 1 & \sum_{i=1}^{h}\sum_{j=1}^{w} G_n(i,j) < th_{bf} \\ 0 & otherwise \end{cases} \tag{2.2}$$

where G_n is the grayscale intensity image of I_n, and (h, w) is the dimension of the frame.

2.4.2 Noise Detection

CBCD applications should handle query videos with heavy noise transformations. For our algorithms to work properly, noisy frames/shots should be identified before any further operation that is based on edge detection or use standard deviation of pixel intensity values.

In image processing, a nonlinear median filter is preferred over a linear filter for cleaning salt & pepper and white noise. Based on this fact, we calculate the average intensity change of an image I_n after a median filter of size $s_{mf} \times s_{mf}$ is applied to the image. If the image slightly changes after the median filter, we assume that less noise exists in the image. Otherwise, when the average intensity change exceeds a threshold th_n, it is regarded as noisy.

$$nf(I_n) = \begin{cases} 1 & \frac{1}{h \times w}\sum_{i=1}^{h}\sum_{j=1}^{w} |G_n(i,j) - M_n(i,j)| > th_n \\ 0 & otherwise \end{cases} \tag{2.3}$$

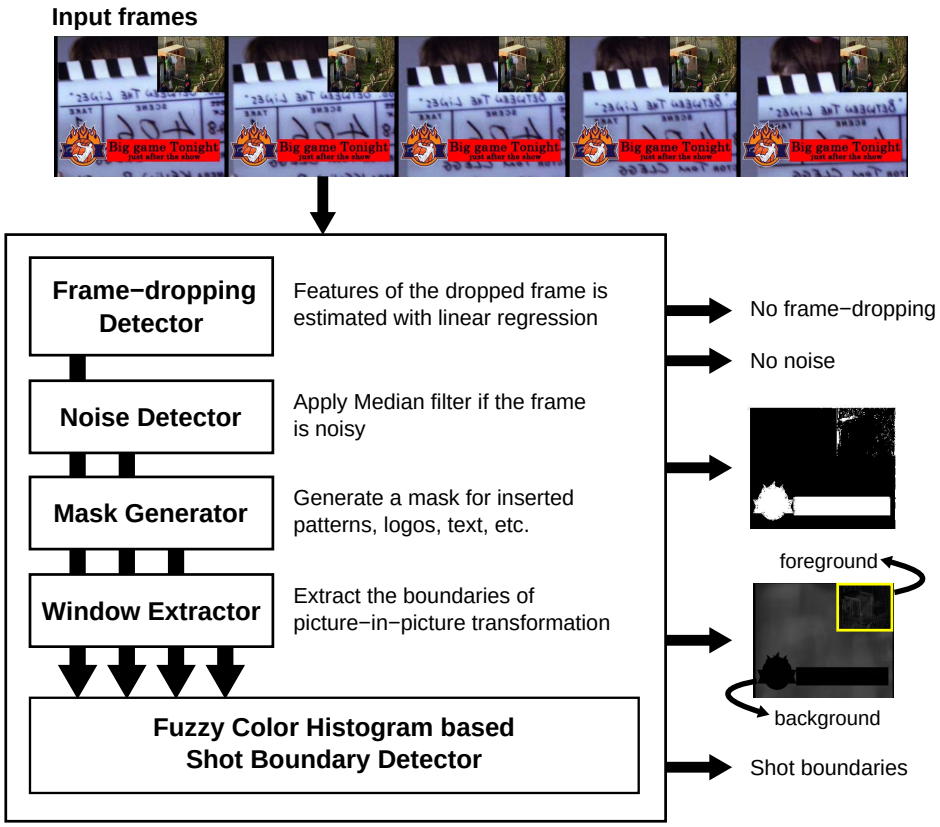


Figure 2.3: The overview of the proposed algorithm.

We evaluate the noise detection method and the impact of the parameters (the size of the median filter and the threshold value) in Section 4.2.

2.4.3 Mask Generation

When a video segment is transformed with various types of transformations summarized in Table 1.1, it clearly changes the content of the frames regarding color, edge, and shape information. A content-based copy detection system should cut out the artificially inserted texts, patterns, logos, etc., if possible. Besides, it should ignore the bordering black areas produced by shift, crop, and letterbox transformations. As a result, the probability of matching with the original video segment is increased. See Figure 2.4 for a sample mask, and how it affects the color histogram.

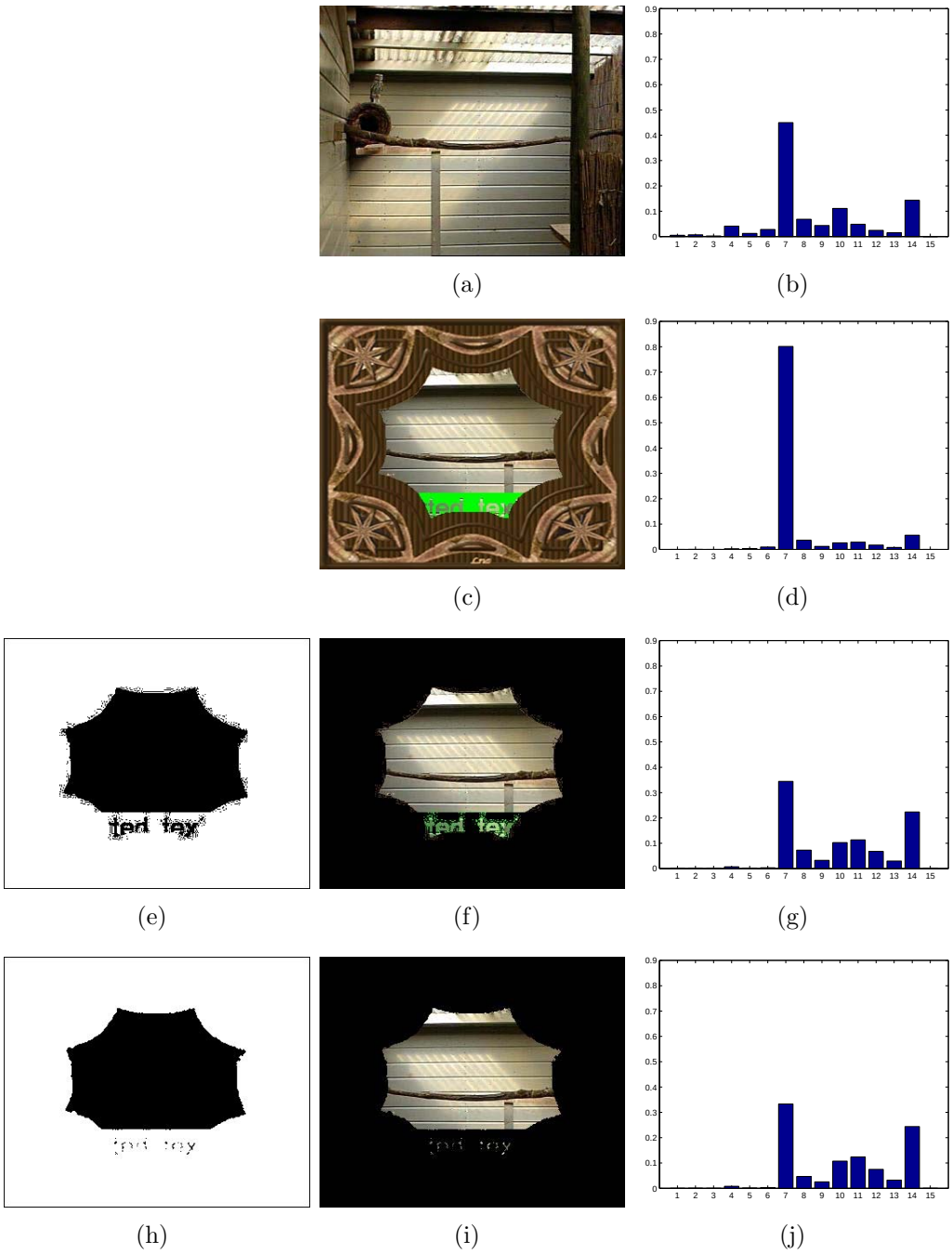


Figure 2.4: Mask generation and its effect on color histogram. (a) original frame, (b) color histogram of the frame, (c) query frame, (d) its histogram, (e) mask of still regions for this query shot, (f) masked query frame, (g) its histogram, (h) mask dilated with a disk structuring element of size 3x3, (i) masked query image, and (j) its histogram. Euclidean distance between histograms are: $D(H_b, H_d) = 0.142$, $D(H_b, H_g) = 0.026$, and $D(H_b, H_j) = 0.035$.

We create a mask of standard deviations greater than the threshold $th_{sr} = 0.01$ for each shot representing still regions while detecting shot boundaries, assuming that a pixel intensity varies from 0 to 1:

$$M_{shot}(i,j) = \begin{cases} 1 & \sigma_{shot}(i,j) > th_{sr} \\ 0 & otherwise \end{cases} \tag{2.4}$$

The mean and standard deviation of the pixel intensity values within a video shot of N frames are calculated with the Equations 2.5 and 2.6, respectively:

$$\mu_{shot}(i,j) = \frac{1}{N}\sum_{k=1}^{N} G_k(i,j) \tag{2.5}$$

$$\sigma_{shot}(i,j) = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(G_k(i,j) - \mu_{shot}(i,j))^2} \tag{2.6}$$

The problem here is that today's computers have a limitation that can hold up to a number of frames together in memory. Therefore, we employ the solution for incremental standard deviation calculation discussed by Donald Knuth [50], who cites Welford [74]:

$$\mu_k(i,j) = \mu_{k-1}(i,j) + \frac{G_k(i,j) - \mu_{k-1}(i,j)}{k} \tag{2.7}$$

$$s_k = s_{k-1} + (G_k - \mu_{k-1}) \times (G_k - \mu_k) \tag{2.8}$$

$$\sigma_k(i,j) = \sqrt{s_k(i,j)/(k-1)} \tag{2.9}$$

where $\mu_1(i,j) = G_1(i,j)$ and $s_1(i,j) = 0$ initially. For a shot with n frames, we save the mask $M_{shot} = M_n$ and the standard deviation of the shot $\sigma_{shot} = \sigma_n$ for further use.

2.4.4 Detection of Picture-in-Picture Transformation

In order to detect the window of picture-in-picture transformation, black framings on the sides of the video segment generated by camcording, crop, or shift

transformations should be extracted first. We mark each row and column starting from the beginning and from the end as border rows if

$$\frac{1}{w}\sum_{c=1}^{w}\sigma_{shot}(i,c) < th_{sr} \tag{2.10}$$

holds for that row. Similarly, blank columns from the beginning and from the end are identified. If Equation 2.10 returns false for a row/column, we stop marking borderlines for that edge. Figure 2.5 shows an example to border detection.

(a) (b) (c)

Figure 2.5: The detection of borders: (a) first frame of a query video shot on which both the picture-in-picture and crop transformations are applied, (b) the standard deviation of the shot, and (c) the border shown in red.

The next step is to detect the vertical lines. We crop out the borders from M_{shot}, and then find the derivatives with a first order difference from both + and − x-axis using the Prewitt edge detector:

$$E_{shot} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} * M_{shot} \tag{2.11}$$

Strong vertical edges are extracted from E_{shot} using Hough lines [32]. Only vertical lines are selected, and compared in order to form a rectangular window. The candidate window(s) and the border information for each shot are stored. Figure 2.6 displays examples of frames whose borders and windows are successfully detected.

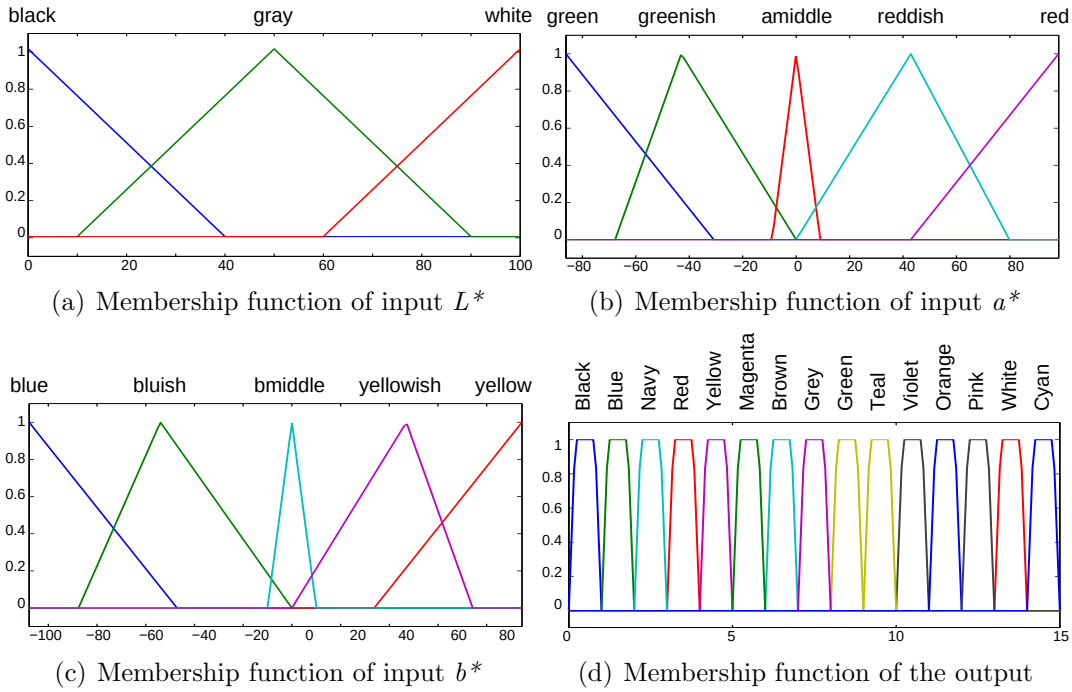
Figure 2.6: Detected borders and windows for query frames. The borders are shown in red and the window frames are shown as green rectangles.

2.4.5 Fuzzy Color Histogram based Shot-boundary Detection

We use a color histogram-based method generated with the fuzzy linking method on $L^*a^*b^*$ color space. A brief discussion on why $L^*a^*b^*$ color space is preferred, how the dimensions are subdivided into regions, their ranges, and the results of an experiment with popular colors, are provided in Section 2.5.

