Video shot boundary detection and key-frame extraction using mathematical models

Youssef Bendraou

Thèse de Doctorat
en cotutelle

Présentée par

Youssef BENDRAOU

Titre

Détection des changements de plans et extraction d'images représentatives dans une séquence vidéo

Discipline : Sciences de l'ingénieur
Spécialité : Informatique et Télécommunications

Soutenue le 16/11/2017, devant le jury composé de

Président :	
Mohamed NAJIM	Professeur à l'ENSEIRB, Bordeaux
Rapporteurs :	
El Mustapha MOUADDIB	Professeur à l'Université de Picardie Jules Verne, Amiens
Guillaume LAVOUE	Professeur à l'INSA, Lyon
Examinateur:	
Mohamed RZIZA	PH à la Faculté des Sciences, Université Mohamed V, Rabat
Invitée:	
Fedwa ESSANNOUNI	Membre attachée au LRIT, Faculté des Sciences de Rabat
Co-directeur de thèse :	
Rachid OULAD HAJ THAMI	Professeur à l'ENSIAS, Rabat
Directeur de thèse :	
Ahmed SALAM	Professeur à l'Université Littoral Côte d'Opale, Calais

Faculté des Sciences, 4 Avenue Ibn Battouta B.P. 1014 RP, Rabat – Maroc
Tel +212 (0) 37 77 18 34/35/38, Fax: +212 (0) 37 77 42 61, http://www.fsr.ac.ma

Université Mohammed V de Rabat
Université du Littoral Côte d'Opale

Laboratoire de Recherche en Informatique et Télécommunications (LRIT)
Laboratoire de Mathématiques Pures et Appliquées J. Liouville (LMPA)

Video shot boundary detection and key-frame extraction using mathematical models

by Youssef BENDRAOU

Video shot boundary detection and key-frame extraction using mathematical models

Youssef BENDRAOU

September 21, 2017

Avant Propos

Les travaux présentés dans ce mémoire ont été effectués au sein du Laboratoire de Recherche en Informatique et Télécommunications LRIT, Faculté des Sciences de Rabat. Cette thèse a été réalisée dans le cadre d'une cotutelle entre l'Université Mohamed V et l'Université du Littoral Côte d'Opale au sein du Laboratoire de Mathématiques Pures et Appliquées LMPA. Ce travail a été réalisé sous la direction du Pr. Rachid OULAD HAJ THAMI et du Pr. Ahmed SALAM et l'encadrement du Pr. Fedwa ESSANNOUNI.

Je tiens tout d'abord à rendre un grand hommage à mon cher professeur et ancien directeur de thèse, Mr. Driss ABOUTAJDINE, qui est décédé récemment. Sans lui, je n'en serais pas là aujourd'hui. Que votre âme repose en paix.

Je tiens également à remercier Mr. Rachid OULAD HAJ THAMI, Professeur de l'enseignement supérieur à l'École Nationale Supérieure d'Informatique et d'Analyse des Systèmes, pour avoir accepté d'être mon directeur de thèse. Son aide m'a été très précieuse. Je remercie également mon co-directeur de thèse Mr. Ahmed Salam, Professeur à l'Université du Littoral Côte d'Opale, de m'avoir donné l'opportunitée d'intégrer son laboratoire de recherche LMPA. Ses observations perspicaces ainsi que ces questions pertinentes m'ont toujours conduit à approfondir mes études dans différents domaines. Sans son soutien précieux, cette recherche n'aurait pas eu lieu.

Je voudrais aussi exprimer ma profonde gratitude à mon encadrante, M. Fedwa ESSANNOUNI, Docteur de la Faculté des Sciences de Rabat, pour son soutien continu durant toute cette thèse, sa motivation, sa patiente, ainsi que son grand savoir. Jamais je n'aurais pu imaginer avoir un meilleur encadrant.

Je remercie Mr. Mohamed NAJIM, Professeur à l'École Nationale Supérieure d'Electronique, Informatique, Télécommunications, Mathématiques et Mécanique de Bordeaux, pour l'honneur qu'il me fait en présidant le jury de ma thèse.

Je remercie également Mr. El Mustapha MOUADDIB, Professeur à l'Université de Picardie Jules Verne Amiens, d'avoir accepté de juger la qualité de mon travail en tant que rapporteur. Ensuite, je remercie Guillaume LAVOUE, Professeur à Institut National des Sciences Appliquées de Lyon, pour avoir accepté de rapporter ce travail et de participer au jury. Je remercie aussi Mr. Mohamed DAOUDI, Professeur à l'Institut Mines-Télécom Lille Douai, d'avoir accepté d'examiner ce travail de thèse. Finalement, je remercie Mr. Mohamed RZIZA, Professeur Habilité à la Faculté des Sciences de Rabat, qui a accepté d'examiner mon travail de thèse.

Acknowledgements

First and foremost, I thank God for the good health and wellbeing that were necessary to complete this thesis.

I would like to pay a great tribute to my dear professor and thesis director, Mr. **Driss Aboutajdine**, who passed away recently. Without him, I would not have been able to get there, Thank You Professor. May your soul rest in peace. I would like also to thank the professor Mr. **Rachid Oulad Haj Thami** for having accepted to be my thesis supervisor.

I would like to express my sincere gratitude to my advisor Prof. **Fedwa Essannouni** for the continuous support, for her patience, motivation, and immense knowledge. Her guidance and advice helped me throughout the research and the writing of this thesis. I could not have imagined having a better mentor and advisor. Thanks also go to my thesis supervisor Prof. **Ahmed Salam** who provided me the opportunity to join their team in the LMPA laboratory. His insightful questions, relevant comments and wise advices allowed me to improve my knowledge in various fields. Without his precious support it would not be possible to conduct this research.

I take this opportunity to express gratitude to all of the department faculty members: University Mohammed V (UMV) and University of Littoral Cote Opale (ULCO) for their help and support. Their contributions were very valuable and essential to the success of this project.

I would like also to thank close friends for accepting nothing less than excellence from me. Their words of encouragement in case of despair, as well as their constructive ideas, whatever they were far from my field of research, were necessary and of immense assistance to me.

Last but not least, I would like to express a special thank to my family for everything: my parents for the unceasing encouragement, support and attention, to my brother and sisters for supporting me throughout writing this thesis and in my life in general.

Abstract

With the recent advancement in multimedia technologies, in conjunction with the rapid increase of the volume of digital video data and the growth of internet; it has become mandatory to have the ability to hastily browse and search through information stored in large multimedia databases. For this purpose, content based video retrieval (CBVR) has become an active area of research during the last decade. The objective of this thesis is to present applications for temporal video segmentation and video retrieval based on different mathematical models.

A shot is considered as the elementary unit of a video, and is defined as a continuous sequence of frames taken from a single camera, representing an action during time. The different types of transitions that may occur in a video sequence are categorized into: abrupt and gradual transition. In this work, through statistical analysis, we segment a video into its constituent units. This is achieved by identifying transitions between adjacent shots. The first proposed algorithm aims to detect abrupt shot transitions only by measuring the similarity between consecutive frames. Given the size of the vector containing distances, it can be modeled by a log normal distribution since all the values are positive.

Gradual shot transition identification is a more difficult task when compared to cut detection. Generally, a gradual transition may share similar characteristics as a dynamic segment with camera or object motion. In this work, singular value decomposition (SVD) is performed to project features from the spatial domain to the singular space. Resulting features are reduced and more refined, which makes the remaining tasks easier. The proposed system, designed for detecting both abrupt and gradual transitions, has lead to reliable performances achieving high detection rates. In addition, the acceptable computational time allows to process in real time.

Once a video is partitioned into its elementary units, high-level applications can be processed, such as the key-frame extraction. Selecting representative frames from each shot to form a storyboard is considered as a static and local video summarization. In our research, we opted for a global method based on local extraction. Using refined centrist features from the singular space, we select representative frames using modified k-means clustering based on important scenes. This leads to catch pertinent frames without redundancy in the final storyboard.

Résumé

Les technologies multimédias ont récemment connues une grande évolution surtout avec la croissance rapide d'Internet ainsi que la création quotidienne de grands volumes de données vidéos. Tout ceci nécessite de nouvelles méthodes performantes permettant d'indexer, de naviguer, de rechercher et de consulter les informations stockées dans de grandes bases de données multimédia. La récupération de données basée sur le contenu vidéo, qui est devenue un domaine de recherche très actif durant cette décennie, regroupe les différentes techniques conçues pour le traitement de la vidéo.

Dans le cadre de cette thèse de doctorat, nous présentons des applications permettant la segmentation temporelle d'une vidéo ainsi que la récupération d'information pertinente dans une séquence vidéo. Une fois le processus de classification effectué, il devient possible de rechercher l'information utile en ajoutant de nouveaux critères, et aussi de visualiser l'information d'une manière appropriée permettant d'optimiser le temps et la mémoire.

Dans une séquence vidéo, le plan est considéré comme l'unité élémentaire de la vidéo. Un plan est défini comme une suite d'image capturée par une mÃłme caméra représentant une action dans le temps. Pour composer une vidéo, plusieurs plans sont regroupés en utilisant des séquences de transitions. Ces transitions se catégorisent en transitions brusques et transitions progressives.

Détecter les transitions présentes dans une séquence vidéo a fait l'objet de nos premières recherches. Plusieurs techniques, basées sur différents modèles mathématiques, ont été élaborées pour la détection des changements de plans. L'utilisation de la décomposition en valeur singulière (SVD) ainsi que la norme Frobenius ont permis d'obtenir des résultats précis en un temps de calcul réduit.

Le résumé automatique des séquences vidéo est actuellement un sujet d'une très grande actualité. Comme son nom l'indique, il s'agit d'une version courte de la vidéo qui doit contenir l'essentiel de l'information, tout en étant le plus concis possible. Ils existent deux grandes familles de résumé : Le résumé statique et le résumé dynamique. Sélectionner une image représentative de chaque plan permet de créer un scénarimage. Ceci est considéré comme étant un résumé statique et local. Dans notre travail, une méthode de résumé globale est proposée.

Contents

List of Figures

List of Tables

List of Acronyms

CBVR	**C**ontent **B**ased **V**ideo **R**etrieval
CBVIR	**C**ontent **B**ased **V**ideo **I**indexing and **R**etrieval
CT	**C**ut **T**ransition
CUS	**C**omparison of **U**ser **Summaries**
DCT	**D**iscrete **C**osine **T**ransform
ECR	**E**dge **C**hange **R**atio
FPS	**F**rames **P**er **S**econd
GT	**G**radual **T**ransition
MSR	**M**inimum **S**parse **R**econstruction
OVP	**O**pen **V**ideo **P**roject
SBD	**S**hot **B**oundary **D**etection
SD	**S**parse **D**ictionary
SVD	**S**ingular **V**alue **D**ecomposition
VS	**V**ideo **S**ummary
WHT	**W**alsh **H**adamard **T**ransform

Dedicated to my parents and supervisors.

Chapter 1

Introduction

1.1 Motivations, problematic and objectives

With the exponential increase of video data and the daily creation of a large number of digital videos, in conjunction with the recent advances in multimedia and the rapid growth of internet, the field of video indexing and retrieval is becoming an active area of research. Manual editing and indexing is the most accurate way but a time consuming task. Consequently, automatic video analysis applications are required for representing, modeling, indexing, retrieving, browsing or searching through information stored in large multimedia data. Such techniques are grouped into a single concept of Content-Based Video Indexing and Retrieval (CBVIR) systems. The available information from videos may include visual content, audio information and video metadata. In this thesis, we are concerned only by the visual contents.