Fuzzification of the inputs is achieved by using triangular membership functions for each component. L^* is divided into 3 regions (white, grey, black), a^* is divided into 5 regions (green, greenish, middle, reddish, red), and b^* also is divided into 5 regions (blue, bluish, middle, yellowish, yellow). This is the approach used in [51]; however, we extracted new colors and found corresponding fuzzy rules. Membership f unctions of the inputs and the output are shown in Figure 2.7.

In conventional color histograms, each pixel belongs to only one histogram bin, depending on whether the pixel is quantized into the bin or not. The conditional probability $P_{i|j}$ of the selected jth pixel belonging to the ith color bin is defined



(a) Membership function of input L^*

(b) Membership function of input a^*

(c) Membership function of input b^*

(d) Membership function of the output

Figure 2.7: Fuzzy membership functions for the inputs (L^*, a^*, b^*) and output.

as a binary equation:

$$P_{i|j} = \begin{cases} 1 & \text{if the } j\text{th pixel is quantized into the } i\text{th histogram bin} \\ 0 & otherwise \end{cases} \tag{2.12}$$

If $L^*a^*b^*$ color space was partitioned into 3x5x5 (for L^*, a^*, and b^*, respectively) subspaces in a conventional manner, this definition would lead to serious boundary issues and problems related to the partition size. However, in the context of fuzzy color histogram, the degree of association μ_{ij} of jth pixel to ith bin is calculated with fuzzy membership functions (see Figure 2.7). L^* component of a pixel might have both a degree of *gray* and *white* together, for instance. Therefore, the color of a pixel is better-represented in fuzzy color histograms, even with a small number of membership functions.

Three components are linked in a Mamdani-style fuzzy inference system [63], according to 26 fuzzy rules (see Section 2.6). The final color histogram is constructed using 15 trapezoidal membership functions for each bin of the output color histogram. Because some colors (olive, purple, silver, lime, maroon) reside

very close to the others in 3-d $L^*a^*b^*$ space, we selected the remaining 15 colors out of 20 (see Section 2.5). Therefore, the final fuzzy color histogram contains 15 bins. The overview of the proposed fuzzy inference system is shown in Figure 2.8.

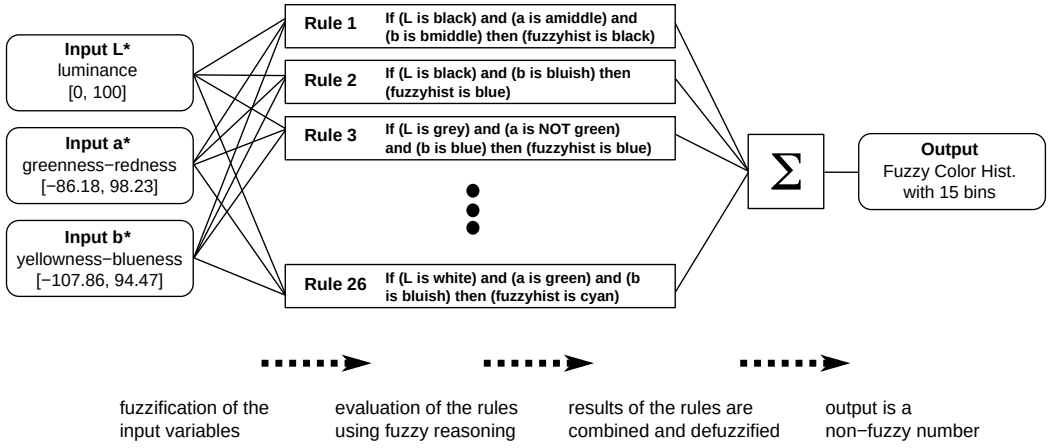


Figure 2.8: The structure of the fuzzy color histogram.

The main advantage of the proposed fuzzy color histogram over a conventional color histogram is its accuracy. Since the system is less sensitive to illumination changes and quantization errors, it performs better on shot boundary detection. Figure 2.9 displays three successive frames in a gradual transition with their fuzzy and gray-scale histograms.

For frame-dropping transformations, we estimate the missing frames using linear regression. The fuzzy color histogram of a dropped frame is predicted by averaging the features of the previous two frames:

$$H_n = \begin{cases} h_n & fd(I_n) = 0 \\ (H_{n-1} + H_{n-2})/2 & otherwise \end{cases} \tag{2.13}$$

The essential idea of using color histogram for shot-boundary detection is that color content does not change rapidly within a shot. Therefore, shot changes are detected when fuzzy color histogram difference exceeds a threshold. The dissimilarity between color histograms of successive frames is calculated with

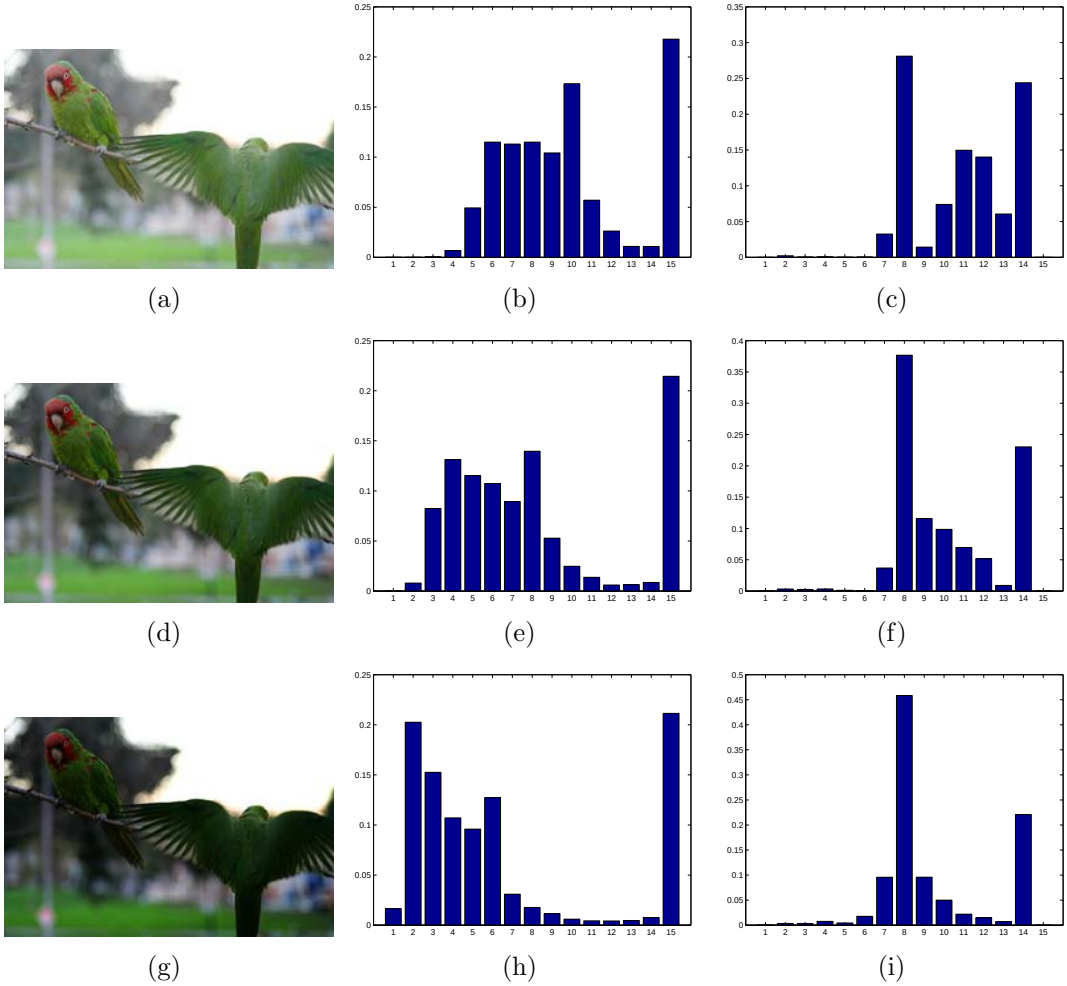


Figure 2.9: Three successive frames in a gradual transition: three successive frames (a, d, and g); grayscale histograms of the frames (b, e, and h); fuzzy color histograms (c, f, and i).

Euclidean distance:

$$D(I_n, I_m) = \sqrt{\sum_{i=1}^{b} (H_n(i) - H_m(i))^2} \tag{2.14}$$

Although the difference between color histograms of successive frames in a video is enough to detect hard-cuts, the detection of gradual transitions (i.e., dissolve and fade) requires special treatment since these transitions are less responsive. In our method, we extend color histogram difference by the algorithm

proposed in [68].

$$d_\tau(t) = \frac{1}{\tau} \sum_{i=0}^{\tau-1} D(t+i, t-\tau+i) \tag{2.15}$$

d_τ detects the transitions of duration less than or equal to τ. We interpret peaks in d_2 greater than $\theta_c = 0.15$ as hard-cuts, and the remaining peaks in d_4, d_8, and d_{16} greater than $\theta_g = 0.09$ as gradual transitions.

2.5 Experiments in *L*a*b** Color Space

We have selected popular colors, and experimented with their values in $L^*a^*b^*$ color space. $L^*a^*b^*$ is commonly preferred over RGB or HSV color spaces, because it is one of the perceptually uniform color spaces which approximates the way that human perceive color. In $L^*a^*b^*$ color space, L^* represents luminance, a^* represents greenness-redness, and b^* represents blueness-yellowness.

a^* and b^* components have more weights than L^* component. Therefore the fuzzy linking method in [51] prefers to subdivide L^* into 3 (dark, dim, and bright), a^* into 5 (green, greenish, middle, reddish, and red), and b^* into 5 (blue, bluish, middle, yellow, and yellowish) regions. We have used the same approach.

Range of $L^*a^*b^*$ color space is important for the fuzzy membership functions. L^* coordinate ranges from 0 to 100. The possible range of a^* and b^* coordinates depends on the color space that one is converting from. When converting from RGB, a^* coordinate range is [-86.1813, 98.2352], and b^* coordinate range is [-107.8617, 94.4758]. We have selected 20 colors from List of Colors [7]. Table 2.2 shows L^*, a^*, b^* values, as well as their fuzzy correspondences for each color.

2.6 Fuzzy Rules

Twenty-six fuzzy rules of the fuzzy inference system are listed in Figure 2.6. These rules are generated according to the fuzzy correspondences of output colors in Table 2.2.

Table 2.2: The colors and their fuzzy correspondences.

Color	**L***	**a***	**b***	**fuzzy L***	**fuzzy a***	**fuzzy b***
Black	0.00	0.00	0.00	black	amiddle	bmiddle
Blue	32.30	79.19	-107.86	black+grey	red	blue
Brown	64.60	10.22	69.09	grey	amiddle	yellowish
Cyan	91.11	-48.09	-14.13	white	greenish	bmiddle
Magenta	60.32	98.24	-60.83	grey	red	bluish
Lime	87.74	-86.18	83.18	white	green	yellow
Grey	76.19	0.00	0.00	grey	amiddle	bmiddle
Maroon	39.03	63.65	53.41	grey	reddish	yellowish
Navy	22.38	62.93	-85.72	black	reddish	blue+bluish
Green	66.44	-68.49	66.10	grey	green	yellow+yellowish
Olive	73.92	-17.13	75.08	grey+white	greenish	yellow
Orange	83.91	3.43	82.63	white	amiddle	yellow
Pink	92.07	11.20	1.05	white	reddish	bmiddle
Purple	44.66	78.07	-48.34	grey	red	bluish
Red	53.24	80.09	67.20	grey	red	yellow+yellowish
Silver	89.53	0.00	0.00	white	amiddle	bmiddle
Teal	69.13	-38.22	-11.23	grey+white	greenish	bmiddle
Violet	50.46	89.85	-77.24	grey	green	blue
White	100.00	0.00	0.00	white	amiddle	bmiddle
Yellow	97.14	-21.55	94.48	white	greenish	yellow

```
if (L is black) and (a is amiddle) and (b is bmiddle) then (fuzzyhist is black)
if (L is black) and (b is blueish) then (fuzzyhist is blue)
if (L is grey) and (a is NOT green) and (b is blue) then (fuzzyhist is blue)
if (L is white) and (a is amiddle) and (b is blueish) then (fuzzyhist is blue)
if (L is white) and (a is greenish) and (b is blueish) then (fuzzyhist is blue)
if (L is black) and (a is reddish) and (b is blue) then (fuzzyhist is navy)
if (L is grey) and (a is red) and (b is NOT blue) then (fuzzyhist is red)
if (L is grey) and (a is reddish) and (b is bmiddle) then (fuzzyhist is red)
if (L is black) and (a is reddish) and (b is yellowish) then (fuzzyhist is red)
if (L is grey) and (a is reddish) and (b is yellow) then (fuzzyhist is yellow)
if (L is white) and (a is amiddle) and (b is yellow) then (fuzzyhist is yellow)
if (L is white) and (a is greenish) and (b is yellow) then (fuzzyhist is yellow)
if (L is grey) and (a is reddish) and (b is blueish) then (fuzzyhist is magenta)
if (L is white) and (a is reddish) and (b is blueish) then (fuzzyhist is magenta)
if (L is grey) and (a is amiddle) and (b is yellowish) then (fuzzyhist is brown)
if (L is white) and (a is reddish) and (b is yellow) then (fuzzyhist is brown)
if (L is grey) and (a is amiddle) and (b is bmiddle) then (fuzzyhist is grey)
if (L is grey) and (a is greenish) and (b is yellow) then (fuzzyhist is green)
if (L is white) and (a is green) and (b is yellowish) then (fuzzyhist is green)
if (L is grey) and (a is greenish) and (b is bmiddle) then (fuzzyhist is teal)
if (L is grey) and (a is green) and (b is blue) then (fuzzyhist is violet)
if (L is white) and (a is red) and (b is yellow) then (fuzzyhist is orange)
if (L is white) and (a is reddish) and (b is bmiddle) then (fuzzyhist is pink)
if (L is white) and (a is amiddle) and (b is bmiddle) then (fuzzyhist is white)
if (L is white) and (a is greenish) and (b is bmiddle) then (fuzzyhist is cyan)
if (L is white) and (a is green) and (b is bluish) then (fuzzyhist is cyan)
```

Figure 2.10: Fuzzy rules.

Chapter 3

Content-based Copy Detection

With the rapid development of multimedia technologies and media-streaming, copyrighted materials become easily copied, stored, and distributed over the Internet. This situation, aside from enabling users to access information easily, causes huge piracy issues. One possible solution to identify copyrighted media is watermarking.

Digital watermarking [53] was proposed for copyright protection and fingerprinting. The basic idea is to embed an information into the signal of the media (audio, video, or photo). Some watermarks are visible (e.g., text or logo of the producer or broadcaster), while others are hidden in the signal, which cannot be perceived by human eye. Today all DVD movies, video games, audio CDs, etc. have fingerprints that prove the ownership of the material.

As a disadvantage, watermarks are generally fragile to visual transformations (e.g., re-encoding, change of the resolution/bit rate). For example, hidden data embedded on a movie will probably be lost when the clip is compressed and uploaded to a video sharing web site. Besides, temporal information of the video segments (e.g., frame number, time-code) are also important in some applications. Watermarking technique is not designed to be used for video retrieval by querying with a sample video clip.

Content-based copy detection (CBCD) is introduced as an alternative, or in fact, a complementary research field to watermarking approach. The main idea of CBCD is that the media visually contains enough information for detecting copies [36]. Therefore, the problem of content-based copy detection is considered as video similarity detection by using the visual similarities of video clips.

Our aim is to propose a multimodal framework for content-based copy detection and video similarity detection. The proposed method consists of three parts: First, a high-level face detector identifies facial frames/shots in a video clip. Matching faces with extended body regions gives the flexibility to discriminate the same person (e.g., an anchor man or a political leader) in different events or scenes. In the second step, a spatiotemporal sequence matching technique is employed to match video clips/segments that are similar in terms of activity. Finally the non-facial shots are matched using low-level visual features.

3.1 Related Work

3.1.1 Video Similarity Detection

There are notable works on video similarity detection in the literature. We preferred categorizing these work into groups based on the features used in the methods. Comparative studies are examined as a separate category.

3.1.1.1 Methods based on Spatiotemporal Similarity Matching

Similarity of temporal activities of video clips has shown promising results in video similarity detection. Techniques based on spatiotemporal sequence matching are robust to many distortions caused by digitization and encoding. Furthermore, they provide the precise temporal location of the matching video parts.

Mohan [65] presents a video sequence matching technique that partitions each frame into 3×3 image, and computes its ordinal measure to form a fingerprint.

The sequences of fingerprints are compared for video similarity matching. It was shown that matching with ordinal signatures over-performs the methods using motion or color signatures. Kim and Vasudev [49] criticize the partition size used in [65], stating that prior methods using partitions over 2×2 do not consider the effects of asymmetrical changes in the partition values. They use ordinal measures of 2×2 partitioned image, and also consider the results of various display format conversions, i.e. letter-box, pillar-box, etc.

3.1.1.2 Methods Using Color Histograms

An early and naive method for detecting identical shots is proposed by Satoh in [69]. Color histogram intersection is used for both detecting shot-boundaries and the matching shots. Yeh and Cheng [78] use a method that partitions the image into 4 regions, and then extract a Markov stationary feature (MSF)-extended HSV color histogram. Visual features extracted from video (one frame per second) are compared with an edit distance-based sequence matching method.

3.1.1.3 Methods Using MPEG Features

Some video similarity detection methods take the advantage of visual features that can be directly extracted from compressed videos. Ardizzone *et al.* [17] use MPEG motion vectors as an alternative to optical flows, and show that proposed motion-based video indexing method that does not require a full decomposition of the video has a computational efficiency.

Bertini, Bimbo, and Nunziati [20] present a clip-matching algorithm that use video fingerprint based on standard MPEG-7 descriptors. An effective combination of color layout descriptor (CLD), scalable color descriptor (SCD), and edge histogram descriptor (EHD) forms the fingerprint. Fingerprints are extracted from each clip, and they are compared using an edit distance. Proposed approach can solve the problems like structural matching (identification of dialogs in movies, anchor man in news, commercials, etc.), duplicate detection (in video

sharing web sites), and copy detection. Use of edit distance allows matching re-edited clips, or clips with different frame rates.

3.1.1.4 Methods Using Interest Points

Joly, Frelicot, and Buisson present a technique for content-based video identification based on local fingerprints in [45]. Local fingerprints are extracted around interest points detected with Harris detector, and matched with an approximate Nearest Neighbors search. In [43, 46], the same authors focus on the retrieval process of the proposed CBCD scheme by proposing *statistical similarity search* (S^3) as a new approximate search paradigm. In [44], Joly *et al.* present distortion-based probabilistic approximate similarity search technique (DPS^2) to speed-up conventional techniques like range queries and sequential scan method in a content-based copy retrieval framework.

Zhao *et al.* [83] extract PCA-SIFT descriptors for matching with approximate nearest neighbor search, and SVMs for learning matching patterns in their near-duplicate keyframe identification method.

3.1.1.5 Trajectory-based Methods

Law-To *et al.* present a video indexing approach using the trajectories of points of interest along the video sequence in [56, 54]. Local fingerprints are based on the descriptors used in [43, 46, 45, 44]. The method is based on two steps: computing temporal contextual information from local descriptors of interest points, and the use of contextual information in a voting function for matching video segments. Evaluations show that taking the labels into account in voting process improves the precision.

Poullot *et al.* focus on enhancing the scalability of content-based copy detection methods in [67], for monitoring a continuous stream (a TV channel) against a database of 250,000 hours of reference video in real-time. The method introduces three improvements (Z-grid for building index, uniformity-based sorting,

and adapted partitioning of the components) to the retrieval process.

3.1.1.6 Methods Using Combinations of Visual Features

Basharat, Zhai, and Shah present a video-matching framework using spatio-temporal segmentation in [18]. Trajectories of SIFT interest points are used for generating video volumes. Then, a set of features (color, texture, motion, and SIFT descriptors) is extracted from each volume, and the similarity between two videos is computed with a bipartite graph and Earth Mover's Distance (EMD).

Can and Duygulu propose an automatic method that finds all repeating video sequences inside a long video or video collection in [24]. This method uses HSV statistics (mean and standard deviation of HSV bands for 5×7 grid), and SIFT features quantized with bag-of-features approach.

3.1.1.7 Comparative Studies

Hampapur and Bolle [36] compare some simple color histogram-based and edge-based methods for detecting video copies. Another study by Hampapur, Hyun, and Bolle [37] compares motion direction, ordinal intensity signature, and color histogram signature matching techniques. As a result of this comparative study, ordinal features seem to overperform other methods.

State-of-the-art copy detection techniques are evaluated in the comparative study by Law-To *et al.* in [55]. Compared descriptors are categorized into 2 groups: global and local. Global descriptors use techniques based on the temporal activity, spatial distribution and spatio-temporal distribution. Local descriptors that are compared in this study are based on extracting Harris interest points for keyframes with high global intensity of motion (AJ), for every frame (ViCopT), and interest points where image values have significant local variations in both space and time. It is stated that no single technique is optimal for all the applications, but ordinal temporal measure is very efficient for small transformations.

3.1.2 Applications of Video Similarity Detection

3.1.2.1 Topic tracking across different channels

Tracking news topics across different TV channels or stations is one of the promising applications of video similarity detection. Users may want to track news stories from multiple sources in order to obtain more objective and comprehensive information.

Satoh *et al.* present a method to detect scene duplicates in order to identify the same event reported in the different programs in [70]. These types of duplicates are composed of different footages taking the same scene, same event at the same time, but from different viewpoints. The proposed method uses matching of temporal pattern of discontinuities obtained from trajectories of feature points, and is invariant to camera angle difference and different video captions. Additionally, a two-stage approach is employed to accelerate the method: filtering by using temporal discontinuity patterns, and precise matching by normalized cross correlation between inconsistency sequences of trajectories. Proposed technique was extended by Wu *et al.* in [77].

Zhai and Shah propose a method that uses the combination of both visual similarity and the spoken content to link news stories on the same topic across multiple channels in [81]. The method tries to match facial keyframes by comparing the 3D color histograms of body regions extended from the faces. Non-facial frames are then matched using affine transformation between keyframes. Textual information is extracted with an automatic speech recognition system, and then the visual and textual information are fused in order to link the stories semantically. Output of the system can be used in a story-ranking task.

Another framework that utilizes low-level similarity, visual near-duplicates and semantic concepts for topic threading and tracking across different channels is proposed by Hsu and Chang [39]. Visual near-duplicates, obtained by modeling and matching images with attributed relational graphs (ARGs), enhance story tracking.

3.1.2.2 News topic threading, novelty/redundancy detection

News topic threading is a video-similarity application for effectively searching and browsing news video clips. While threading a news story with video clips gathered from different TV channels or news networks, some reports include fresh textual and visual content (novelty), some of them carry information that is already known (redundancy). Reorganizing news stories by novelty along with avoiding redundant ones promises a better usability and user experience.

Wu *et al.* present techniques for browsing the gradual change of a news topic over time, identifying the novel story, as well as the evolving topic and redundant story in [76]. For grouping time-evolving news stories, a co-clustering approach is used in integrating textual and visual concepts. As a result, a topic structure binary tree can be modeled to represent the dependencies among different news stories. A recent work [75] focuses on measuring novelty and redundancy of news stories in multiple languages. All the mentioned studies use textual information, extracted by speech recognition techniques, complementary to visual features in order to detect video similarities and identify novel/redundant stories.

3.1.2.3 Commercial film detection and identification

There are important reasons for identifying commercials. Some companies may want to verify that their commercials are broadcast. Consumers may need a commercial management system for the classification of commercials, and for observing competitors behaviors.

Duan *et al.* [31] propose a multimodal scheme for identifying, categorizing and finding commercials. Finding the product information with a text-processing module, identifying boundaries of commercial by using both visual and audio features, and SVM-based classifier are the important parts of the proposed framework.

3.2 Detecting and Matching Facial Shots

The first part of video similarity detection of our CBCD framework is to detect facial shots in reference and query videos. Although the detection process for reference videos does not require any specific adjustment, we need to consider the video manipulations applied to query videos while detecting faces in query videos. For example, a Median filter is applied on noisy query frames, a smaller scale is selected for minimum face size within the window of picture-in-picture transformation, and so on.

After obtaining faces from detected facial shots, we extract visual features and match them to find the matching video segments. The following subsections give details of face detection, false-alarm elimination, feature extraction and the matching parts of the method.

3.2.1 Face Detection with Haar-like Features

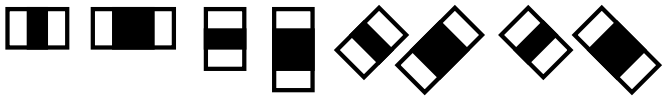
We use an object detector, proposed by Viola and Jones [73], improved by Lienhart and Maydt [59], for detecting faces in video frames/shots.

The face classifier (named as *cascade of boosted classifiers working with haar-like features*) is trained with positive and negative instances. Responses of Haar-like features (shown in Figure 3.1) are extracted, and a decision tree-based classifier is trained for face samples. These features are specified by their shapes (e.g., a-1, b-2), position within the region of interest, and the scale.

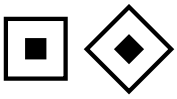
In order to search for the face in the frame, the algorithm moves the search window across the frame while checking each location using the classifier. The scan procedure is done several times at different window scales for finding faces with different sizes. For this purpose, the classifier is designed so that it can be easily resized. A binary decision is generated as a result of the classifier.

In our implementation, we preferred using Canny edge detector to reject some

(a) Edge features



(b) Line features



(c) Center-surround features

Figure 3.1: Haar-like features.

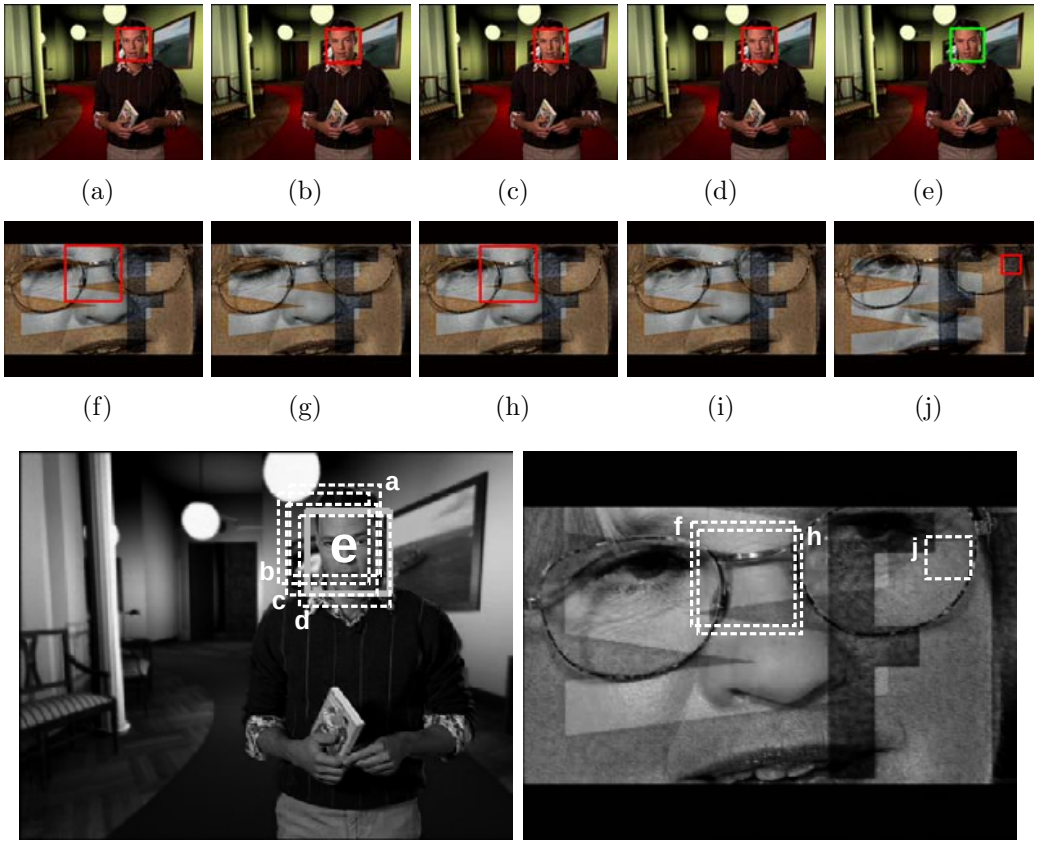
image regions that contain very few or too much edges and thus cannot contain faces. The particular threshold values are tuned for face detection and in this case the pruning speeds up the processing. The detector finds faces with at least 20×20 pixels, and returns only the largest object (if any) in the image. For the windows of picture-in-picture transformation, we use a smaller scale (10×10 pixels), since the face are generally half-size of the ones in full frame.