A video is generally structured into frames, shots, sub shots or scenes as illustrated in Figure 1.1. Before performing any kind of processing, usually, the first step of CBVIR systems is to segment a video into its main components. A shot is considered as the elementary unit of a video, and is defined as a continuous sequence of frames taken from a single camera, representing an action during the time. Detecting shots in a video is known as the shot boundary detection (SBD) problem. A sub-shot occurs when the visual content of the current shot changes dramatically. A scene is defined as set of decor that represents the place of the action (i.e. beach, forest, countryside, building, etc). In a video sequence, a scene may include several shots as it can seen in Figure 1.1, where shots 1 et 3 belong to scene A. Detecting sub-shots and scenes are referred to micro-segmentation and macro-segmentation, respectively. Generally both fields need a shot boundary detection first. Different types of transitions are added between shots to form a video sequence. Segmenting a video into shots is equivalent to detecting the transitions between shots. Once a video is classified into shots, further applications can be performed. Usually considered as a first step in CBVIR, SBD is a crucial step towards

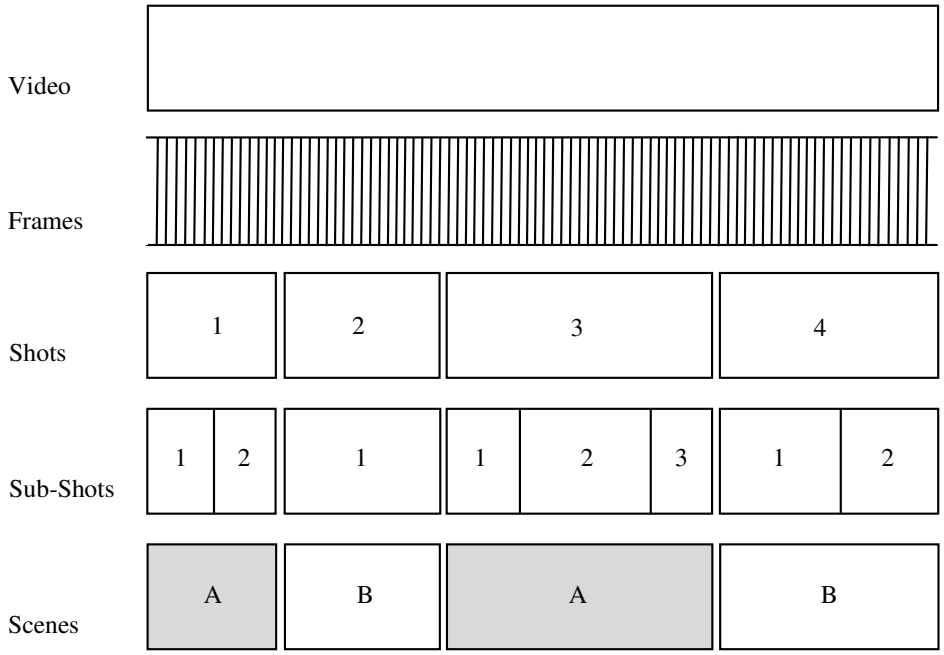


FIGURE 1.1: Hierarchical structure of a video

subsequent high-level applications. For example, a wrong shot classification will negatively affect the expected results of the key-frame extraction.

Video analysis and retrieval are challenging tasks due to the variety of video types and the several transitions and special effects that can be added. In addition, various other factors can represent a major challenging to SBD and CBVIR applications. In shot boundary detection, the various illumination changes that may occur in a scene, the fast object or camera motions and the special effects may lead to error detections. A robust SBD method should perform good detections for all types of transitions for any arbitrary video sequence with minimized manual predefined parametrization [3]. Different techniques have been developed in the past with reliable performances. In the present thesis, the shot boundary detection problem is addressed in detail, dealing with methods from the firsts to the recent ones. This allowed to analyze and operate several ideas previously considered, leading to a better identification of the problematic. As a result, different approaches based on multiple mathematical models were implemented to solve the SBD problem successfully. First, the sum of absolute histogram differences between consecutive frames are calculated to constructed the continuity signal. A thresholding selection, based on statistical analysis, gives a set of candidate frames for which different post processing are performed to eliminate false detections. Experimental results have shown the good performance of the proposed method which is able to recover all shot cuts, thus reaching a percentage of 100% for the recall criterion (i.e. no missed shot).

Another solution using the singular value decomposition (SVD) is proposed. Different theorems provided by the SVD are explored for detecting both hard and gradual transitions. In our contribution, the Frobenius norm is used to estimate the best low rank approximation from the singular value decomposition of concatenated block based histograms (CBBH). Each frame will be mapped into k-dimensional vector in the singular space according to each segment. The classification of continuity values is achieved via double thresholding technique for detecting the hard cuts. The folding-in technique, also known as SVD-updating, is then achieved for the first time to detect the gradual transitions. This allows an accurate detection in a very reduced computation time, since there is no need to recalculate the SVD decomposition for segment correction. Various simulations and tests were carried out on different video databases related to annual TRECVid evaluation datasets. More details about experimental results are described later in the report. Once a video is segmented into shots, a direct application is the key-frame extraction. For this purpose, we present a last method which returns a storyboard of a given video sequence.

1.2 Thesis structure

In Chapter 2, the shot boundary detection and video abstraction issues are introduced and a literature review of existing methods is presented. Our solution for cut detection, via statistical analysis, is discussed in Chapter 3. Chapter 4 reviews the proposed modeling for gradual shot transition, which is a more difficult task. The singular value decomposition is used in our system to present an efficient solution in detecting different types of transitions. Various simulations and tests were carried out over a benchmark of video datasets to prove the efficiency of the proposed approach. Chapter 5 presents our idea for video summarization based on important scenes. Different implementations and experiments have shown the good performance of our framework. The last chapter concludes this thesis and presents some perspectives for future works. Figure 1.2 gives an overview of the thesis structure.

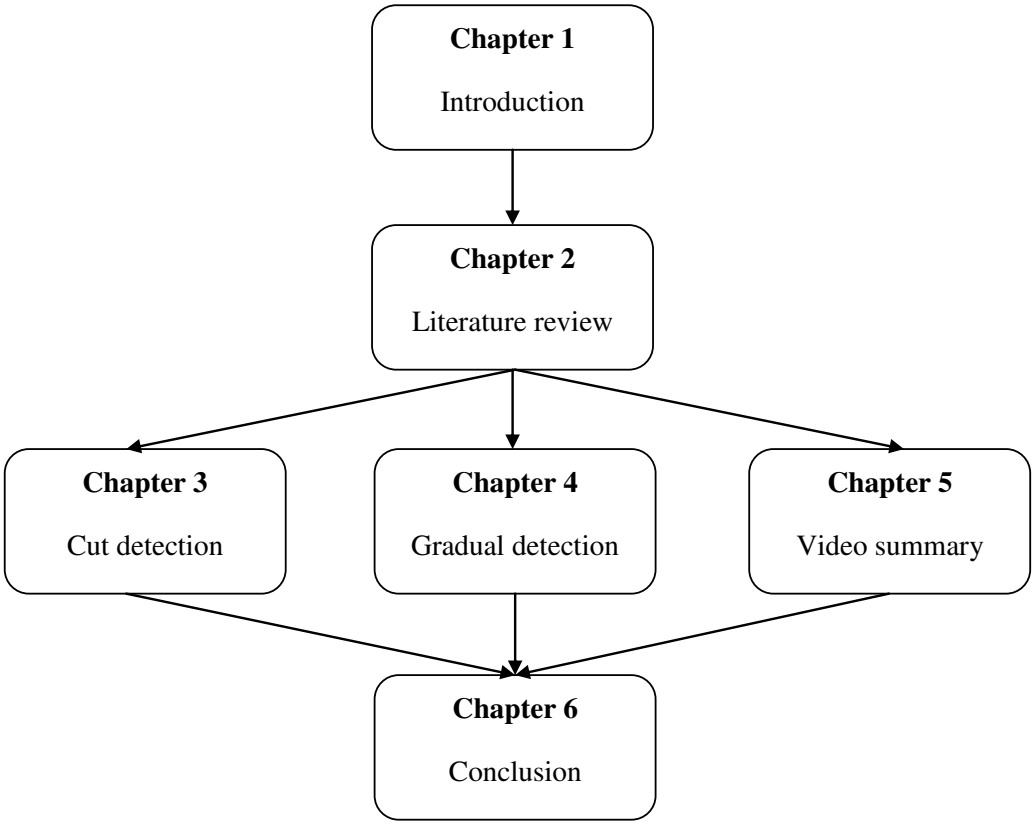


FIGURE 1.2: Highlights of the present thesis.

1.3 List of publications

The present work has led to the following publications:

International journals

- Y. Bendraou, F. Essannouni, A. Salam and D. Aboutajdine, Video cut detection method based on a 2D luminance histogram using an appropriate threshold and a post processing, *WSEAS Transactions on Signal Processing*, pp: 99-106, vol. 11, 2015.
- Y. Bendraou, F. Essannouni, A. Salam and D. Aboutajdine, Shot boundary detection via adaptive low rank and svd-updating, *Computer Vision and Image Understanding* (CVIU) Elseiver, pp: 20-28, vol. 161, 2017.
- Y. Bendraou, F. Essannouni and A. Salam, From local to global video summary using singular values decomposition of centrist features, Submitted to *Multimedia Tools and Applications*.

International conferences

- Y. Bendraou, F. Essannouni, A. Salam and D. Aboutajdine, Video shot boundary detection method using histogram differences and local image descriptor, *2nd World Conference on Complex Systems*, (WCCS'14), pp: 665-670, Agadir, Morocco, November 10-12, 2014.
- Y. Bendraou, F. Essannouni, A. Salam and D. Aboutajdine, Video Cut Detector via Adaptive Features using the Frobenius Norm, *12th International Symposium on Visual Computing*, (ISVC'16), LNCS in Advances in Visual Computing, Part II (2) pp: 380-389, Las Vegas, NV, USA, December 12-14, 2016.

National workshops and events

- Y. Bendraou, F. Essannouni, A. Salam and D. Aboutajdine, Video cut detection: literature review and statistical analysis, *JDTIC*
- Y. Bendraou, F. Essannouni, A. Salam and D. Aboutajdine, Cut detection using adaptive threshold, *URAC*
- Participation in the summer school on Levy process. University of Lille, July 2016.

Chapter 2

Literature review

In this chapter, we present a literature review of two main research areas in content based video retrieval (CBVR). Firstly, we list various methods of video shot boundary detection (SBD) including abrupt and gradual transitions. Secondly, we discuss several methods of video summarization (VS) and their different classifications in the literature. In the present, methods discussed are listed from the beginning and arranged by decades.

2.1 Video shot boundary detection

Algorithms of shot boundary detection (SBD) have an old and rich history in automatic video analysis. Different techniques have been developed in the literature, during these decades, with reliable performances. The aim of such methods is to segment a video sequence into its elementary units: shots. A video shot is defined as a continuous sequence of frames taken from a single camera and representing an action over time. As they represent the elementary unit which produces a video, shots are usually considered to be the primitives for higher level content based video retrieval and video analysis. According to the literature [4, 5, 6, 7, 8] the transitions between shots can be classified in two types : It may be an abrupt or a gradual transition as shown in Figure 2.1. An abrupt shot transition, also called hard cut or cut, is a sudden change from a video shot to another one. In the following of this report, we note a cut by CT. The difficulties in CT detection are the camera and objects motion, lighting variations and special effects. The second type, which is the gradual shot transition [4, 9, 10], occurs when the transition between two shots is accomplished progressively over several frames. Even if there exists different types of gradual transitions such as the fade in, fade out, the dissolve and the wipe; we refer all of them by GT. Later in the present document, each case will be explained independently. The GT detection is a more complicated task than the CT detection due to their

FIGURE 2.1: Different type of transitions, including hard cut and gradual transitions

diversity and several additional difficulties. More details are addressed in section 2.1.2. Various types of GT transitions including fades, dissolves and wipes are illustrated in Figure 2.1.