3.2.2 Eliminating False Alarms by Tracking

The *cascade of boosted classifiers working with haar-like features*-based face detector in OpenCV [10] tends to generate many false alarms. From our observations, these false alarms can survive very few frames.

We modified the face detection algorithm by taking spatial and temporal information into account to eliminate false detections, so that face detection would work more stable than the original method.

The method works in the following manner. Each detected face is considered as a candidate. The candidate faces with a stable behavior both in time and location (space) are assumed to be the real faces. To accomplish the aforementioned stability, we track candidate faces in successive frames. If a candidate face appears at least f_s successive frames, algorithm fuses multiple face detections and



(k) Face detected (l) No face reported

Figure 3.2: Examples of face detection with false alarm elimination.

marks as a face of the related shot.

Figure 3.2 shows two face detections, and how false alarms are eliminated in detail. Red rectangles are candidate (unstable) detections, while green ones are stable faces. Successive frames from different video clips (a-e and f-j) are shown in this figure. (k) and (l) are the spatial locations of detected faces over 5 frames for each clip. Since candidate faces are spatio-temporally stable during (k), we extract the last candidate face (e). However, in the second example, candidate faces are not consecutive and spatially stable. Therefore, no face is extracted from this sequence of video.

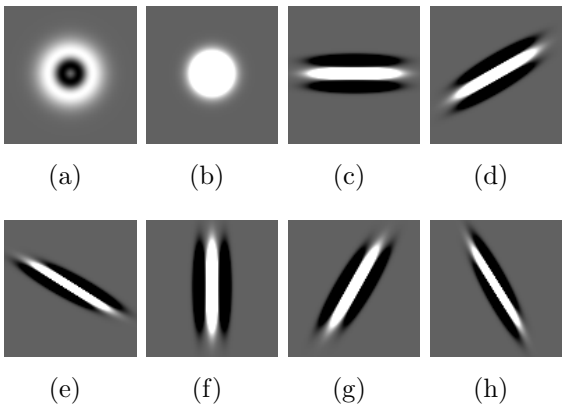


Figure 3.3: Point-spread functions designed with difference of offset Gaussians. All the parameters of Gaussian functions are adapted from [62]. (a) and (b) are the spot filters, linear combinations of two circular concentric Gaussian functions. (c-h) are the bar filters with different orientations, designed with linearly combining three offset-identical Gaussian functions.

3.2.3 Improving the Accuracy of Facial Shot Detection

We have extracted all the facial shots with extended body regions from our reference videos. For testing purposes, 1000 faces are randomly selected from facial shots, manually labeled as face or non-face. 75.5% of the extracted faces were real faces with body regions, the rest were false alarms.

For increasing the accuracy of facial shot detection, we classified these images according to their responses to spot and bar filters in different orientations (see Figure 3.3) proposed in [62]. Each facial region is divided into 4 patches, and then the mean and variance of each filter response is calculated for each patch. A 64-d feature vector is computed using the responses of 8 filters (4 patches x 8 filters x 2 values).

Each feature vector is normalized. Among a number of classifiers experimented (i.e., k-nearest neighbor, decision tree, SVM, Naive Bayes); the Nearest Neighbor (NN) classifier gives the best accuracy. To evaluate the performance of NN on this dataset, we tested with 10-folds cross-validation. 81.3% of the instances were correctly classified. As a result of this classification, we increased the accuracy of facial shot detection from 75.5% to 86.2% (see Table 3.1).

Table 3.1: The confusion matrix of the NN classification of facial and non-facial images using bar and spot filter responses.

		Classified as	
		Face	Non-face
Class	Face	676	79
	Non-face	108	137

3.2.4 Extracting Faces from Shots

Since our aim is not recognizing faces, label persons, etc., extracting only the faces does not seem to be an efficient way to match faces in video clips. Yet, face matching has some well-known drawbacks, such as sensitivity to pose and illumination changes. To overcome this problem, we employ the method proposed by Zhai and Shah [82]. Instead of extracting visual features from the face, we extend the detected region to cover the upper part of the body. Therefore, we can match the shots with the same person (e.g., an anchor man or a political leader) by considering the clothes or some background as well. Figure 3.4 displays detected faces and their extended body regions.

3.2.5 Matching Facial Shots

After finding the facial shots from both query and reference videos, some visual features are extracted from the extended body region image of each face. We preferred using color-based MPEG-7 descriptors (explained in detail in Section 3.5.1). Edge-based methods are excluded because of the distortions applied on query videos. Besides, homogeneous texture descriptor requires a minimum of 128×128 image, which is not the case for most of the extracted face images. We give detailed evaluation of the method with different visual features and their combinations in Chapter 4.

(a) Frames with detected faces

(b) Detected faces

(c) Extended body regions

Figure 3.4: Examples of extended body regions: first five examples are faces of the same person in different events/scenes. Because our goal is to match shots instead of faces, we use extended body regions (c). Solely the facial regions do not give discriminative visual features; on the other hand, differences of clothing help us identify the same person in different scenes.

3.3 Subsequence Matching of Activity Time-Series

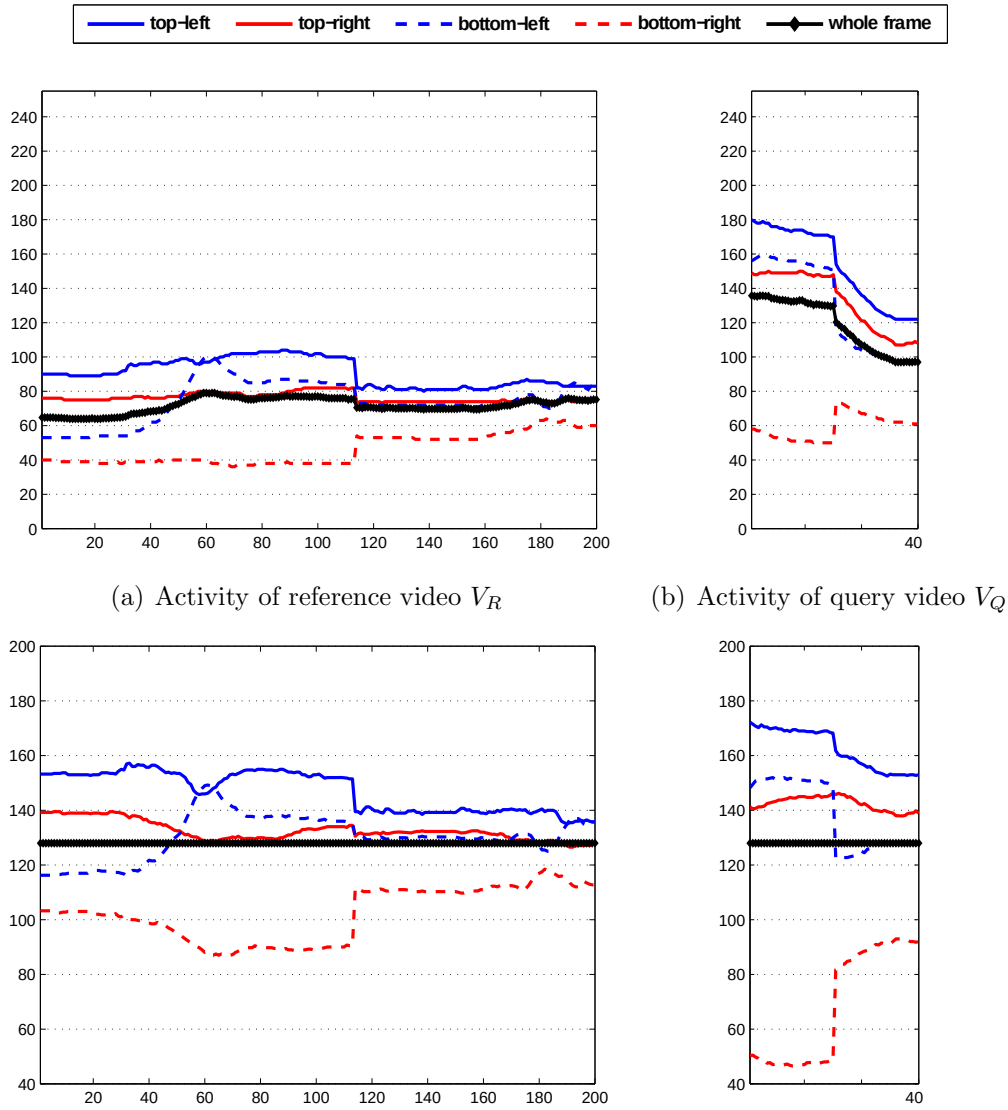
As we mentioned earlier, spatiotemporal sequence matching is a technique robust to many distortions caused by digitization and encoding. In addition, it provides the precise temporal location of the matching video parts. These two features are crucial for a video copy detection system.

In contrast with the prior works [49, 65], we preferred using the numerical intensity averages of partitions instead of their ordinal measures. The reason is that when the length of the query video is small (query videos used in TRECVID are between 3 seconds to 3 minutes [4]), there might be more than one video sequence that have very similar fingerprints.

3.3.1 Problem Definition

Here are some notions used in this section. $V = \{V[0], ..., V[n-1]\}$ represents a video with n frames. $V[i] = \{V^1[i], V^2[i], V^3[i], V^4[i]\}$ denotes i^{th} frame with 4 features, which are the average intensity values of 4 partitions (for *top-left*, *top-right*, *bottom-left*, and *bottom-right* regions). Then V^j represents a sequence of j^{th} partition. A subvideo of video V with N frames is defined as $V[p : p+N-1]$, where the first frame is $V[p]$.

The problem of subsequence matching of time-series can be defined as follows: Given a query video V_Q with N frames, find the matching subset of reference video V_R with M frames, if the dissimilarity between two video clips $D(V_Q, V_R)$ is less than a threshold ϵ.



(a) Activity of reference video V_R

(b) Activity of query video V_Q

(c) Activity of reference video V_R

(d) Activity of query video V_Q

Figure 3.5: Average intensity values for reference (a) and query (b) videos. Normalized average intensity values for reference (c) and query (d) videos.

3.3.2 Activity Subsequence Matching Method

Figure 3.5 represents a reference video with 200 frames, and a query video with 40 frames. Although they look quite different, we know that the query video V_Q is originated from subvideo $V_R[81:140]$.

Due to the manipulations in query generation process (changing quality, gamma value, contrast, etc.), average intensity values of the query frames may be higher or lower than the original video. Therefore, we need to normalize average intensity values for both reference and query videos (see Figure 3.5). This is done by adjusting the mean of the values to be the middle value of gray scale. The procedure was referred as histogram equalization in [49].

After this point, the problem becomes *matching time-series (signals) with different amplitudes*. To overcome afromentioned differences in the amplitudes of time-series, we define $\alpha_X[i]$ as the maximum distance of a partition intensity value to the center-point ($c = 128$) for the i^{th} frame of video V_X, and $\beta_X[p : p+N-1]$ as the maximum value of $\alpha_X[i]$ for all frames of subvideo $V_X[p : p+N-1]$. Then β_X is calculated for all frames of V_X.

$$\alpha_X[i] = \max_j \left| V_X^j[i] - c \right|, \text{ where } j \in \{1,2,3,4\} \tag{3.1}$$

$$\beta_X[p : p+N-1] = \max_i \alpha_X[i], \text{ where } i \in [p, p+N-1] \tag{3.2}$$

$$\beta_X = \beta_X[1 : M] \tag{3.3}$$

By using α and β functions, we calculate the dissimilarity between a query video and a reference subvideo as:

$$D(V_Q, V_R[p : p+N-1]) = \frac{\sum_{i=1}^{N} \sum_{j=1}^{4} \left| \overline{V_Q^j[i]} - V_R^j[p+i] \right|}{N} \tag{3.4}$$

$$\overline{V_Q^j[i]} = \frac{V_Q^j[i] - c}{\beta_Q / \beta_R[p : p+N-1]} + c \tag{3.5}$$

Therefore, the dissimilarity between a query and a reference video can be defined as the minimum of subvideo dissimilarities:

$$D(V_Q, V_R) = \min_p \frac{D(V_Q, V_R[p : p+N-1])}{N}, \text{ where } p \in [1, M-N] \tag{3.6}$$

If the dissimilarity $D(V_Q, V_R)$ is less than a threshold value ϵ, we report that the query video V_Q can be a copy of V_R starting from the frame number p with a decision score of $1 - D(V_Q, V_R)$. Threshold value depends on the detected transformations applied to the video. If the query video has noise, or picture-in-picture transformation, the dissimilarity values would be higher. Therefore, we increase the decision threshold ϵ in such cases.

Activity series of four query videos originated from the same reference sub-video is shown in Figure 3.6.

3.4 Non-facial Shot Matching with Low-level Visual Features

The third step of our proposed method consists of extracting low-level visual features from reference and query videos, and keyframe-based matching of video segments. If a shot is marked as facial (i.e., a face was detected in one of the frames of this shot), facial shot matching technique handles detecting the copies. Low-level feature matching part is complementary to facial shot matching; however, it uses low-level color and texture information of the whole frame.

3.4.1 Visual Features and Similarity Measures

The following MPEG-7 features (three color, and two texture-based descriptors) are extracted and compared for image-to-image similarity. MPEG-7 visual descriptors are explained in detail in [9, 64].