Generally, the basis of any SBD method consists of measuring the visual discontinuities between successive frames to detect the video shot transitions. Practically, it involves three main steps: extracting features from video frames, constructing a continuity signal and classifying video segments. Most of existing approaches [6, 7, 11] use a similarity measure from adjacent frame features, where, if this similarity is higher than a predefined threshold, a shot boundary is detected. Previous studies confirmed that there is two broadly approaches for selecting the frame features, which are the pixel domain processing and the compressed domain. Comparison between different methods showed that those who operate in the pixel domain are more accurate compared to methods in the compressed domain which are faster. Various other classifications of SBD techniques were proposed, where different metrics are used for different features. Similarly, classification techniques can be divided into those using a threshold (global, local or adaptive) and those based on machine learning. The work presented in [12] was subject of a survey and has proposed a formal study with a novel classification. Authors, first, identify and list techniques of visual content representation, then, techniques of constructing a continuity signal and finally algorithms of classification and decision. Detailed comparisons and discussions of SBD techniques were studied during these decades, resulting in excellent surveys [5, 10, 13, 3, 12]. In this section, a detailed study of related works on scene change detection is addressed. In our work, most of existing methods are listed from the first ones, using

classical features and similarities, to the recent ones, based on more complex features or using machine learning for decision. In our classification, we first discuss CT detection solutions followed by the GT transitions systems. We favor this categorization since the cut detection is considered as a solved problem, contrary to the gradual identification, which is a still open research area. Following the analysis of advantages and disadvantages of each approach, we propose other solutions to the SBD problem, detailed in next chapters.

2.1.1 Cut transition identification

Early works using classical features and metrics

In the first proposed techniques, generally, we found as visual features: pixel intensities, histograms or edges. The discrete cosine transform (DCT) was also used in many early studies. Although compressed features (i.e., DCT and variants) have proven to be fast, their use has decreased due to their unsatisfactory results, unlike conventional features where several alternatives were proposed. One of the first metrics that have been used, we can list the pixel wise comparison [5] which evaluate the differences in the intensity values of corresponding pixels in two consecutive frames. The easiest way to detect if two frames are different is to calculate the sum of absolute pixel differences (SAD) using 2.1 and compare it against a threshold.

$$D(I_k, I_{k+1}) = \sum_{i,j}^{N} |I_k(i,j) - I_{k+1}(i,j)| \,, \tag{2.1}$$

where $I_k(i,j)$ denotes the intensity pixel in the (i,j) location at the k^{th} frame and N the number of pixels. Besides using the SAD which is equivalent to the Manhattan distance, one can use several other distances as similarity measure such as:

- The sum of squared differences (SSD), also known as Euclidean norm:

$$D(I_k, I_{k+1}) = \sum_{i,j}^{N} (I_k(i,j) - I_{k+1}(i,j))^2 \,. \tag{2.2}$$

- The mean absolute error (MAE) is a normalized version of the SAD:

$$D(I_k, I_{k+1}) = \frac{1}{N} \sum_{i,j}^{N} |I_k(i,j) - I_{k+1}(i,j)| \,. \tag{2.3}$$

- The mean squared error (MSE) is a normalized version of the SSD:

$$D(I_k, I_{k+1}) = \frac{1}{N} \sum_{i,j}^{N} \left(I_k(i,j) - I_{k+1}(i,j)\right)^2 . \tag{2.4}$$

- The Euclidean distance which is the natural distance in a geometric interpretation:

$$D(I_k, I_{k+1}) = \sqrt{\sum_{i,j}^{N} \left(I_k(i,j) - I_{k+1}(i,j)\right)^2}. \tag{2.5}$$

- A weighted version of the Manhattan distance, called The Canberra distance and defined by:

$$D(I_k, I_{k+1}) = \sum_{i,j}^{N} \frac{|I_k(i,j) - I_{k+1}(i,j)|}{|I_k(i,j)| + |I_{k+1}(i,j)|}. \tag{2.6}$$

- The cosine distance which represents the angular distance of two vectors. Let I_k be defined by a row vector h_k, then it can be written as:

$$D(I_k, I_{k+1}) = 1 - \frac{h_k * h'_{k+1}}{\|h_k\|_2 \, \|h_{k+1}\|_2}. \tag{2.7}$$

- The Pearson distance based on the correlation coefficient ρ between the two vectors h_k and h_{k+1}:

$$D(I_k, I_{k+1}) = 1 - \rho(h_k, h_{k+1}). \tag{2.8}$$

The main drawback of such approaches (i.e. intensity pixels), whatever the metric used, is that they are unable to differentiate between a large change in a small area and a smaller change in a large area. Obviously, pixel-based techniques are the most sensitive to surrounding disturbances (i.e. noises, illumination changes) that may interfere with a given scene. Several variants of pixel-based methods have been proposed [14, 15, 16] to reduce the motion influence. In [16], a frame is divided into 12 regions in a 4×3 pattern. A block-matching process using a 24×18 search window is applied to generate a set of block match values based on motion vectors. This allows for each region to find its best fitting region in the next frame. The highest and lowest match values are discarded and the remaining values are averaged to produce a global match value. A cut is declared when the global match value exceeds a given threshold. Despite these improvements, pixel based comparisons still remain sensitive to camera and object motion. For example, a camera pan will produce a significant change to the majority of pixels. Among the early proposed works, the one elaborated by Nagasaka and Tanaka [17] in 1991. Their study

consists in testing various features and measures. The normalized χ^2 test was selected as the best measure to calculate the distance between two histograms. To reduce noises and camera flashes effects, frames are divided into sub frames and each pair of sub frames of consecutive frames are compared. The largest distances are discarded, and the decision is made by measuring the differences of remaining subframes. Also in 1991, Ueda et al. [18] propose a system based on the correlation of color between two adjacent frames in a motion picture. Encountered difficulties were the camera zooming or panning and object motion, particularly when a large object moves across the frame. Such situations may cause big changes in the correlation. The problem was solved by judging the cut change using the rate of correlation change instead of simply using the magnitude of correlation change. Similarly to [17] a frame is divided into regions, where each one is used as a template, then the best matched region is sought in the following frame. The amount of movement in the small region is calculated from the distance between the two regions. Those (i.e. regions) without sufficient information or representing homogeneous surfaces are eliminated prior to the matching process.

In 1993, Arman et al. [19] were the first to introduce the DCT coefficients as features to perform scene change detection. Authors use the already encoded information in the JPEG encoded video form. Only a subset of blocks is considered to form a representative vector for each frame in the DCT space. This allows to significantly saving the time processing as the cost of decompressing would not be considered. The inner product of consecutive vectors is then calculated and compared to a global threshold to declare the cuts. Although their approach is fast, it has several limitations. To avoid misclassification, further refinement is considered using histogram difference in the HSI color space. In 1995, several other methods based on MPEG compressed domain were proposed [20, 21, 22, 23, 14]. In the present work, we briefly give an overview of MPEG compressed video. For more details about the concept of DC-images, DC-sequences and how to extract them from compressed videos, please refer to [14, 24]. A structural hierarchy of an MPEG video and the cut detection problem in a compressed video are addressed in [24]. Authors illustrate various feature extraction steps in the compressed domain and show how to compute mean, variance and region histograms directly from the compressed video. This allows implementing most of existing cut detection methods, developed for uncompressed video, directly on the compressed video. Actually, three temporally interleaved images are found in the MPEG bitstream, called picture types or frame types: 1) I-frames, which are the intra coded pictures. Since all macroblocks are coded without prediction, I-frames serve as a starting point for incoming predictions. 2) P-frames are the predicted pictures, where macroblocks may be coded with forward prediction from references

made from previous I and P pictures. 3) B-frames represent the Bi-directionally predicted pictures. Macroblocks may be coded with interpolated (forward or backward) prediction from past and future I or P references. All these frames are grouped into a structure called group of pictures (GOP). Usually, a GOP begins and ends with an I-frame followed by a number of P and B frames. In [20], authors use the variance of DC-coefficients in I and P frames and motion vectors information to characterize scene changes. In another way, [22] use only the DC-coefficients of the I-frame and address the SBD as a statistical hypothesis testing problem using luminance histogram. Three tests to determine cut locations are presented, however the exact location of CTs cannot be performed with this technique [14]. Alternatively, Liu et al. [21] make use of only parameters encoded in P and B pictures to detect scene changes. Zhang et al. [23] assume that the number of valid motion vectors in P or B frames tend to be low in the presence of a shot boundary. Their algorithm is an hybrid approach which integrates both the video content encoded in DCT-coefficients and the motion vectors between frames. With more than 1000 citations, [14] can be considered as the most representative system which analysis several algorithms for detecting scene changes on compressed video. By performing minimal decoding on the compressed bitstream without full-frame decompression, authors assume that only the essential information is retained. Obviously, no one can disclaim that operating in the compressed domain offers significant time and cost savings. At the same time, SBD methods and solutions in the pixel domain are not lacking and their numbers increase in powers. In 1995, Zabih et al. [9] introduce a popular alternative solution by using edges as visual features for representing frames. The method can detect and classify a variety of scene changes including cuts, fades and dissolves. However, in this section, we only address the CT detection problem. Detailed review of GT systems is presented in the next section. The authors notice that when a transition occurs, new edges appear, in incoming frames, far from the location of old edges. Similarly, edges disappear from current frames. The first ones are called entering edge pixels and denoted ρ_{in} and the later, called exiting edge pixels, are denoted by ρ_{out}. A high value of ρ_{in} may represent a cut, a fade in or the end of a dissolve, while a high value of ρ_{out} may assume a cut, a fade out or the beginning of a dissolve. Based on this, their measure of dissimilarity, called edge change ratio (ECR), is defined as follows:

$$ECR(k) = max(\frac{\rho_{in}^{k}}{\sigma_k}, \frac{\rho_{out}^{k-1}}{\sigma_{k-1}}) \tag{2.9}$$

where ρ_{in}^{k} and ρ_{out}^{k-1} represent respectively the number of incoming and outgoing edge pixels in frame k and $k-1$. σ_k is the number of edge pixels in frame k. If the obtained ECR is greater

than a Threshold T, a cut is detected. This method provides a large number of false detection when a high-speed motion occurs in the video scenes. However, edge-based methods [25, 26] are relatively more robust against camera motion and can detect both hard cuts and gradual transitions. Their major drawback resides in their high computationally cost. Zhang et al. [27] presented a comprehensive study of existing techniques. Their work is subject of a comparison between a set of different features including the pixel wise comparison, the likelihood ratio and the histogram differences. Various metrics and measures are tested to detect what they called the camera breaks (i.e. CT detection). A motion analysis algorithm was applied to eliminate false interpretation of camera motion. A multi-pass approach to improve both the accuracy and the time processing have also been developed. The pair wise comparison was formulated as a binary function, which count the number of pixels that changed from two consecutive frames according to the metric:

$$DP_i(x,y) = \begin{cases} 1 & \text{if } |P_i(x,y) - P_{i+1}(x,y)| \geq t \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

A CT is declared if large number of pixels have changed. This metric is very sensitive to camera motion, which can be reduced using a smoothing filter. In addition, instead of comparing individual pixels, one can compare corresponding blocks or regions from two frames using the mean and the variance of intensity values. This is called the likelihood ratio and is defined by:

$$\frac{\left(\Sigma_i^t + \Gamma_i^t\right)^2}{\sigma_i^t \times \sigma_i^{t+1}} > t \tag{2.11}$$

with,

$$\Sigma_i^t = \frac{\sigma_i^t + \sigma_i^{t+1}}{2} \quad \& \quad \Gamma_i^t = \left(\frac{\mu_i^t + \mu_i^{t+1}}{2}\right)^2 \tag{2.12}$$

where μ_i^t and σ_i^t are respectively the mean and the variance of the i^{th} block in the t^{th} frame. The drawback of this metric is that two regions with a completely different visual content may have the same mean and variance, and this will lead to missed shots. Their comparison also involves histogram differences in both grey level and color components. As measure for similarities, the sum of absolute differences and the χ^2-test [17] were implemented. The study confirmed that histogram difference is less sensitive to object motion than the pair-wise comparison, since it ignores the spatial changes in a frame. However, histograms may also produce missed shots when two frames with similar histograms share a different content. Authors reported that the three types of features face a potential problem in the presence of a high speed motion or a sharp illumination change between two frames; thus resulting in false detections. With

more than 1700 citations, this work is considered as a reference in the field of temporal video segmentation. Similarly to [27], few years later, Boreczky et al. [5] perform a comparison of existing techniques, till that time, including pixel intensity differences [27, 28, 16], histogram differences [18, 17], edge tracking [9], DCT features [19] and those using motion vectors to differentiate between CTs and camera motions like zoom or pan [18, 16]. The majority of listed methods are implemented and tested on different video types (e.g. Television, news, movies, commercials). They found that the DCT features are the fastest and that the motion vectors are not sufficient alone. Histogram based methods give the best trade-off between simplicity, accuracy and speed. However, to enhance the accuracy, motion vectors may be used, as post processing, to eliminate false detections caused by camera motion.

In 1994, Hampapur et al. [28] define a transition as an editing effect and provide several other definitions related to image, image sequence, feature, shot, video, difference image, edit frame, scene activity and edit activity. In their work, various mathematical transition effects are modeled based on video production techniques. Those models are then used to classify frames whiting a shot and frames representing boundaries. An extension of this work was proposed, one year later, by the same authors in another study [29], where they underline that most existing techniques ignore the inherent structure of a video and do not use explicit models of video. In addition, video models are also used to define segmentation error measures. As a matter of fact, such models have proved, a decade later, to be important and necessary for classifying various types of transition effects, as the data features can be used to train a machine learning model. Later in the document, we will discuss such approaches. In the meantime, several methods will be proposed. In 2000, Gargi et al. [30] presented another comparative study, where color histogram algorithms, MPEG compressed video and block-matching techniques are considered. Various implementations and tests were carried out, particularly for histograms as they were the most popular features for SBD. Multiple color spaces (e.g. RGB, HSV, YIQ, XYZ, Lab, Luv, MTM and OPP) are considered using a variety of similarity measures including:

1. Bin-to-bin difference:

$$HD(h_i, h_j) = \frac{1}{2N} \sum_t |h_i[t] - h_j[t]| \tag{2.13}$$

2. Chi-square χ^2 test:

$$HD(h_i, h_j) = \frac{1}{N^2} \sum_t \frac{(h_i[t] - h_j[t])^2}{h_j[t]^2} \tag{2.14}$$

3. Histogram intersection:

$$HD(h_i, h_j) = 1 - \frac{1}{N} \sum_t \min(h_i[t], h_j[t]) \tag{2.15}$$

Except for few color components and metrics, generally histograms give close results. The 3D histogram intersection in the MTM color space reaches the highest score. The luminance is an important feature for detecting the shots, but it didn't perform well alone. Histogram differences based methods are the most used for video cut detection, since they are fast and accurate [5, 8, 10, 30]. Several similar studies have been performed so far in this sense, with a difference in the parameters used, such as color space, threshold calculation or in a pre processing step. In most cases the similarity measure is calculated according to 2.16.

$$CHD_k = \frac{1}{N} \sum_{r=0}^{2^B-1} \sum_{g=0}^{2^B-1} \sum_{b=0}^{2^B-1} |p_k(r,g,b) - p_{k-1}(r,g,b)|\,, \tag{2.16}$$

where $p_k(r,g,b)$ is the number of pixels color (r,g,b) in the frame k of N pixels. If this distance is greater than a predefined threshold, a cut is detected. In their work, Priya et al. [31] divided each frame into R regions. The bin wise histogram difference between each block of two successive frames is calculated using the same equation 2.16. The similarity between two consecutive frames is represented by the sum of similarities between all regions in those frames.

$$BBHD_k = \sum_{r=1}^{R} CHD_{k,r}. \tag{2.17}$$

In their work, they use a global threshold value. The drawback of this method is that it may produce missed shot if two frames have a quite similar histogram while their contents are dissimilar. In 2001, Koprinska et al. [13] presented a work that gives an overview of existing techniques for video segmentation which operate on both uncompressed and compressed video stream. Similarly, Alan Hanjalic [3] provided an excellent analysis of the shot boundary detection problem in detail. During this decade ($1990 - 2002$), several works and efforts have been supported to better understand and solve the shot boundary detection problem. Most studies have taken into consideration only the cut detection and have lead to good detection. That said, the emergence of new technologies has increased, exponentially, the number and the complexity of the new created videos. In [3], the author mentioned that some CTs are easy to detect using any arbitrary feature and metric. However, he underlines that a good feature must detect the most difficult changes in order to reduce the missed shots, while being robust

to the various disturbances (i.e. illumination and motion changes) to decrease the false detections. Even if a large number of SBD techniques have been presented in the early years, most methods were evaluated on a relatively small data set due to the lack of large annotated video collections. It can be confirmed that during these years, the field was born and much efforts were provided to better solve the SBD problem. It is true that the best methods were proposed after 2002 with the progression of technologies, however, this is strongly due to the early proposed works and established research.

Recent related works

Considering the nature, the diversity and the complexity of a video signal, it is no longer appropriate to utilize directly conventional features such as pixel intensities, edges, or histograms. From 2001 to 2009, the National Institute of Standards and Technology (NIST) sponsored the TRECVid conference series for video processing, in which SBD was one of the evaluation tasks. During those years, more than 57 different SBD methods were proposed and tested over the annual TRECVid benchmarking exercise, using a common data set and common scoring metrics. This has significantly promoted the progress of the SBD field. Since then, a variety of algorithms, using either, simple, combined or multiple features with or without pre-processing, have been developed during this decade, including the fast framework [32], the SVD based-methods [33, 34], the fuzzy color histogram [35], the fuzzy-rule-based approach (AVCD-FRA) [36] and the Walsh-Hadamard transform (WHT) [37]. The work in [38] presented a high-level overview of the most significant approaches related to TRECVid shot boundary detection task, with a comparison of performances focussing on the TRECVid 2005 dataset. More detailed comparisons and discussions of SBD techniques were studied in [5], [3] and [12]. In 2007, Jinhui Yuan et al. [12] presented a formal study of shot boundary detection.

Apart from traditional methods, and from many different approaches based on other features and similarity measures, in 2004, Whitehead et al. [39] propose a new approach that uses feature tracking as a metric dissimilarity. The authors use a corner-based feature tracking mechanism to indicate the characteristics of the video frames over time. The inter-frame difference metric is the percentage of lost features from frames k to $k + 1$. In the case of a cut, features should not be tracked. However, there are cases where the pixel areas in the new frame coincidentally match features that are being tracked. In order to prune these coincidental matches, they examine the minimum spanning tree of the tracked and lost feature sets. In their work, they also propose a method to automatically compute a global threshold to achieve a high detection rate. Although this subject has long existed and that much effort has led to

good results, most SBD methods fail to find a tradeoff between detection accuracy and computational cost. Motivated by the real-time applications requirement, in 2009, Y. Li et al. [32] presented a fast framework using candidate segment selection. A segment is processed (i.e. considered as a candidate segment) if its first and last frames share a different visual content. In fact, consecutive frames within a short temporal segment are usually high correlated. Thus, several segments will be skipped, allowing to save computational time, while maintaining the same detection accuracy. To measure the distance between the ïňĄrst and last frames of each segment, the SAD of the pixel intensities in the luminance component is employed. Then, to distinguish between non boundary segments and candidate segments that may contain a transition, an adaptive local thresholding is adopted. To improve the speed, a bisection based comparisons is performed. In their work [32], authors divide a video into segments of 21 frames and merge every ten segments together into a basic thresholding unit. The local threshold for each unit was defined as follows:

$$T_L = 1.1\mu_L + 0.6\frac{\mu_G}{\mu_L}\sigma_L. \tag{2.18}$$

where μ_G is the mean of all the distance values (i.e. global mean), μ_L denotes the mean of the distance values in a thresholding unit (i.e. local mean) and σ_L represents the local standard deviation. For CT detection, authors used the same criteria employed in [14]. Although their method is fast, it misses a large number of shots producing a poor recall rate. Following the same concept of candidate segment selection, the SVD-based method [34], presented in 2013, can be considered as the fastest one. In their work, the normalized HSV color histograms, denoted h_i, are extracted from each frame f_i and used as features. The column vectors of N frames are grouped together to compose the frame-feature matrix $H = [h_1, h_2, ...h_N]$, where N is the length of segments. The singular values decomposition (SVD) of the matrix H is then performed for dimensionality reduction. Applying the SVD, color histograms will be mapped into refined feature space. For CT detection, the same idea of thresholding unit and bisection-based comparisons proposed in [32] are used. To enhance the recall criterion, a different adaptive threshold is adopted:

$$T_L = \mu_L + a\left(1 + \ln\left(\frac{\mu_G}{\mu_L}\right)\right)\sigma_L. \tag{2.19}$$

This leads to decrease the number of missed shots which improves the detection accuracy. In addition, working on refined and reduced feature space allows to save the computational time. Another recent good method is proposed in [36] in 2013. Fuzzy rules are defined for scene

cut identification. Spatial and temporal features are incorporated to describe video frames, and model cut situations according to temporal dependency of video frames as a set of fuzzy rules. The method identifies cut transitions only using a fuzzy logic, without any thresholding, which is more flexible. Their algorithm is robust to object and camera movements as well as illumination changes, nonetheless, the major drawback of the method is that it only detects cut transitions. In term of accuracy, the Walsh-Hadamard transform method [37], proposed in 2014, is considered as the best one according to the F1-score criterion over the TRECVID 2007 video database. In their work, color, edge, texture, and motion are used as vector of features. Extraction is performed by projecting the frames on selected basis vectors of WHT kernel and WHT matrix. The weighted features are combined to form a single continuity signal $\phi(k)$, used as input for Procedure Based shot transition Identification process (PBI). The method classifies shot transitions into abrupt and gradual transitions, with high rates for different criteria. The CT detection is performed by finding the peaks of the continuity values $\phi(k)$ using the peak finding procedure. A peak is declared as a cut if it is greater than a global threshold T.

We can conclude from this state of art that a good video cut detection method highly depends on features, similarity measure and thresholds used. We found that the major challenges to CT detection techniques are the various disturbances caused by illumination changes, object and camera motion. These disturbances usually lead to misclassifications. We noticed that methods based on histogram differences give a good tradeoff between speed and accuracy, however, they remain limited. Their major drawback is their sensitivity to illumination conditions of the video. Small variations of light in a same shot can be declared as a cut. A large number of methods for CT detection were proposed reaching almost perfect results. Therefore, from 2010, CT detection was considered as a resolved problem. Another challenging task is to develop a method that is not only insensitive to disturbances, but which should also detect the gradual transitions. As reported in [3], an original and robust SBD method should detect different types of transitions for any arbitrary video sequence.

2.1.2 Gradual shot transition identification

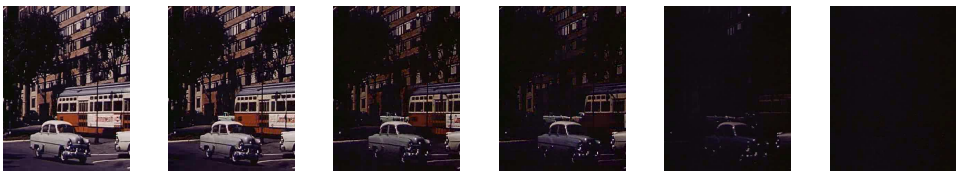
Although CT methods have appeared well before those dealing with the GT changes, these are also quite old. Otherwise, their number is incomparable with CT approaches, due to their difficulties. The GT detection has therefore not experienced such success in terms of results compared to the CTs, which are considered to be almost perfect. The first reliable GT detection method [1] was proposed by Lienhart in 2001, achieving a detection rate of 69%. In their

comparative study [5], authors noticed that algorithms do "a poor job of identifying gradual transitions".