3.4.1.1 Scalable Color Descriptor

Scalable color descriptor (SCD) is a color histogram in the HSV color space, encoded by a Haar transform. The histogram values are extracted, normalized

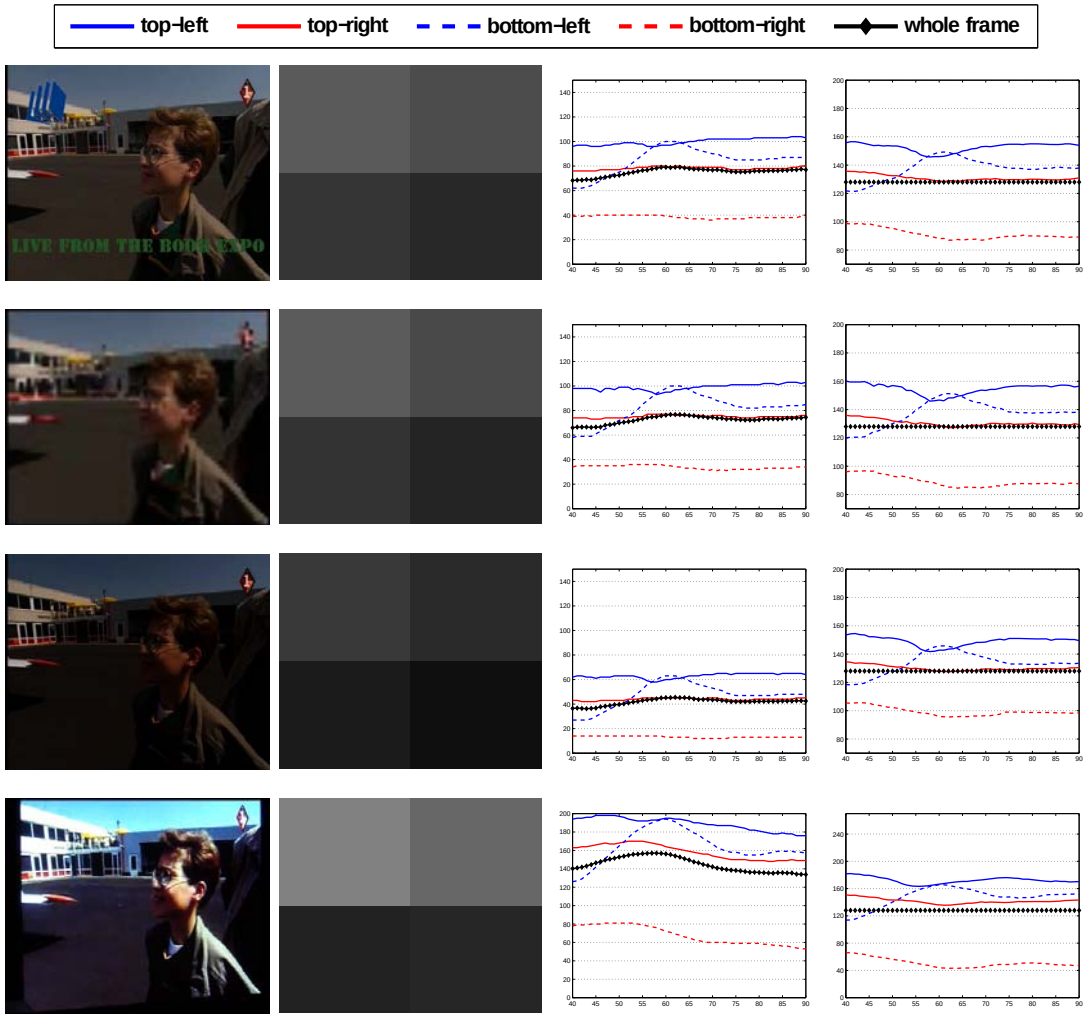


Figure 3.6: Generation of the activity sequence: frames from query videos that correspond to the same reference video *(first column)*, average intensity values for 2x2 partitions *(second column)*, spatio-temporal activities of frames *(third column)*, and normalized spatio-temporal activities of frames *(fourth column)*. The first query video is very similar to the original video, except for the logo insertion. The second and the third videos have gamma-change and strong re-encoding transformations. The fourth video is recorded with a camcorder. Although the spatio-temporal activities and average intensity values are very different, their normalized intensity sequences are close to each other.

and nonlinearly mapped into integer representation. This descriptor is scalable in terms of bin numbers and bit representation accuracy. In our method, we used the number of coefficients (histogram bins) as 128. l_1-norm based matching is used for comparing scalable color descriptors.

3.4.1.2 Color Layout Descriptor

Color layout descriptor (CLD) is a compact and resolution-invariant color feature that efficiently represents spatial distribution of colors for high-speed image retrieval applications. Input image is divided into 8×8 blocks, transformed by discrete cosine transformation (DCT), and DCT coefficients for the luminance and the chrominance are extracted.

For matching two CLDs, $\{DY, DCr, DCb\}$ and $\{DY', DCr', DCb'\}$, the following distance measure is proposed in [64]:

$$D = \sqrt{\sum_i w_{yi}(DY_i - DY_i')^2} + \sqrt{\sum_i w_{bi}(DCb_i - DCb_i')^2} + \sqrt{\sum_i w_{ri}(DCr_i - DCr_i')^2} \tag{3.7}$$

3.4.1.3 Color Structure Descriptor

Color structure descriptor (CSD) is a color feature descriptor that represents an image by both color distribution (similar to color histogram) and the local spatial structure of the color. An 8×8 element is used to embed color structure information into the descriptor. As an advantage over color histogram, CSD can discriminate between two images, similar in terms of a given color, but different regarding the structure of the groups of pixels having that color. CSD uses the l_1-norm for matching as the similarity measure.

3.4.1.4 Homogeneous Texture Descriptor

Homogeneous texture descriptor (HTD) is designed to search and browse through large collections of similar patterns. The region texture is represented using the mean energy and energy deviation in 30 frequency channels.

The similarity between a query image (TD_{query}) and a reference image ($TD_{reference}$) is measured by summing the weighted absolute difference between two sets of vectors:

$$D(TD_{query}, TD_{reference}) = \sum_{k} \left| \frac{TD_{query}(k) - TD_{reference}(k)}{\alpha(k)} \right| \tag{3.8}$$

where the recommended normalization value $\alpha(k)$ is the standard deviation of all $TD_{reference}(k)$ values. For intensity invariant matching, the first component is not used in computing the dissimilarity.

3.4.1.5 Edge Histogram Descriptor

Spatial distribution of five types of edges is calculated in edge histogram descriptor (EHD). The image is divided into 4×4 subimages, and then the edges in 16 subimages are categorized into five types: vertical, horizontal, 45° diagonal, 135° diagonal, and non-directional edges. As a result, 80 histogram bins are required.

For matching edge histograms, edge distribution for the whole image and some horizontal and vertical semi-global distributions are required to improve the performance. These global and semi-global histograms can be directly calculated from the 80 local histogram bins. We use a total of 150 bins (80 for local, 5 for global, 65 for semi-global), and l_1-norm for similarity matching:

$$D(A, B) = \sum_{i=0}^{79} |h_A(i) - h_B(i)| + 5 \times \sum_{i=0}^{4} |h_A^g(i) - h_B^g(i)| + \sum_{i=0}^{64} |h_A^s(i) - h_B^s(i)| \tag{3.9}$$

where $h_A(i)$ and $h_B(i)$ represent the normalized local histogram bin values, $h_A^g(i)$ and $h_B^g(i)$ represent the normalized bin values for the global edge histograms, and finally $h_A^s(i)$ and $h_B^s(i)$ represent semi-global-edge histograms of image A

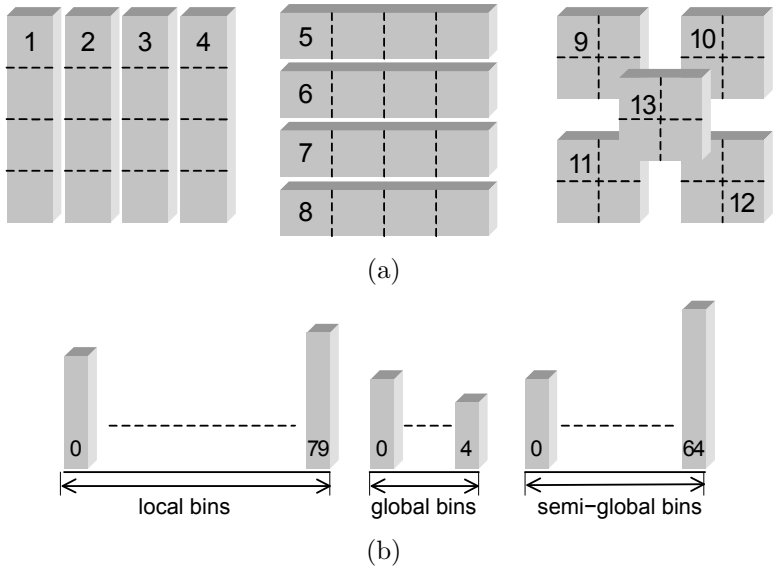


Figure 3.7: Edge histogram descriptor: subsets of local edge histograms to generate semi-global edge histograms (a), and overall edge histogram with 150 bins (b).

and B, respectively. Semi-global edges and the overall edge histogram are shown in Figure 3.7.

3.4.2 Extracting Features from Query and Reference Videos

Selecting representative frames, namely *keyframes*, is a common approach for reducing the amont of data to store and to index for efficient content-based search. In the preprocessing (off-line) stage, the visual features of each keyframe of reference videos are computed and stored in a structure. A keyframe is defined as the median frame of the shot in our system.

It is important to index the low-level visual features extracted during the off-line step in order to accelerate the similarity search. We use k-d trees to store SCD (128-d), CSD (64-d), and EHD (150-d) descriptors. Because these descriptors use l_1-norm as the similarity measure, exact and approximate nearest neighbor search is highly efficient using ANN [3] library. The remaining descriptors (CLD and HTD) need to be compared one-by-one.

Extracting features from query keyframes requires a mask for picture-in-picture transformation window (if any), and the still regions. After discarding patterns, texts, and other inserted videos from the query keyframes, the rest of the frame represents the original content better. The overview of low-level feature matching part is shown in Figure 3.8.

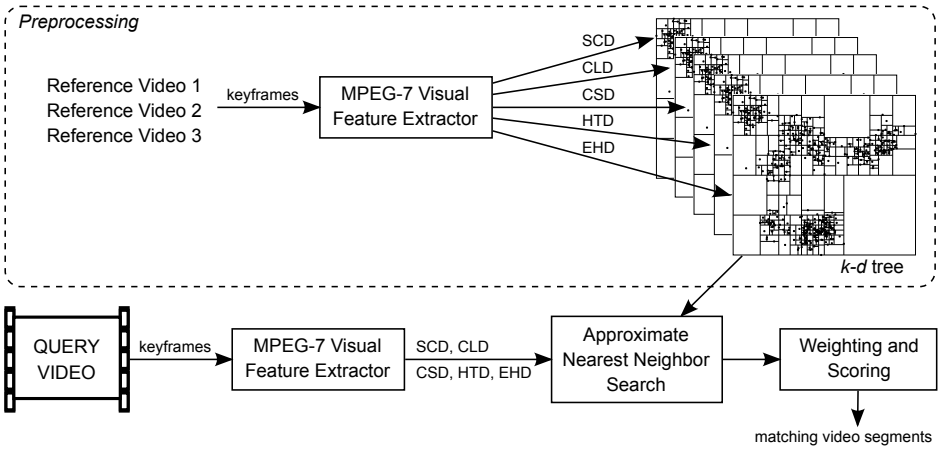


Figure 3.8: The overview of the low-level feature matching algorithm.

3.4.3 Variable-weighted Feature Similarity Calculation

A common approach for visual feature weighting is to assign fixed weights to each visual feature, as used in [20]. However, in video copy detection, some copies can be only identified by edge-based feature similarity, while another one may better respond to color layout based similarity. This is simply a result of various transformations applied on query videos. If noise is added to video, we can use color structure information. If there occurs a change in the color (e.g., camcording, change of gamma, etc.), edge-based comparisons are likely to give better matches. As a result, an automatic copy detection system cannot decide which visual feature is appropriate for matching a query video. Most of the content-based image/video retrieval systems prefer using fixed weights for visual features, or simply take the average of the similaries of different features.

Our solution is to use a variable-weighted feature similarity calculation based

on the success rate of the visual similarities of different features. When finding the matches between interest points of different images for panoramic image stitching applications, two closest matches are compared to each other. If two distances are too close to each other, the algorithm discards the feature point. Based on this idea, we define the success rate (weight) of a feature as the ratio of similarity values of the most similar match to the 5^{th} one. So the weights for each visual feature are calculated for each query keyframe separately.

Keyframe-based similarities are calculated with variable weighted MPEG-7 visual features. Then the most similar and most voted matching reference videos are reported as copy candidates.

3.5 Combining Results

Each method (facial shot matching, activity subsequence matching, and low-level feature matching) returns the best matches for all the queries. When combining the results, some of them point to the same reference video and similar temporal locations. These candidate results are merged and reported with the rest of the matching candidates. Consequently at most three copies are reported for each query video.

Chapter 4

Evaluations and Experiments

4.1 TRECVID CBCD Task Dataset

4.1.1 Reference Dataset

The reference dataset consists of approximately 100 hours of Sound & Vision data used as training and test videos for TRECVID 2007 search and HLF tasks, plus another 100 hours of Sound & Vision data prepared for TRECVID 2008 search and HLF tasks. In total there are 438 reference video files.

- TV 2007 Sound & Vision Development: 110 video files, 30.66GB, ∼50 hours
- TV 2007 Sound & Vision Test: 109 video files, 29.27GB, ∼50 hours
- TV 2008 Sound & Vision Test: 219 video files, 59.9GB, ∼100 hours

4.1.2 Query Dataset

Query dataset prepared for TRECVID 2008 CBCD task is constructed using ∼200 hours of reference videos (see Section 4.1.1) and videos not in the reference

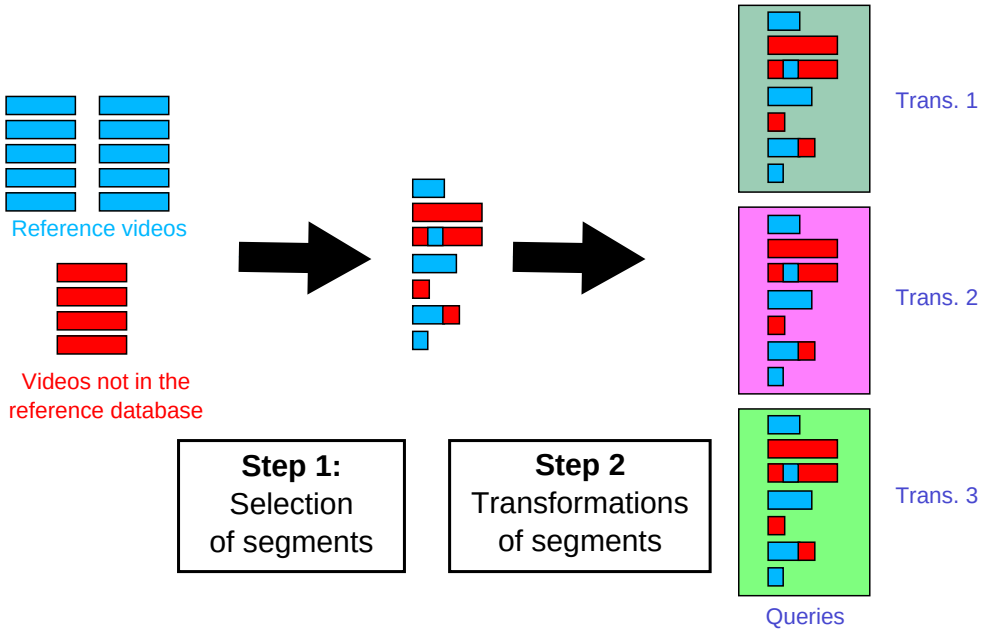


Figure 4.1: Generation of query videos by INRIA-IMEDIA [4]

database (to test false positive rate). The 2007 BBC rushes video was used as non-reference data. Some of the queries are composed of a segment of reference videos, while some may contain no reference video segments. There are three types of queries:

- Type 1: Query video is a transformed fragment of reference data;
- Type 2: Query video contains a transformed fragment of reference data;
- Type 3: Query video is a transformed fragment of a video not in reference database.

67 video segments are prepared for each type. By applying 10 different transformations (cf. Table 1.1) to all generated videos, final query videos are generated. The process is depicted in Figure 4.1. As a result, there are total of 2010 MPEG-1 videos (34.17GB), which is about 80 hours of video segments with various transformations applied. The important events that occur in query videos are shown in Figure 4.2.

Figure 4.2: The important events that occur in query videos: successive frames with frame-dropping transformation (*first row*), cut and dissolve transitions (*second row*), fade-in transition (*third row*), shot-boundaries for a picture-in-picture transformation-applied video, foreground changes at 1087, background at 1779 (*fourth row*), fast moving object in the scene (*last row*). Shot-boundaries during cut/gradual transitions (row 2-3), and for background and foreground videos (row 4) are detected, while dropped frames (row 1) and fast object movements are ignored.

4.2 Evaluation of Noise Detection

We used the query video set of TRECVID 2008 CBCD task, and extracted 1 frame per 2 seconds for each of 2010 videos. After decoding the videos, 33 478 images are manually labeled as 1 or 0, indicating that the frame is highly noisy or not.

Median filters of different sizes are evaluated and compared in an ROC curve (see Figure 4.3). It is shown through experiments that the setting with $s = 3$ and $th_n = 3.51$ gives an accuracy of 90.9% with a false alarm rate of 14.8%.

Most of the false alarms are caused by query videos that have noise originally, but not as a transformation. It should be noted that frames with sea, wavy water, or a textured background generally give high noise detection outputs.

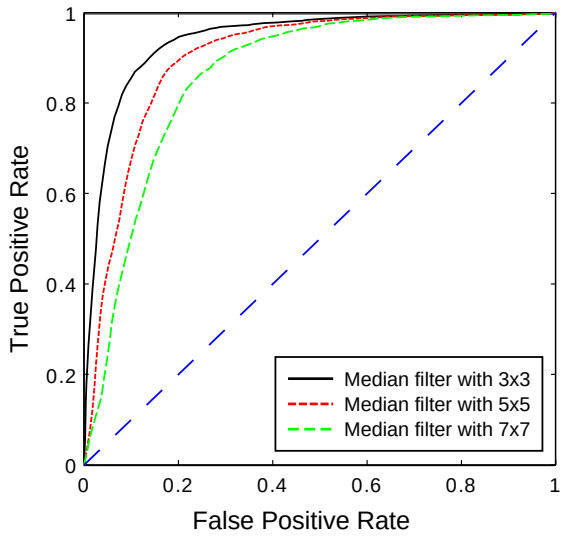


Figure 4.3: The ROC curve of the noise detection method with different median filter settings.

4.3 Evaluation of Picture-in-Picture Transformation Detection

Out of 2010 query videos, 545 of them include picture-in-picture transformation. We obtained the scale and offset information of all picture-in-picture transformations by processing the ground-truth data used for generating query videos.

Our method has reached 86.79% of recall rate, where false alarm rate is 16.93%. Missed picture-in-picture transformations are generally caused by complex transformations, i.e., (8), (9), and (10) in Table 1.1.

4.4 Evaluation of Shot-boundary Detection Algorithm

We selected a set of shot-boundary detection algorithms from the literature for a comparative evaluation of our method. The factors we considered for the selection of these algorithms are the ease of their implementation and the presence of distinct features to be used in comparison. The source codes for most of these algorithms are not available. For the algorithms with available source code, the frame-dropping transformation causes many false alarms with default settings. Therefore, we decided to create our own implementations in order have a consistent and fair evaluation of the algorithms. Since many design details are unspecified in the literature, we tried to find the optimum values for the parameters experimentally.

The following algorithms are selected for our test:

1. *Color histogram (CH):* Short transitions and hard-cuts can be detected by using simple color histogram-based methods. In this method, *RGB* and *L*a*b** color spaces are quantized into 27 equal subspaces. The histogram

h_n of image I_n is defined as:

$$h_n(b_1 \times b_2 \times b_3) = \frac{|\{p|p \in I_n(r,c)\}|}{h \times w} \tag{4.1}$$

where $p_r/b = b_1,\ p_g/b = b_2,\ p_b/b = b_3$ for RGB color space. If the histogram difference of two successive frames exceeds a threshold value, a shot boundary is found. Details of such an algorithm are provided in [26, 58, 28, 27, 22].

2. *Probabilistic block intensity (PBI):* Probabilistic block intensity is a statistical method based on the mean and standard deviation of the pixels in image regions. This technique is discussed in [22], and implemented in [48]. Although its tolerance to noise is a great advantage, this method tends to generate many false alarms. In our experiments, each frame is divided into 16 blocks.

3. *Edge tracking (ECR):* Edge change ratio based shot-boundary detection methods are discussed in [58, 22]. The ratio of the edges that enter and exit between two successive frames are used to determine shot boundaries. Edge-based methods are less sensitive to illumination changes, and they give better results in gradual transitions.

4. *Local keypoint matching (KM):* Recognizing the objects and scenes throughout the video is the basic idea of the keypoint matching-based shot-boundary detection methods. The algorithm proposed in [40] matches the objects between consecutive frames, and determines if there is a shot boundary. We use scale invariant feature transform [61] and a simple matching algorithm for this purpose.

Our tests with 50 query videos, which represents each transformation type with at least 4 videos, showed that fuzzy color histogram-based shot-boundary detection method can achieve higher accuracy values, while reducing false alarms. Table 4.1 gives the recall and precision values for the compared algorithms. F1 scores are also provided as a measure that considers both the precision and the recall rates.

Table 4.1: Experimental results of shot-boundary detection algorithms.

Method	**Recall**	**Precision**	**F1**
RGB CH	0.6284	0.6862	0.6560
*L*a*b** CH	0.5939	0.6624	0.6263
PBI	0.3218	0.0225	0.0421
ECR	0.5862	0.3542	0.4416
KM	0.4789	0.4496	0.4638
Fuzzy CH	0.7165	0.8348	0.7711

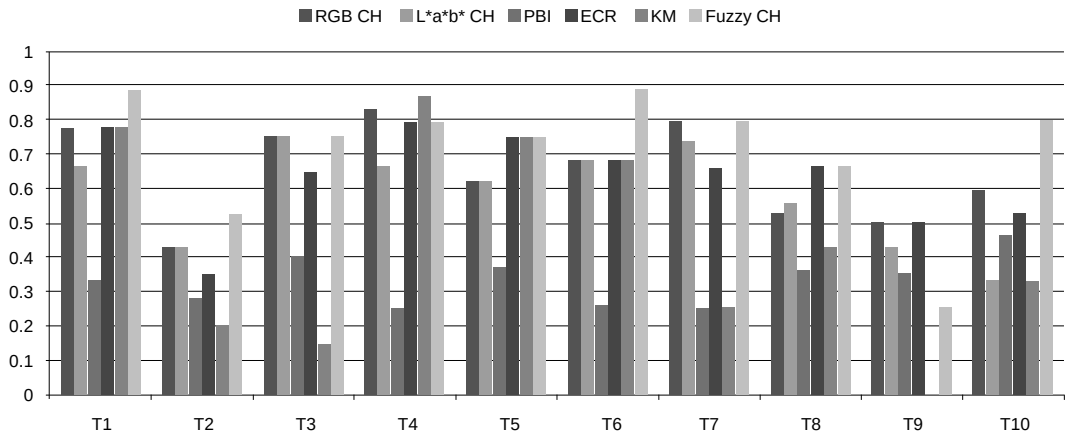


Figure 4.4: Recall values of shot-boundary detection algorithms for different transformation types.

It should be noted that the methods selected for comparison could perform much better for detecting shot-boundaries of videos on which none of the transformations listed in Table 1.1 are applied. Our test set consists of videos manipulated with these transformations. The challenge here is to detect all shots, including background and foreground videos for picture-in-picture transformations, without being affected by frame-dropping, noise, pattern insertion, strong re-encoding, etc.

Methods have different accuracy values depending on the transformation type. Figure 4.4 shows the recall values of shot-boundary detection methods for 10 types of transformations. For most of the transformations, proposed fuzzy color histogram-based method performs better than the other techniques.

Transformations of T2, T8, T9, and T10 (see Table 1.1), which include picture-in-picture transformation, are the most challenging ones. We increase the overall recall rate in these transformations from 48.74% (best among others) to 62.18% (Fuzzy CH). Our method also achieves a lower false alarm rate with a precision of 93.67%, whereas the precision values of the other methods could only reach up to 53.21%.

It may be expected that the proposed fuzzy color histogram-based method will have some drawbacks when the processed videos/frames are in grayscale. In order to evaluate the performance of the proposed method under this circumstance, we converted the same 50 query video into grayscale, and then run the SBD algorithm for these videos. Results are given as a confusion matrix in Table 4.2.

Table 4.2: The confusion matrix of fuzzy color histogram-based shot-boundary detection method using the same 50 query videos, but with grayscale frames.

		Classified as	
		SB	Non-SB
Class	SB	207 *(tp)*	85 *(fn)*
	Non-SB	66 *(fp)*	∼

Experiments with grayscale videos show that the recall (70.8%) and the precision (75.8%) values are slightly affected by this change. Although our method is a color histogram-based technique, grayscale pixels can be defuzzified into the colors other than white/gray/black, depending on the results of 26 fuzzy rules. This property provides enough flexibility for the method to detect shot-boundaries of grayscale videos. We also observed that the increase in false-alarm rate is mostly because of the shot-boundaries detected close to the beginning of gradual transitions. Nevertheless, the proposed method still outperforms other techniques.

4.5 Evaluation of Facial Shot Matching

The color-based MPEG-7 descriptors used in facial shot matching method are color structure (CSD), scalable color (SCD), and color layout (CLD) descriptors. The number of correct detections obtained by using each descriptor is listed in Table 4.3. Note that color descriptors are generally illumination-dependent feature vectors. We were only able to implement the similarity function of CLD illumination-invariant by ignoring DC components of the descriptor.

Table 4.3: Number of correct detections for facial shot matching method with different visual features. Total number of copies that can be detected for each transformation type is 134.

					Proposed Method
	CSD	SCD	CSD+SCD	CLD	CSD+SCD+CLD
T1	0	0	0	18	18
T2	25	25	34	28	36
T3	38	40	46	42	48
T4	22	28	35	41	49
T5	24	21	30	49	54
T6	22	21	31	37	49
T7	12	14	16	47	51
T8	28	34	36	21	40
T9	26	25	31	10	32
T10	12	12	20	22	30

Out of 9,612 query faces and 32,597 reference faces, our method successfully retrieved a total of 407 copies, which corresponds to ∼30% of the copies. We also believe that detecting facial shots in reference videos improves the quality of a video database and enables users to query the database with face samples.

4.6 Evaluation of Activity Subsequence Matching

Subsequence matching of activity time-series method is evaluated for the CBCD task of TRECVID 2008. Among 2010 query videos, 1340 of them are originated from a reference video. CBCD evaluation software generates the analysis reports for each transformation type. The objective of the task is to detect all 134 copies with the correct reference video and temporal locations.

Our activity matching method consists of three parts: detecting full-frame copies, detecting copies of foreground videos generated with picture-in-picture transformation (T2), and flip transformation (T8-T9). We present the experimental results of these parts separately. Subsequence matching based on ordinal measure is taken as the baseline for our comparison. The results in terms of the number of correct detections for each transformation type are given in Table 4.4.

Table 4.4: Number of correct detections for activity subsequence matching method. Ordinal measure is taken as a baseline. Total number of copies that can be detected for each transformation type is 134.

	Baseline				**Proposed Method**
	Ordinal	Normal	Window	Flip	Normal+Window+Flip
T1	52	55	2	–	55
T2	2	2	36	–	37
T3	42	46	–	1	47
T4	74	76	–	–	76
T5	67	70	–	–	70
T6	73	74	–	–	74
T7	65	62	–	2	63
T8	11	17	1	22	40
T9	2	2	–	20	22
T10	14	16	4	7	27

It is seen from the results that considering the activities of picture-in-picture

transformation windows (for T2), and matching with the mirror of each query video (for T8 and T9) increase the accuracy of the proposed method.