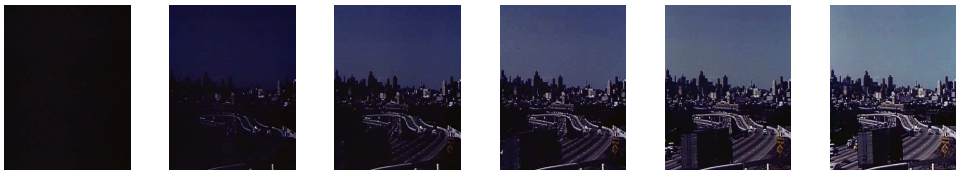
One of the first works designed for detecting gradual transitions was presented by Zhang et al. [27] in 1993. As the inter-frame difference during a GT is smaller than for a CT, two thresholds are used to compare adjacent histograms. When a distance exceeds the first threshold T_{low}, the current frame is considered as the beginning of a GT. The frame is then compared to the following frames in the video. If a distance exceeds the second threshold T_{high} while the difference between consecutive frames is smaller than T_{low}, a gradual transition is declared. The method provides false positives caused by camera and object motion, since they share similar distances. In their work, authors also deal with several motion patterns. For example, when a camera movement (e.g. pan or zoom) is detected, the gradual transition is removed. Although this improves the false alarm rate, it did not handle false positives caused by complex camera motion or object motion [40]. Furthermore, it failed to detect GTs with camera motion during the transition. In fact, these unconvincing results are justified by many additional difficulties encountered during the detection of GTs. Apart from the known disturbances, listed previously (i.e. motions, illumination, noises, etc.), which damage the good detection of CTs, the GT detection requires both spatial and temporal (e.g. spatio-temporal) analysis. Unlike CTs, where the change is observed from one frame to another, comparing two consecutive frame features will never allow to detect the presence of a GT segment. This may result in some confusions, where a GT could be considered as a CT and vice-versa. A video shot containing a GT transition will share specific features over the time, which makes the temporal analysis of segment of features mandatory. Moreover, when studying a set of frames, a GT transition may share the same features than a dynamic segment with lot of motion in the scene, leading to supplementary efforts. In addition, the various existing types of GT segments (i.e. dissolve, fade in, fade out and wipe) make detection even more challenging task. Some existing algorithms are developed to identify only one type independently, while others are designed to detect multiple editing effects simultaneously. In the following, we list each type of GT transition individually, then we discuss other techniques which can detect different types of transitions at the same time.

Fade Out/In

A fade out occurs when the shot gradually turns into a single monochrome frame, usually dark. A fade in takes place when the scene gradually appears on screen. Traditionally, fades in/out are used at the beginning or to conclude a movie or act. An example of fade out/in is illustrated



(A) Fade out



(B) Fade in

FIGURE 2.2: Example of fade out and fade in.

in Figure 2.2. The study of fades locations was proposed in different works [10, 41, 42, 43]. The idea is to first locate all monochrome frames as candidates of fade in/out. The key detection is the recognition of monochrome frames, using the means and standard deviation of pixel intensities to represent the visual content. In [41], authors notice that during a fade in/out, the two adjacent shots are temporally and spatially well separated. In [44], authors assume that the visual effect of a fade editing on the output screen is achieved by a simple addition of two pictures. Let the previous and next shots noted by f_n and g_n, respectively. Then, fade out and fade in can be mathematically modelled according:

$$S_n(i,j) = \begin{cases} f_n(i,j) & 0 \leq n < L_1 \\ \left[1 - \left(\frac{n-L_1}{F}\right)\right] f_n(i,j) + \left(\frac{n-L_1}{F}\right) C & L_1 \leq n \leq (L_1 + F) \\ g_n(i,j) & (L_1 + F) < n \leq L_2 \end{cases} \tag{2.20}$$

$$S_n(i,j) = \begin{cases} f_n(i,j) & 0 \leq n < L_1 \\ \left[1 - \left(\frac{n-L_1}{F}\right)\right] C + \left(\frac{n-L_1}{F}\right) g_n(i,j) & L_1 \leq n \leq (L_1 + F) \\ g_n(i,j) & (L_1 + F) < n \leq L_2 \end{cases} \tag{2.21}$$

where C is the video signal, $S_n(i,j)$ the resultant video signal, L_1 the length of previous shot, F the length of fading sequence and L_2 the length of total sequence. In their work [44], authors propose a simple algorithm for detecting fade out and fade in transitions in both uncompressed and compressed video sequences. Based on the models (2.20) and (2.21), a fade is detected using the horizontal span $X(n)$ of luminance histogram. The process of detecting fades is very simple as the mean and the standard deviation are sufficient to find monochrome frames. Actually, methods for detecting only fade transitions are very few and have soon been replaced by other techniques able to detect several transitions at the same time.

(A) Shots with similar color distributions.

(B) Shots with different color distributions.

FIGURE 2.3: Example of two types of dissolve.

Dissolve

A dissolve transition happens when a shot gradually replaces another one. One disappears as the following appears, and for a few seconds, they overlap, and both are visible. In the process of dissolve, two adjacent shots are temporally as well as spatially associated [10]. It can be also considered as special case of a fade where the monochrome frame is replaced by the next incoming shot. Sample of dissolves are shown in figure 2.3. In 2001, Lienhart [1] defines a dissolve $D(x,t)$ as a mixture of two video sequences, where the first sequence is fading out while the second is fading in:

$$D(x,t) = f_1(t).S_1(x,t) + f_2(t).S_2(x,t), \qquad 0 \leq t \leq T. \tag{2.22}$$

where $S_1(x,t)$ and $S_2(x,t)$ stand for the shots which form the dissolve. In his work, author classifies dissolve into two types:

1. Cross-dissolve, for which $f_1(t)$ and $f_2(t)$ are defined as follows:

$$\begin{aligned} f_1(t) &= \tfrac{T-t}{T}, \\ f_2(t) &= \tfrac{t}{T}. \end{aligned} \tag{2.23}$$

2. Additive dissolve where $f_1(t)$ and $f_2(t)$ are defined according:

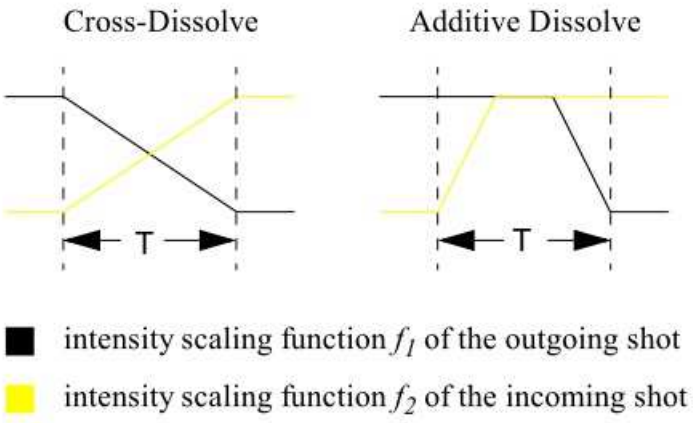


FIGURE 2.4: Intensity scaling functions (taken from [1]).

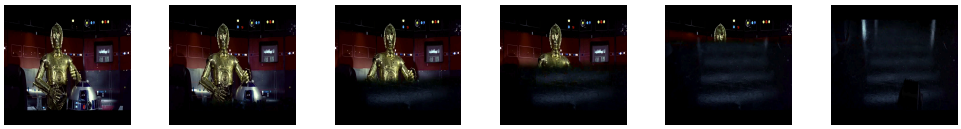
$$f_1(t) = \begin{cases} 1 & t \leq c_1 \\ \frac{T-t}{T-c_1} & \text{else} \end{cases}$$
$$f_2(t) = \begin{cases} \frac{t}{c_2} & t \leq c_2 \\ 1 & \text{else} \end{cases} \tag{2.24}$$

where c_1 and $c_2 \in]0, T[$. Intensity scaling functions for both cross and additive dissolve are described in Figure 2.4.

In another work, Lienhart and Zaccarin [41] propose a system for reliable dissolve detection in which dissolve synthesizer and machine learning are used. Their method had reached a detection rate of 69%, which was the best performance at that time (2002).

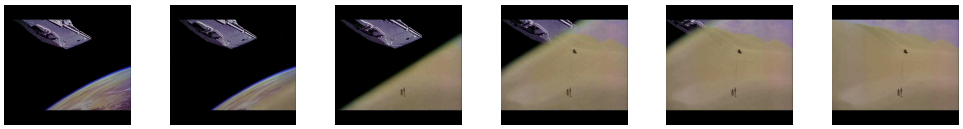
Wipe

The last type, which is the wipe transition, is more dynamic and is considered as the most difficult to model and to detect. It happens when a shot pushes the other one off the screen. In this case, two adjacent shots are spatially separated at any time, but not temporally separated [10]. Its difficulty lies in the number of types of wipe transitions that exists. Indeed, when a shot is moving from the screen (i.e leaving place to the other incoming shot), the movement can be either horizontal (i.e. from bottom to top or vice versa), vertical (e.g. from left to right), oblique (i.e. from a corner to the opposite one), starting from the center, going towards the center or others, etc. To better understand, different types of wipe transitions are illustrated in Figure 2.5. An interesting method for wipe detection is the spatiotemporal slice analysis [45]. A



(A) Motion from bottom to top.

(B) Motion from right to left.



(C) Wipe with oblique motion.

(D) The movement is going to center.

FIGURE 2.5: Various types of wipe transitions.

video sequence is represented by a 3-D volume composed of a set of spatiotemporal 2-D slices. Each slice contains regions of texture and uniform colour. For different styles of wipes, there are corresponding patterns on the spatiotemporal slices. In their work, authors transformed the detection of wipes to the recognition of specific patterns on temporal slices. Another wipe detection method was proposed in [46], also based on the fact that two adjacent shots before and after wipes are spatially well separated at any time. Authors also represent a video as a three dimensional discrete function, where spatiotemporal data blocks and a temporal overlap factor are defined. A wipe transition is detected if the blocks at different spatial locations contain sudden changes in their pixel luminance tracks at different time points, but within a limited time interval. In 2007, Shan Li and Moon-chuen Lee [47] propose an effective method for detecting several types of wipe transitions. In their work, an ideal wipe is modeled as:

$$S(x,y,t) = \begin{cases} S_2(x,y), & \forall (x,y) \in \xi_1 \cup \xi_2 \cup ... \cup \xi_{t-1}, \\ S_1(x,y), & \text{otherwise} \end{cases} \tag{2.25}$$

where $S(x,y,t)$ is the pixel intensity at position (x,y) in frame t, $1 \leq t \leq N$ with N is the total number of frames in the sequence. S_1 and S_2 are the current shot and the next one, respectively. ξ_t denotes the scene change region between frames t and $t+1$, and is defined:

$$\xi_t = \{(x, y) \,|S\,(x, y, t) \neq S(x, y, t + 1)\}\,. \tag{2.26}$$

From this model, properties of independence and completeness are defined to characterize an ideal wipe; frame ranges of potential wipes are then located by finding sequences which are a close to an ideal wipe. The Bayes rule is applied to each potential wipe to statistically estimate an adaptive threshold for the purpose of wipe verification. Their method can detect various wipe effects. Some missed shots caused by motions are reduced by estimating the scene change regions. However, the method fails in detecting wipe transitions within the same scene or with fast motions. Another limitation is the use of a priori knowledge and contextual information of wipes that are incorporated in a statistical detection framework.

Unified approaches

Among the first methods designed to detect both fades and dissolves, we find the work presented by Zabih et al. [9]. Authors extended their work for CT detection using the edge change ratio (ECR), as explained in the previous section, for detecting GT transitions. They notice that during fades and dissolves, edges of the current shot gradually disappear while edges of the new shot become apparent. In other words, the number of exiting edge pixels is high and the number of the entering edge pixels is low during the first half of a gradual transition. This situation is reversed in the second half of a GT transition. Consequently, the value of the ECR increases during a gradual transition. Their method can detect both fades and dissolves, however the false positive rate, especially caused by camera zooms, was unsatisfactory [8, 40]. Moreover, when strong motions occur before or after a cut, it may be classified as a dissolve or fade. Another method that exploits edges was proposed by Yu and Wolf [48], where and edge count is incorporated to capture the changing statistics of fades, dissolves and wipes. In a different way, a variance based approach was firstly proposed by Alattar [49] to detect dissolves, then to detect fades [50]. The main idea is to analyse the temporal behavior of the variance of the pixel intensities in each frame. Author shows that the variance curve of an ideal fade has a semi-parabolic shape while for an ideal dissolve, it has a parabolic shape. Therefore, the GT detection becomes a pattern matching problem within the variance time series. The first order derivative at the boundaries (i.e. before and after) of a GT transition should be zero and a positive constant during an ideal GT. A major limitation is that the behavior of an ideal transition do not match the actual video sequences [40]. In fact, the two main assumptions made by the author are: 1) the transition is linear and 2) there is no motion during the transition.

As these assumptions are not always true, the parabolic curve is not sufficiently distinct which jeopardizes the expected results. To overcome this problem, a B-spline polynomial curve fitting technique was proposed by Nam and Tewfik [51] to estimate the actual transition curve.

In other research, it was pointed that various types of gradual transitions may exhibit similar and unique characteristics over a continuity signal. It was modeled in different works [52, 53, 51] as following:

$$f_t = \alpha_t f_p + \beta_t f_n, \tag{2.27}$$

where f_p and f_n refer to the previous and next shots, respectively. The parameter α_t represents a non linear decreasing function varying from 1 to 0. Generally, for dissolve $\beta_t = 1 - \alpha_t$. Based on the above model (2.27), J. Nam et al. in [51] exploits characteristic transition structures in the underlying special edit effects. B-spline interpolation curve fitting technique is used for estimating the associated linear-like production features and make use of fitting to determine the presence of transition effects. Unlike other previous works, their method aims to identify each specific type of gradual effects such as dissolve, fade and wipe transitions. Another method modeling a dissolve or fade as a time-varying superposition of two shots was proposed in [52]. Authors consider trajectories formed by two frames $f_b - f_a$ and $f_d - f_c$ during a dissolve and by substituting those trajectories, the model in (2.27) yields to:

$$f_d - f_c = (\alpha_d - \alpha_c)(\alpha_b - \alpha_a)^{-1}[f_b - f_a] \tag{2.28}$$

During a dissolve, the normalized correlation between any two trajectories is 1; thus it may be presented as a straight line in the space. Recently in 2012, a unified model for detecting different types of video shot transitions was presented in [53]. P. P. Mohanta et al. formulate frame estimation scheme using the previous and next frames according to (2.29). Transition parameters and frame estimation errors based on global and local features are used to solve boundary detection and classification.

$$f_i = a_i f_{i-1} + b_i f_{i+1} \tag{2.29}$$

where a_i, b_i depend on α_i and are to be estimated. Using both global and local features, the method is more robust against different motion perturbations, and classifies frames into: no change, abrupt or gradual change. In 2013, Lu et al. [34] proposed a fast GT detection is based on SVD. In their work, a signal $S(t)$, which shares specific characteristics in the presence of a gradual transition, is defined as follow:

$$S(t) = \left| \frac{(\beta_p, \beta_t)}{\|\beta_p\| \,.\, \|\beta_t\|} - \frac{(\beta_t, \beta_n)}{\|\beta_t\| \,.\, \|\beta_n\|} \right| \tag{2.30}$$

where βp and βn are frame features of the previous and next shots, respectively. The signal $S(t)$ takes value in the interval $[0, 1]$. For $t = 0$ or $t = N$ (with N is the segment length), $S(t)$ is close to 1, while for $t = N/2$, $S(t)$ is close to 0. Consequently, the curve of $S(t)$ is similar to an inverted isosceles triangle. A pattern matching based on multiple constraints is performed to recognize gradual transitions. The method gives good results and can detect simultaneously cut and gradual transitions; however the gain in speed processing slightly affects the obtained results. Motivated by the rapid progress of this research subject, which is a still open topic, we propose a new and more general model that can detect several types of transitions simultaneously. Explanation and discussion of this work are presented in the following sections related to chapter 4. A consequent application of the shot boundary detection is the keyframe extraction. Intuitively, when segmenting a video sequence into shots, extracting frames from each shot can be seen as a local static video summarization. Other metrics and assumptions can be added to model the key-frame extraction. In the next section, we discuss some existing methods for video summarization.

2.2 Video summarization

Various multimedia applications are rapidly growing with the extensive use of digital video technology and due to recent advances in networks and telecommunications. Besides SBD, video abstraction has been motivated by many other applications including sports games, video surveillance, movies browsing and archiving, medical diagnostic, and so on [54, 55, 56]. It is also considered as an important process in the video indexing [57, 58, 40]. As the name infers, video abstraction allows users to have maximum information about the video content in the minimum time. It consists in producing a short synopsis of a video, which can be either in form of a succession of still images, also known as key-frames, or moving pictures called video skims. Given a video sequence $V = \{f_1, f_2, ..., f_N\}$, where N is the total number of frames in the video. Hence, the key-frames set E and the video skim VS can be defined as follows:

$$E = \{f_1, f_2, ..., f_k\} \tag{2.31}$$

with f_i denotes the i^{th} extracted representative frame.

$$VS = S_1 \oplus S_2 \oplus ..... \oplus S_k \tag{2.32}$$

where S_i is the i^{th} excerpt from the video to be included in the skim sequence and $\oplus$ is the aggregation operator. The main advantage of a video skimming over key frame extraction is its ability to include audio and motion elements that potentially enhance both the expressiveness and information of the summary. On the other hand, key frames are not restricted by any timing or synchronization issues, they are more suitable for browsing and navigation issues. Video skims and key frames are often generated differently; these two forms of video abstract can be transformed from one to the other. Video skims can be created from key frames by joining fixed-size segments, sub-shots, or the whole shots that enclose them, as employed in [58]. Furthermore, the key frames set can be created from the video skim by uniform sampling or selecting one frame from each skim excerpt. One can note many redundancies in the same shot among the frames; then, some frames that best represent the shot contents are selected as key frames to describe the shot. In the following, we first outlines each type independently, where subclassifications are discussed. Then we conclude this chapter by describing the hierarchical video summarization, which can be constructed from both static key frame extraction as well as dynamic video skimming.

2.2.1 Key frame extraction

These are also called representative frames, still-image abstracts, storyboard or static video summarization. It consists of extracting a set or a collection of salient images from the underlying video source. Key frame extraction can be easily generated using a uniform sampling or even more efficient sampling algorithms. However, these methods may produce many key frames without a semantic importance thus failing to represent the video sequence. In fact, reliable algorithms must select key frames which contain as much significant content as possible and without redundancies. Fundamental features of such techniques, as described in [2], are depicted in Figure 2.6. The features used for key frame extraction include colors distribution, dominant color, textures, edges, contrast, shapes, motion vectors, spatial distribution of motion activity, MPEG-7 motion descriptors, discrete cosine or FFT coefficients, camera activity, and features derived from image variations caused by camera motion. According to [2], techniques developed to extract key frames can be classified into five classes: (i) local or global methods, (ii) clustering based methods, (iii) reference frame-comparison, (iv) curve simplification and (v) object or event detection.

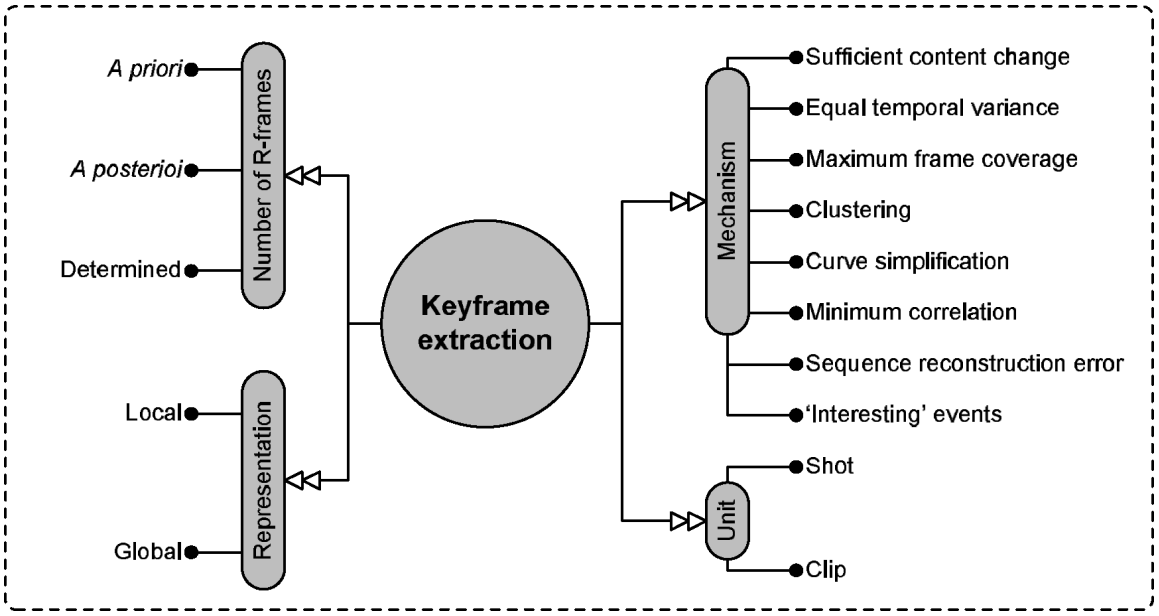


FIGURE 2.6: Attributes of key frames extraction techniques (taken from [2]).