4.7 Evaluation of Low-level Feature Matching

Non-facial shot matching with low-level visual features is the most successful matching part of the system for the transformations of text/logo insertion (T3), strong re-encoding (T4), and gamma change (T5) (see Table 4.5).

Table 4.5: Number of correct detections for low-level feature matching method. Total number of copies that can be detected for each transformation type is 134.

	T1	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**
Correct detections	51	1	97	113	114	102	64	75	37	27

Although the matching resulted in low correct detection rates for camcording (T1), picture-in-picture transformation (T2), and complex transformations (T9-T10); shot matching with low-level features has a huge efficiency for detecting video copies. We were able to represent ∼200 hours of reference videos with only 87,598 keyframes. *Approximate Nearest Neighbor* search on a *k-d* tree structure for this number of feature vectors runs very efficiently.

As a future work, visual features of picture-in-picture windows can be extracted and compared with reference keyframes in order to increase the accuracy of low-level feature matching method on transformations of T2. For improving the results for camcording transformation (T1), illumination-invariant features or similarity measures can be selected.

4.8 Overall Results

Number of correct detections for each method and the combined results are compared in Table 4.6. Note that there are some copies detected by more than one method. The overall correct detections are not the sum, but the union of the correctly detected query videos.

Table 4.6: Number of correct detections for each proposed method and their combination.

	Method 1 Matching Facial Shots	**Method 2** Activity Subsequence Matching	**Method 3** Low-level Feature Matching	**Combined Method** Correct	Hit	Miss
T1	18	55	51	82	61.20%	38.80%
T2	36	37	1	60	44.78%	55.22%
T3	48	47	97	113	84.33%	15.67%
T4	49	76	113	126	94.03%	5.97%
T5	54	70	114	127	94.78%	5.22%
T6	49	74	102	122	91.05%	8.95%
T7	51	63	64	103	76.87%	23.13%
T8	40	40	75	103	76.87%	23.13%
T9	32	22	37	70	52.24%	47.76%
T10	30	27	27	58	43.29%	56.71%
All	407	511	681	964	71.94%	28.06%

The results given in Table 4.6 show that our video copy detection framework achieves high correct detection rates for the transformations T3, T4, T5, and T6, mostly because of the frame-dropping detection, mask generation, noise detection, and border detection parts of the method. Similarly, a little complex transformations like T7 and T8 have very promising results.

Detecting copies of query videos with camcording (T1) and picture-in-picture transformation (T2) can be improved as a future work. Currently low-level feature matching method only works for the whole query frame; however, it can be

modified in a way that visual features of the picture-in-picture transformation window can be extracted and compared with the reference features.

The transformations of T9 and T10 are the most complex ones in this task; yet, our framework was able to detect about half of the copies. Coordinators of TRECVID states that "*CBCD task should investigate more realistic transformations by dropping very complicated transformations that are a combination of other transformations and found by this years result to be very difficult to detect*" in their report for TRECVID 2008 [13].

4.9 Comparisons with other groups in TRECVID 2008

We compare our results with the best 8 runs of the groups participated in CBCD task of TRECVID 2008. Three of the best results are submitted by INRIA-IMEDIA team [47]. *Joly* is a combination of dissociated dipoles features extraction [42] in sampled keyframes, features indexing and retrieval with distortion-based similarity search structure [44], and spatio-temporal registration of retrieved features. *ViCopT* performs a tracking of visual local features and index them differently according to some labels of behaviour [56], applies distortion-based similarity search structure directly on the local features extracted in keyframes [44], and uses a robust voting algorithm based on labels of behavior [54]. The run named *Joly+ViCopT* is the combination of two approaches, which is invariant to flip, resize, strong noise, and picture-in-picture transformation. INRIA-IMEDIA group was also responsible for the query video generation and CBCD evaluation software preparation in TRECVID 2008.

The method used by INRIA-LEAR [30] extracts SIFT features from keyframes, and by using bag-of-features approach and Hamming Embedding they generate image descriptors. The similarity scores between video clips are geometrically verified and the scores are aggregated to generate video segment matches. Orange Labs [34] uses visual features calculated around regions of interest, and

an adaptive and parameter-free method for scoring the matches. Tsinghua University with Intel China Research Center [57] propose a CBCD system that uses SURF descriptors [19] and ANN-based matching. Details of the CBCD systems used by IBM T.J. Watson Research Center and Columbia University were not published in the notebook papers of TRECVID 2008.

Our comparisons with other groups are based on the correct detection rate (CDR), and the total query processing time (QPT) for all of the 2010 query videos. Correct detection values of each transformation are calculated with the CBCD evaluation software. Total QPTs are computed from the run files for other groups.

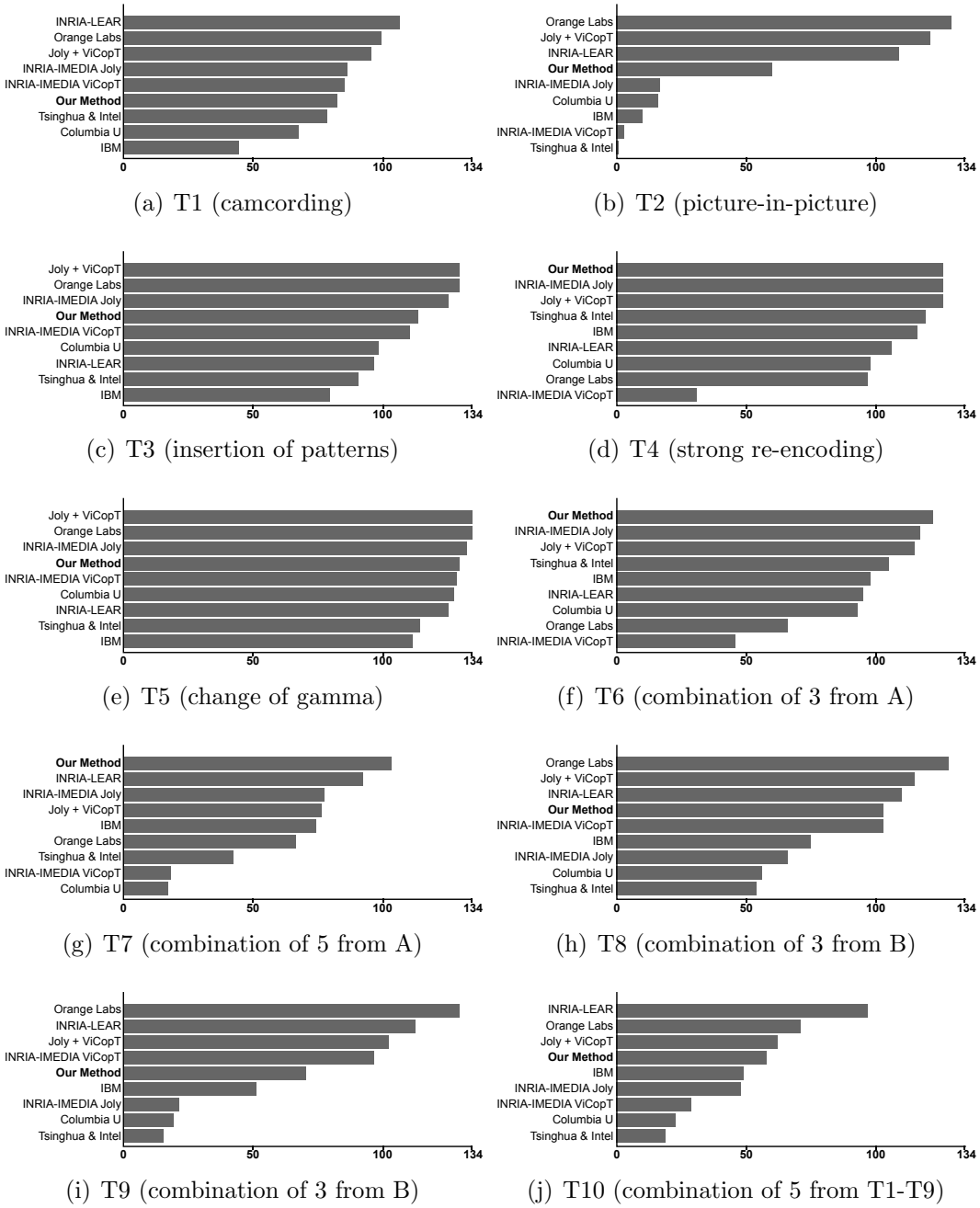
Because we implemented each part of the method separately (for evaluation and comparison purposes) rather than a complete copy detection system, our query processing time is estimated. Recall that our copy detection framework has a shot-boundary detection part (which also extracts masks for still regions, window of picture-in-picture transformation, detects noise and frame-dropping transformations) and three separate matching parts. In theory, while the shot-boundaries are detected, facial-shot matching and low-level feature matching parts can be run in parallel with the streaming outputs of shot-boundary detection. Activity subsequence matching part does not depend on the shot-boundaries; therefore, it can also work as a separate process.

To speed up the process, some techniques are applied to the methods. In facial shot detection, the method skips to the end of the shot when a face is detected and extracted. In activity subseqeunce matching, we employ a pruning-like method in order to discard reference subvideos with very low similarities. However, shot-boundary detection part should process each frame one-by-one, and run with 100fps rate. Therefore we estimate the total QPT based on the processing time of shot-boundary detector for 3,891,542 query frames, which can be completed in 648 minutes.

The correct detection rates and total query processing time of each run are compared in Table 4.7. The results of each transformation are sorted and visually presented in Figure 4.5.

Table 4.7: Correct detection of each transformation type for our method and the best 8 groups participated in TRECVID'08 CBCD task. *CDR:* correct detection rate, *QPT:* total query processing time in minutes.

	Joly	ViCopT	Joly+ViCopT	INRIA-LEAR	IBM	Columbia	OrangeLabs	Tsinghua&Intel	**Proposed**
T1	86	85	95	106	44	67	99	78	82
T2	17	3	121	109	10	16	129	–	60
T3	125	110	129	113	79	98	129	90	113
T4	126	31	126	106	116	29	97	119	126
T5	129	128	134	114	125	111	134	132	127
T6	115	66	117	105	98	46	95	93	122
T7	77	18	76	92	74	17	66	42	103
T8	66	103	115	110	75	56	128	54	103
T9	21	96	102	112	51	19	129	15	70
T10	48	29	62	97	49	23	71	19	58
Total	810	669	1077	1064	721	482	1077	642	964
CDR	60.44%	49.92%	80.37%	79.40%	53.80%	35.97%	80.37%	47.91%	71.94%
QPT	4037	571	9657	4112	696	14851	693	759	648



(a) T1 (camcording)

(b) T2 (picture-in-picture)

(c) T3 (insertion of patterns)

(d) T4 (strong re-encoding)

(e) T5 (change of gamma)

(f) T6 (combination of 3 from A)

(g) T7 (combination of 5 from A)

(h) T8 (combination of 3 from B)

(i) T9 (combination of 3 from B)

(j) T10 (combination of 5 from T1-T9)

Figure 4.5: Correct detection of each transformation type for our method and the best 8 groups participated in TRECVID'08 CBCD task.

Chapter 5

Conclusions

In this thesis, we first propose a fuzzy color histogram-based shot-boundary detection method for the videos where heavy transformations (such as cam-cording, insertions of patterns, strong re-encoding) occur. In addition to detecting shot-boundaries using fuzzy color histogram, we extract a mask for still regions and the window of picture-in-picture transformation. Experimental results show that the proposed method effectively detects shot boundaries with a small false alarm rate as compared to the state-of-the-art shot-boundary detection algorithms.

In the second part of the thesis, a multimodal framework for content-based copy detection and video similarity detection is presented. The proposed method consists of three steps for video segment matching: facial-shot matching, activity subsequence matching, and low-level feature matching. We were able to make a fair comparison by testing the method on the query and reference dataset of CBCD task of TRECVID 2008. Our results were compared with the results of top-8 most successful techniques submitted to this task. Experimental results show that the proposed method performs better than most of the state-of-the-art techniques, in terms of both effectiveness and efficiency. It is clear that the system already achieves high correct detection rates for the transformations of text/logo insertion, strong re-encoding, gamma change, and noise addition; however, there is still some potential for detecting queries with camcording and picture-in-picture transformations.

Our future extensions will focus on how to improve the effectiveness in transformations like camcording, picture-in-picture, and very complex ones. The results for camcorded query videos can be improved by translating the frames with the camera parameters calculated automatically from the video, if the video includes camcording transformation. For picture-in-picture transformations, we may need to improve picture-in-picture window extraction method, and consider these windows in low-level feature matching part. Moreover, we plan to give the evaluation results with a criterion that takes both misdetections and false-alarms into account.

Bibliography

[1] Final List of Transformations, Available at http://www-nlpir.nist.gov/projects/tv2008/active/copy.detection/final.cbcd.video.transformations.pdf, 2008.

[2] YouTube Copyright Policy: Video Identification tool, Available at http://www.google.com/support/youtube/bin/answer.py?answer=83766&hl=en-uk, 2009.

[3] ANN - Approximate Nearest Neighbor Searching by David M. Mount and Sunil Arya, available at http://www.cs.umd.edu/~mount/ann/, 2009.

[4] Building video queries for Trecvid 2008 copy detection task, Available at http://www-nlpir.nist.gov/projects/tv2008/trecvid2008copyqueries.pdf, 2009.

[5] Content-based copy detection, Guidelines for the TRECVID 2008 Evaluation, Available at http://www-nlpir.nist.gov/projects/tv2008/#4.5, 2009.

[6] Digital Information Growth Outpaces Projections, Despite Down Economy, Available at http://italy.emc.com/about/news/press/2009/20090518-01.htm, 2009.

[7] List of Colors, Wikipedia, Available at http://en.wikipedia.org/wiki/list_of_colors, 2009.

[8] List of Warner Bros. films, Available at http://en.wikipedia.org/wiki/list_of_warner_bros._films, 2009.

[9] MPEG-7 Overview, Available at `http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm`, 2009.

[10] Open Source Computer Vision Library, Available at `http://opencvlibrary.sourceforge.net`, 2009.

[11] The Diverse and Exploding Digital Universe, Available at `http://www.emc.com/collateral/analyst-reports/diverse-exploding-digital-universe.pdf`, 2009.