Local and global extraction

In some research, authors classify the video summarization methods depending on their representation [59]. Whether the method is designed for key-frame extraction or video skimming construction, it can be classified as a local or a global method. Considering temporal information, techniques proposed in local search for key frames compare sequentially current frames with previous selected key-frames until a large difference between the compared frames is obtained. In other words, a key-frame is selected if it differs significantly from its neighboring frames. Most of them use the color histogram differences to extract new key-frames or by analyzing other features. One of the simplest way is to segment the video into shots, and then to select representative frames from each shot. In early proposed works [17, 60], the first, the last or other distinct frames divided by a specific time distance of each shot are returned as the key-frames of the shot. Similarly, Zhang et al. [27] segment the video into shots using twin comparisons, then select key-frames based on motion patterns within the shots. Such techniques are inappropriate for non-stationary shots where the visual content may change a lot. To address this problem, Panagiotakis et al. [61] designed a key-frame selection algorithm based on the three iso-content principles: iso-content distance, iso-content error and iso-content distortion. The selected key-frames are equidistant in video content according to the used principle. In another way and without necessarily detecting boundaries, there exist other so-called local methods, which select a frame as representative if and only if its visual content is significantly

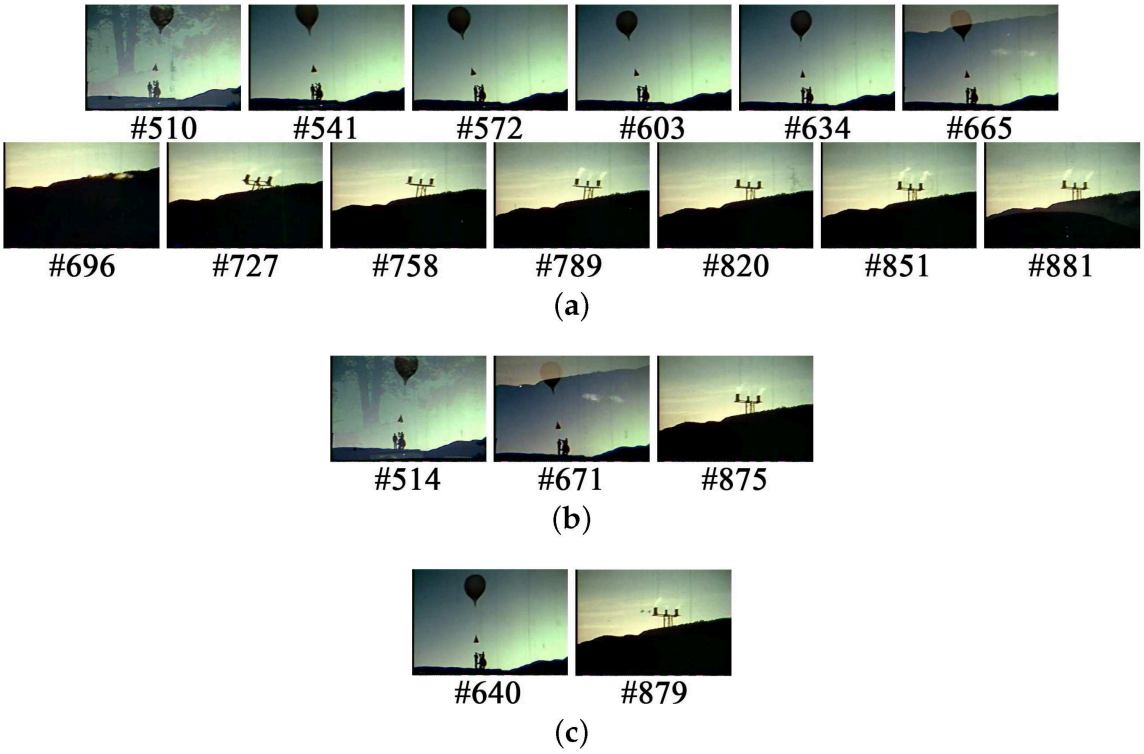


FIGURE 2.7: Key frames selection. (a) Uniformly down-sampled images; (b) Local search results example; (c) Results example of global search.

different from previous key-frames. To measure the sufficient content change, several metrics were used previously such as the histogram difference [62, 63], the intra and inter view correlations in a joint embedding space [64], and the fast full search block matching algorithm [65]. The authors in [66] propose to consider statistics of the macro-block features extracted from the MPEG compressed stream. The work in [67] make use of the accumulated energy function calculated from image-block displacements between two successive frames to measure the distance between frames. The advantages of the local online comparisons include their simplicity and low computational complexity, however, the key frames selected reflect local characteristics of the shot rather than the entire video properties. In addition, the irregular distribution and undefined number of key frames make these algorithms inappropriate for applications that need an even distribution or a fixed number of key frames. Redundancy can be present for the extracted key-frames, as shown in Figure 2.7.

Different from local algorithms, global methods do not consider temporal information and aim to extract key-frames which are expected to be the most representative for the whole video. In global comparison methods, generally the number of key-frames is determined at the beginning of the algorithm. Rather than fixing a specific number, it can represent a proportion ratio

of the total number of frames or over the video length (i.e. duration) and may vary depending on users or desired applications. These approaches are appropriate in telecommunications where offered resources and storage capacity are limited. Ideally, the key-frames extraction problem, with a predefined size k, can be formulated as an optimization problem of finding the optimal set $E = \{f_1, f_2, ..., f_k\}$, which differs least from video frames [2]:

$$\{f_1, f_2, ..., f_k\} = \underset{f_i}{\text{argmin}}\{k|\rho(E, V)|1 \leq i \leq n)\} \tag{2.33}$$

where n is the number of frames in the original video sequence, and ρ is a similarity measure. Other constraints can be added to this model which define the viewpoint of users on what constitutes the optimal key-frame set, for example a *visual coverage*, the number of objects or faces, etc. In their work, Mundur et al. [68] proposed a Delaunay Triangulation (DT) based method to cluster the video frames. The HSV color histogram is extracted from each frame to represent the row vector feature. Their major drawback resides in the computation time, which takes around 10 times the video length. Similarly to DT based method, several other clustering methods have been proposed such as K-means [69], [70], and the graph cuts [71]. Many other ideas have been proposed to exploit the global characteristics of a video. Some of them are listed in what follows.

1) Temporal comparison: These algorithms select key frames which have equal temporal variance. The objective function can be chosen as the sum of differences between temporal variances of all the segments. The temporal variance in a segment can be approximated by the cumulative change of contents across consecutive frames in the segment or by the difference between the first and last frames in the segment. For instance, Divakaran et al. [72] obtain key frames by dividing the shot into segments with equal cumulative motion activity using the MPEG-7 motion activity descriptor, and then, the frame located at the middle point of each segment is selected as a key frame.

2) Maximum coverage: These algorithms extract key frames by maximizing their representation coverage. The idea is to select a fixed number of key-frames which can represent as many frames as possible [57, 2]. If the number of key frames is not fixed, then these algorithms minimize the number of key frames subject to a predefined fidelity criterion. Alternatively, if the number of key frames is fixed, the algorithms maximize the number of frames that the key frames can represent [57]. In the same way, by minimizing a cross correlation criterion among the video frames by means of a genetic algorithm, a small set of key-frames is extracted to provide an efficient description of the visual content [73].

3) Minimum reconstruction error: These algorithms extract key frames to minimize the sum of the differences between each frame and its corresponding predicted frame reconstructed from the set of key frames using interpolation. These algorithms are useful for certain applications, such as animation. Lee and Kim [74] use an iterative procedure to select a predetermined number of key frames, in order to reduce the shot reconstruction error as much as possible. In [75], the authors propose a key frame selection algorithm based on the extent to which key frames record the motion during the shot. In the algorithm, an inertia-based frame interpolation algorithm is used to interpolate frames.

Clustering based approaches

This kind of algorithms cluster frames and then choose frames closest to the cluster centers as the key frames. First designed methods [70, 76, 77] aim to segment the video into shots, then depending on specific clustering algorithms, key-frame are extracted to compose the video summary. Before a new frame is selected, its similarity with the centroid of clusters is computed to avoid redundant frames. In [70], key-frame selection is employed only to the clusters which are big enough. The same strategy is used in [77], where clusters (i.e., shots) shorter than one second are discarded. The scheme in [76] is based on cluster validity analysis and is designed to work without human supervision. A partitioning clustering is applied n times to all frames. Once the optimal number of clusters is found, each cluster is represented by one characteristic frame. In [78], a key frame extraction and foreground isolation method using k-means clustering and mean squared error method is proposed for variable frame rate videos. The foreground objects are selected in the video even as removing the noise occurred in the recording. In [79], an automated algorithm of video key frame extraction based on dynamic Delaunay graph clustering is proposed using an iterative edge pruning strategy. A structural constraint in form of a lower limit on the deviation ratio of the graph vertices further improves the video summary. In addition, an information-theoretic pre-sampling where significant valleys in the mutual information profile of the successive frames in a video are used to extract more useful frames. Various video key frame visualization techniques for efficient video browsing and navigation purposes are incorporated. Figure 2.8 shows an example of video summarization using Clarans algorithm.

Reference frame comparison

The main idea of such algorithms is to generate a reference frame and then extract key frames by comparing frames in the shot with the reference frame. For instance, Ferman and Tekalp

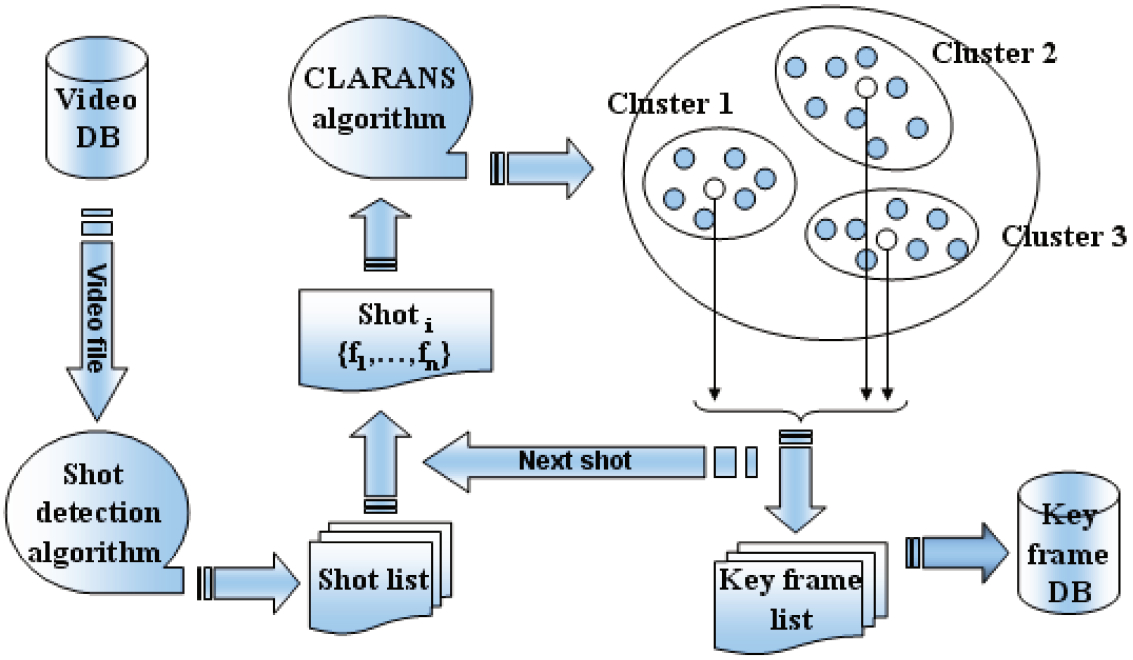


FIGURE 2.8: An example of key frame extraction using clustering.

[80] construct an alpha-trimmed average histogram describing the color distribution of the frames in a shot. Then, the distance between the histogram of each frame in the shot and the alpha-trimmed average histogram is calculated. Key frames are located using the distribution of the distance curve. In another work, Sun et al. [81] construct a maximum occurrence frame for a shot. Then, a weighted distance is calculated between each frame in the shot and the constructed frame. Key frames are extracted at the peaks of the distance curve. The merit of the reference frame-based algorithms is that they are easy to understand and implement. The limitation of such techniques is that they depend on the reference frame: If the reference frame does not adequately represent the shot, some salient contents in the shot may be missing from the key frames.

Object and event detection

These algorithms concern key frame extraction and object/event detection in order to guarantee that the extracted key frames contain information about objects or events. Liu and Fan [82] select initial key frames based on the color histogram and use the selected key frames to estimate a GMM for object segmentation. The segmentation results and the trained GMM are further used to refine the initial key frames. Song and Fan [83] propose a joint key frame extraction and object segmentation method by constructing a unified feature space for both processes,

where key frame extraction is formulated as a feature selection process for object segmentation in the context of GMM-based video modeling. The key frames provide temporal interest points for classification of video events. The merit of the object/event-based algorithms is that the extracted key frames are semantically important, reflecting objects or the motion patterns of objects. The limitation of these algorithms is that object/event detection strongly relies on heuristic rules specified according to the application. As a result, these algorithms are efficient only when the experimental settings are carefully chosen.