[12] The future of online video, Chad Hurley, Available at `http://googleblog.blogspot.com/2008/09/future-of-online-video.html`, 2009.

[13] TRECVID 2008 goals, tasks, data, evaluation mechanisms and metrics, available at `http://www-nlpir.nist.gov/projects/tvpubs/tv8.papers/tv8overview.pdf`, 2009.

[14] YouTube, Available at `http://www.youtube.com`, 2009.

[15] YouTube Video Identification Beta, Available at `http://www.youtube.com/t/video_id_about`, 2009.

[16] Zoinks! 20 Hours of Video Uploaded Every Minute!, Ryan Junee, Available at `http://www.youtube.com/blog?entry=on4emafa5ma`, 2009.

[17] E. Ardizzone, M. L. Cascia, A. Avanzato, and A. Bruna. Video indexing using mpeg motion compensation vectors. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems (ICMCS '99)*, page 725, Washington, DC, USA, 1999. IEEE Computer Society.

[18] A. Basharat, Y. Zhai, and M. Shah. Content based video matching using spatiotemporal volumes. *Computer Vision and Image Understanding*, Jan 2008.

[19] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[20] M. Bertini, A. D. Bimbo, and W. Nunziati. *Lecture Notes in Computer Science*, Jan 2006.

[21] G. Boccignone, A. Chianese, V. Moscato, and A. Picariello. Foveated shot detection for video segmentation. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(3):365–377, Mar 2005.

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

[23] C. Cai, K. Lam, and Z. Tan. An efficient scene-break detection method based on linear prediction with bayesian cost functions. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(9):1318–1323, Sep 2008.

[24] T. Can and P. Duygulu. Searching for repeated video sequences. In *Proceedings of the International Workshop on Multimedia Information Retrieval (MIR '07)*, Sep 2007.

[25] F. Chung and B. Fung. In *Proceedings of the 5th ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR '03)*, pages 157–162, Nov 2003.

[26] C. Cotsaces, N. Nikolaidis, and I. Pitas. Video shot detection and condensed representation. a review. *IEEE Signal Processing Magazine*, 23(2):28–37, Mar 2006.

[27] A. Dailianas, R. B. Allen, and P. England. Comparison of automatic video segmentation algorithms. In *Proceedings of SPIE, Integration Issues in Large Commercial Media Delivery Systems*, volume 2615, pages 2–16, 1995.

[28] T. Danisman and A. Alpkocak. Dokuz Eylül University Video Shot Boundary Detection at TRECVID 2006. In *Proceedings of the TREC Video Retrieval Evaluation (TRECVID), Available at* `http://www-nlpir.nist.gov/projects/tvpubs/tv6.papers/dokuz.pdf`, 2006.

[29] S. Das, S. Sural, and A. Majumdar. Detection of hard cuts and gradual transitions from video using fuzzy logic. *International Journal of Artificial Intelligence and Soft Computing*, 1(1):77–98, Jan 2008.

[30] M. Douze, A. Gaidon, H. Jegou, M. Marszalek, and C. Schmid. INRIA-LEARs video copy detection system. In *Proceedings of the TREC Video Retrieval Evaluation (TRECVID), Available at* `http://www-nlpir.nist.gov/projects/tvpubs/tv8.papers/inria-lear.pdf`, Nov 2008.

[31] L.-Y. Duan, J. Wang, Y. Zheng, J. Jin, H. Lu, and C. Xu. Segmentation, categorization, and identification of commercial clips from tv streams using multimodal analysis. In *Proceedings of the 14th annual ACM international conference on Multimedia (MULTIMEDIA '06)*, Oct 2006.

[32] R. O. Duda and P. E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of ACM*, 15(1):11–15, January 1972.

[33] H. Fang, J. Jiang, and Y. Feng. A fuzzy logic approach for detection of video shot boundaries. *Pattern Recognition*, 39(11):2092–2100, Jan 2006.

[34] N. Gengembre and S.-A. Berrani. The orange labs real time video copy detection system - trecvid 2008 results. In *Proceedings of the TREC Video Retrieval Evaluation (TRECVID), Available at* `http://www-nlpir.nist.gov/projects/tvpubs/tv8.papers/orangelabs.pdf`, Nov 2008.

[35] C. Grana, G. Tardini, and R. Cucchiara. MPEG-7 compliant shot detection in sport videos. In *Proceedings of the Seventh IEEE International Symposium on Multimedia*, Nov 2005.

[36] A. Hampapur and R. Bolle. Comparison of distance measures for video copy detection. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME '01)*, pages 737–740, Aug 2001.

[37] A. Hampapur, K. Hyun, and R. Bolle. Comparison of sequence matching techniques for video copy detection. In *Proceedings of the International Conference on Storage and Retrieval for Media Databases*, volume 4676, pages 194–201, Jan 2002.

[38] J. Han and K. K. Ma. Fuzzy color histogram and its use in color image retrieval. *IEEE Transactions on Image Processing*, 11(8):944–952, Aug 2002.

[39] W. Hsu and S. F. Chang. Topic tracking across broadcast news videos with visual duplicates and semantic concepts. In *Proceedings of the IEEE International Conference on Image Processing (ICIP '06)*, pages 141–144, Sep 2006.

[40] C. Huang, H. Lee, and C. Chen. Shot change detection via local keypoint matching. *IEEE Transactions on Multimedia*, 10(6):1097–1108, Oct 2008.

[41] R. Jadon, S. Chaudhury, and K. Biswas. A fuzzy theoretic approach for video segmentation using syntactic features. *Pattern Recognition Letters*, 22(13):1359–1369, Jan 2001.

[42] A. Joly. New local descriptors based on dissociated dipoles. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval (CIVR '07)*, pages 573–580, New York, NY, USA, 2007. ACM.

[43] A. Joly, O. Buisson, and C. Frélicot. Statistical similarity search applied to content-based video copy detection. In *Proceedings of the 21st International Conference on Data Engineering Workshops*, page 1285, Mar 2005.

[44] A. Joly, O. Buisson, and C. Frélicot. Content-based copy retrieval using distortion-based probabilistic similarity search. *IEEE Transactions on Multimedia*, 9(2):293–306, Feb 2007.

[45] A. Joly, C. Frélicot, and O. Buisson. Robust content-based video copy identification in a large reference database. In *Proceedings of ACM International Conference on Image and Video Retrieval (CIVR '03)*, pages 414–424, 2003.

[46] A. Joly, C. Frélicot, and O. Buisson. Content-based video copy detection in large databases: A local fingerprints statistical similarity search approach.

In *Proceedings of the IEEE International Conference on Image Processing (ICIP '05)*, volume 1, pages 505–508, Aug 2005.

[47] A. Joly, J. Law-to, and N. Boujemaa. INRIA-IMEDIA TRECVID 2008: Video copy detection. In *Proceedings of the TREC Video Retrieval Evaluation (TRECVID), Available at* `http://www-nlpir.nist.gov/projects/tvpubs/tv8.papers/inria-imedia.pdf`, Nov 2008.

[48] R. Kasturi and R. Jain. Dynamic vision. In R. Kasturi and R. Jain, editors, *Computer Vision: Principles*, pages 469–480. IEEE Computer Society Press, 1991.

[49] C. Kim and B. Vasudev. Spatiotemporal sequence matching for efficient video copy detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(1):127–132, Jan 2005.

[50] D. E. Knuth. *Fundamental Algorithms*, volume 2 of *The Art of Computer Programming*. Addison-Wesley, Reading, Massachusetts, second edition, 1973.

[51] K. Konstantinidis, A. Gasteratos, and I. Andreadis. Image retrieval based on fuzzy color histogram processing. *Optics Communications*, 248(4-6):375–386, Jan 2005.

[52] O. Kucuktunc, M. Bastan, U. Gudukbay, and O. Ulusoy. Bilkent University Multimedia Database Group at TRECVID 2008. In *Proceedings of the TREC Video Retrieval Evaluation (TRECVID), Available at* `http://www-nlpir.nist.gov/projects/tvpubs/tv8.papers/bilmdg.pdf`, Nov 2008.

[53] G. Langelaar, I. Setyawan, and R. Langedijk. Watermarking digital image and video data. a state-of-the-art overview. *IEEE Signal Processing Magazine*, 17(5):20–46, Sep 2000.

[54] J. Law-To, O. Buisson, V. Gouet-Brunet, and N. Boujemaa. Robust voting algorithm based on labels of behavior for video copy detection. In *Proceedings of MM'06*, Jan 2006.

[55] J. Law-To, L. Chen, A. Joly, I. Laptev, and O. Buisson. Video copy detection: a comparative study. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval (CIVR '07)*, Jan 2007.

[56] J. Law-To, V. Gouet-Brunet, O. Buisson, and N. Boujemaa. Local behaviours labelling for content based video copy detection. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR '06)*, volume 3, pages 232–235, Jan 2006.

[57] Y. Liang, X. Liu, Z. Wang, J. Li, B. Cao, Z. Cao, Z. Dai, Z. Guo, W. Li, L. Liu, Z. Meng, Y. Qin, Q. Shi, A. Tian, D. Wang, Q. Wang, C. Zhu, X. Hu, J. Yuan, P. Yuan, and B. Zhang. THU and ICRC at TRECVID 2008. In *Proceedings of the TREC Video Retrieval Evaluation (TRECVID), Available at* `http://www-nlpir.nist.gov/projects/tvpubs/tv8.papers/thu-icrc.pdf`, Nov 2008.

[58] R. Lienhart. Comparison of automatic shot boundary detection algorithms. In *Proceedings of SPIE*, volume 3656, pages 290–301, Jan 1999.

[59] R. Lienhart and J. Maydt. An extended set of haar-like features for rapid object detection. In *Proceedings of the International Conference on Image Processing (ICIP '02)*, volume 1, pages 900–903, 2002.

[60] A. Llorente, S. Zagorac, S. Little, R. Hu, A. Kumar, S. Shaik, X. Ma, and S. Rger. Semantic video annotation using background knowledge and similarity-based video retrieval. In *Proceedings of the TREC Video Retrieval Evaluation (TRECVID), Available at* `http://www-nlpir.nist.gov/projects/tvpubs/tv8.papers/ mmis.pdf`, Nov 2008.

[61] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[62] J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *Journal of the Optical Society of America A*, 7:923–932, 1990.

[63] E. H. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7(1):1–13, 1975.

[64] B. S. Manjunath, P. Salembier, and T. Sikora. *Introduction to MPEG-7, Multimedia Content Description Interface*. John Wiley and Sons, Ltd., Jun 2002.

[65] R. Mohan. Video sequence matching. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '98)*, volume 6, pages 3697–3700, May 1998.

[66] O. B. Orhan, J. Liu, J. Hochreiter, J. Poock, Q. Chen, A. Chabra, and M. Shah. University of Central Florida at TRECVID 2008 Content Based Copy Detection and Surveillance Event Detection. In *Proceedings of the TREC Video Retrieval Evaluation (TRECVID), Available at* `http://www-nlpir.nist.gov/projects/tvpubs/tv8.papers/ucf.pdf`, Nov 2008.

[67] S. Poullot, O. Buisson, and M. Crucianu. Z-grid-based probabilistic retrieval for scaling up content-based copy detection. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval (CIVR '07)*, Jul 2007.

[68] D. Pye, N. Hollinghurst, T. Mills, and K. Wood. Audio-visual segmentation for content-based retrieval. In *Proceedings of the Fifth International Conference on Spoken Language Processing*, Jan 1998.

[69] S. Satoh. News video analysis based on identical shot detection. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME '02)*, volume 1, pages 69–72, Jul 2002.

[70] S. Satoh, M. Takimoto, and J. Adachi. Scene duplicate detection from videos based on trajectories of feature points. In *Proceedings of the International Workshop on Multimedia Information Retrieval (MIR '07)*, Jan 2007.

[71] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and TRECVID. In *Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval (MIR '06)*, pages 321–330, New York, NY, USA, 2006. ACM.

[72] B. Truong, C. Dorai, and S. Venkatesh. New enhancements to cut, fade, and dissolve detection processes in video segmentation. In *Proceedings of the Eighth ACM International Conference on Multimedia (MULTIMEDIA '00)*, pages 219–227, Oct 2000.

[73] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01)*, volume 1, pages 511–518, 2001.

[74] B. Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4:419–420, 1962.

[75] X. Wu, A. G. Hauptmann, and C. W. Ngo. Measuring novelty and redundancy with multiple modalities in cross-lingual broadcast news. *Computer Vision and Image Understanding*, 110(3):418–431, Jan 2008.

[76] X. Wu, C. W. Ngo, and Q. Li. Threading and autodocumenting news videos: a promising solution to rapidly browse news topics. *IEEE Signal Processing Magazine*, 23(2):59–68, Mar 2006.

[77] X. Wu, M. Takimoto, S. Satoh, and J. Adachi. Scene duplicate detection based on the pattern of discontinuities in feature point trajectories. In *Proceedings of the 16th ACM international conference on Multimedia (MM '08)*, Oct 2008.

[78] M.-C. Yeh and K.-T. Cheng. Video copy detection by fast sequence matching. In *Proceedings of the ACM International Conference on Image and Video Retrieval (ACM CIVR '09)*, Apr 2009.

[79] J. Yuan, H. Wang, L. Xiao, W. Zheng, J. Li, F. Lin, and B. Zhang. A formal study of shot boundary detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(2):168–186, Feb 2007.

[80] L. Zadeh. Fuzzy sets. *Information Control*, 8:338–353, 1965.

[81] Y. Zhai and M. Shah. Tracking news stories across different sources. In *Proceedings of the 13th annual ACM international conference on Multimedia (MULTIMEDIA '05)*, Nov 2005.

[82] Y. Zhai and M. Shah. Tracking news stories across different sources. In *Proceedings of the 13th annual ACM international conference on Multimedia (MULTIMEDIA '05)*, pages 2–10, New York, NY, USA, 2005. ACM.

[83] W. L. Zhao, C. W. Ngo, H. K. Tan, and X. Wu. Near-duplicate keyframe identification with interest point matching and pattern learning. *IEEE Transactions on Multimedia*, 9(5):1037–1048, Aug 2007.