Curve simplification

These algorithms represent each frame in a shot as a point in the feature space. The points are linked in the sequential order to form a trajectory curve and then searched to find a set of points which best represent the shape of the curve. Calic and Izquierdo [84] generate the frame difference metrics by analyzing statistics of the macroblock features extracted from the MPEG compressed stream. The key frame extraction method is implemented using difference metrics curve simplification by the discrete contour evolution algorithm. The merit of the curve simplification-based algorithms is that the sequential information is kept during the key frame extraction. Their limitation is that optimization of the best representation of the curve has a high computational complexity. Due to the subjectivity of key frame definition, there is no uniform evaluation method. In general, the error rate and the video compression ratio are used as measures to evaluate the result of key frame extraction. Key frames giving low error rates and high compression rates are preferred. In general, a low error rate is associated with a low compression rate. The error rate depends on the parameters in the key frame extraction algorithms. Examples of these parameters are the thresholds in sequential comparison-based, global comparison-based, reference frame-based, and clustering-based algorithms, as well as the parameters to fit the curve in the curve simplification-based algorithms. Users choose the parameters according to the error rate that can be tolerated.

Sparse dictionary selection

In other recent works [85, 59], the problem of video summarization is formulated as a dictionary selection problem using sparsity consistency as follow:

$$\min_{X} : \frac{1}{2} \|V - VX\|_F^2 + \lambda \|X\|_1 \tag{2.34}$$

with V is the initial frame feature matrix, where each column vector $v_i \in \mathbb{R}^d$ denotes the feature vector of the frame f_i. $X \in \mathbb{R}^{n \times n}$ is the pursuit coefficient matrix. $\|.\|_F$ is the Frobenius norm and the l_1 norm is used here to ensure sparsity. The purpose of such methods is to select a dictionary of key-frame such that the original video can be best reconstructed from this representative dictionary. Thereafter, optimization algorithms are introduced to solve the dictionary selection model. In [59], they reconsider the video summary task as a minimum sparse reconstruction problem so that the original video may be reconstructed with few key-frame. The sparse constraint L_0 norm is used instead of the relaxed constraint $L_{2,1}$ norm. Additional constraints are also defined and their model for video summary is constructed as follow:

$$\begin{aligned} & \min_{S} : \ \tfrac{1}{2} \left\| F - F_K A \right\|_2 + \lambda \left\| S \right\|_0 \\ s.t. \quad & F_K = FS \\ & A = f(F, F_K) \end{aligned} \tag{2.35}$$

where A is the reconstruction coefficients of F by F_K using the function $f(F, F_K)$. $\|.\|_0$ and $\|.\|_2$ are the L_0 and L_2 norm of a matrix or vector, respectively. λ is the trade-off between the two parts of the object function. In the optimisation model (2.35), the first part is to decrease the least-square reconstruction error (LSRE), while the second part confines the number of key-frame as much as possible.

2.2.2 Dynamic video skimming

Also called a moving-image abstract, moving storyboard, or summary sequence [2], dynamic video skimming methods condense the original video into a much shorter version that consists of important segments selected from the original video [86, 87, 88]. This shorter version can be used to browse or to guide the editing of the original video. An example of this process is shown in figure 2.9. The merits of dynamic video skimming include the following:

1. It preserves the time-evolving nature of the original video.
2. Audio track can be included in skims.
3. It is often more entertaining and interesting to watch a skim rather than a slide show of key frames.

The limitations of dynamic video skimming include the following:

1. The sequential display of video skims is time-consuming.

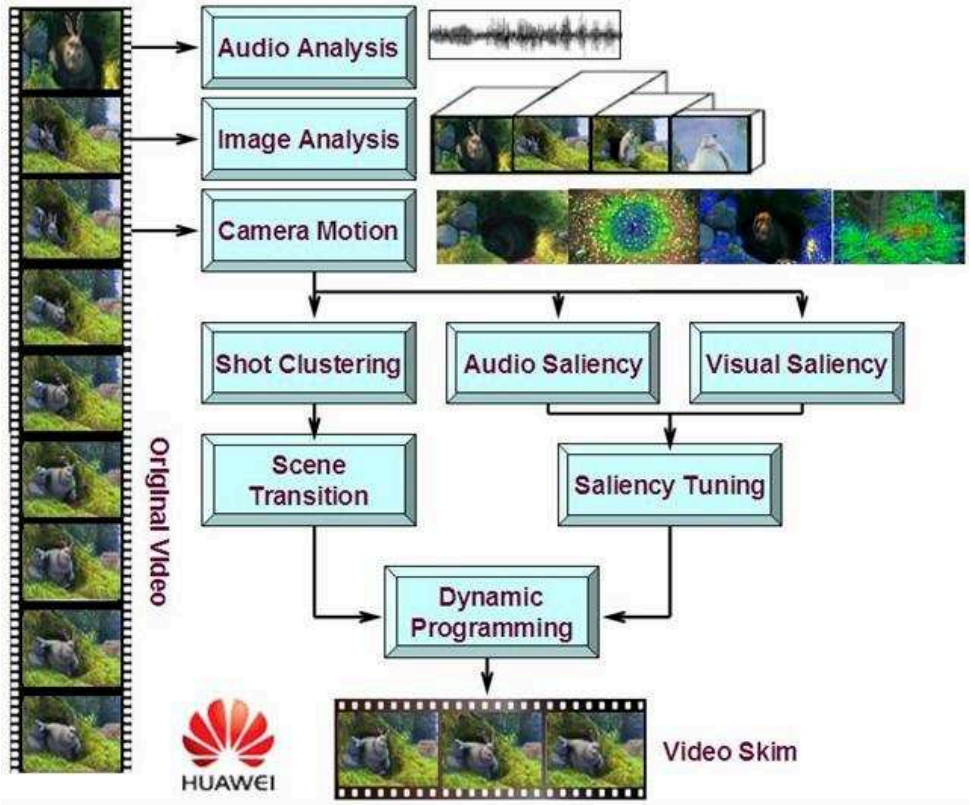


FIGURE 2.9: Video Skimming with Audio-Visual features

2. The content integrity is sacrificed, while video highlights are emphasized.

We found in the literature three main approaches to video skimming: redundancy removal, object or event detection, and multi-modal integration.

Redundancy removal

This approach removes uninformative or redundant video segments from the original video and retains the most informative video segments that are concatenated to form a skim. Ngo et al. [71] represent a video as a complete undirected graph and use the normalized cut algorithm to optimally partition the graph into video clusters. At most one shot is retained from each cluster of visually similar shots in order to eliminate redundant shots. Gao et al. [89] propose a video summarization algorithm suitable for personal video recorders. In the algorithm, according to the defined impact factors of scenes and key frames, parts of shots are selected to generate an initial video summary. Then, repetitive frame segment detection is applied to remove redundant information from the initial video summary.

Object or event detection

Many object-based skimming, uses face detection on broadcast video programs. In these algorithms faces are the primary targets, as they constitute the focus of most consumer video programs. Pertinent events can be used in highlight-based video skims. For instance goals are detected as important events in summaries sports videos. In [90] a Bayesian network-based method is proposed for shot boundary detection, shot view classification, mid-level visual feature extraction, and construction of the related Bayesian network. The shot boundaries are firstly detected. Using the hidden Markov model, the video is segmented into large and meaningful semantic units, called play-break sequences. Many features are derived from each of these units, the Bayesian network is used to extract high level semantic feature from these features. The Bayesian network is constructed using the Chow-Liu tree. The joint distributions of random variables of the network are modeled by applying the Farlie-Gumbel-Morgenstern family of Copulas. The authors claim detecting seven different events in soccer videos; namely, goal, card, goal attempt, corner, foul, offside, and non highlights.

Audio-visual and textual based summarization

For videos whose content is largely contained in the audio, such as news programs and documentaries, the spoken texts can assist video summarization. Once caption texts or speech transcripts in a video are available, a text summary can be integrated with the visual summary into the video skim, or the video sections corresponding to the selected texts can be concatenated to generate the video skim. For instance, Taskiran et al. [91] divide a video into segments by pause detection, and derive a score for each segment according to the frequencies of the words in the audio track for the segment. A summary is produced by selecting the segments with the highest scores while maximizing the coverage of the summary over the full video. Gong [92] summarizes the audio and visual content of a source video separately and then integrates the two summaries using a bipartite graph. The audio content summarization is achieved by selecting representative spoken sentences from the audio track, while the visual content summarization is achieved by preserving visually distinct contents from the image track. In [93], detection of highlights is formulated on the basis of saliency models for the audio, visual and textual information transmitted in a video sequence. Audio saliency is evaluated by signals that quantify multi-frequency waveform modulations, extracted through nonlinear operators and energy tracking. Visual pertinence is measured through a spatio-temporal attention model using color and motion. Text is taken from subtitles available with most movie distribution.

The multi modal curves are integrated in a single attention curve, where the presence of an event may be signified in one or multiple domains. This multi-modal curve improves results from unimodal or audiovisual-based skimming.

2.2.3 Hierarchical video summarization

Hierarchical video abstracts can be constructed from static video summary as well as video skimming. Taskiran et al. [94] group key frames extracted from shots using pixels colors, edges, and other features and structure them in a hierarchical manner using a similarity pyramid. Girgensohn and Boreczky [95] select key frames using the complete link method of hierarchical agglomerative clustering in the color feature space. Geng et al. [96] proposed a hierarchical video summarization method based on video structure and highlights. In this method, frames, shots, and scenes are clustered using visual and audio attention models. According to the measured ranks, the skim ratio and the key frame ratio of the different video structure units are calculated and used to construct summaries at different levels in a hierarchical video summary. Ciocca and Schettini [97] omit useless key frames using supervised classification of visual features, with other visual features based on visual attention model. Then, the key frames are grouped into clusters to allow multilevel summary using both low and high-level features. In [98], a hierarchical video structure summarization approach using Laplacian Eigenmap is proposed. A set of reference frames is selected from the video sequence to form a reference subspace to measure the dissimilarity between two arbitrary frames. The shot-level key frames are first detected from the continuity of inter-frame dissimilarity, and the segment and scene levels representative frames are classified based on k-mean clustering. In [99], a hierarchical video summarization algorithm is proposed. It includes two levels: an entire-level and an object-level. The holistic-level summarization allows global comprehension of the original video, whereas the object-level summarization extracts the description of each object, including trajectory, direction, time, changes of appearance and indication of the loitering behavior. The two abstracts are expressed as two different energy minimization problems, which are resolved using heuristic algorithms.

2.3 Summary

During this thesis, our first research focused on the video summarization field. When reading the literature review of different classifications and methods, we noticed the existence of two types of summaries: static and dynamic. We start by analyzing static techniques which can

be categorized also into two types: (i) video summary using shot boundary detection, known as local methods. (ii) summaries without necessary detecting transitions, called global techniques. Following this, our interest was to study the shot boundary detection area. This has lead to propose different algorithms for cut detection and to design a new approach for gradual transition identification.

In this chapter, we have discussed several existing approaches for both shot boundary and video summary. Based on this, we were able to evaluate the difficulties and the different limitations. Taking into consideration the advantages and drawbacks of several approaches, we provided multiple solutions for detecting different types of transitions. Our researches in video summary concerned the static storyboard construction, where a new global method was proposed. It is worth mentioning that proposed techniques for SBD can be easily extended for static key-frame extraction.

Chapter 3

Video cut detection using simple and projected features

The simplest and most commonly used transitions are the sudden cuts. This chapter presents proposed applications and highlights the works published in [100, 101, 102]. Two different approaches for video cut detection are presented. The first one is developed according to statistical analysis of distances [100, 101] and histograms are used as features. The second [102] is based on projecting histograms into a reduced space using singular value decomposition.

3.1 Statistical analysis

As discussed in the previous chapter, each type of methods has its advantages and drawbacks, and we claim that no method can detect all the shot cuts with a perfect precision. This is due to the various factors that may lead to misclassifications, as seen in the literature review. During our research, we noticed that the misclassifications can be divided into two types: 1) False alarm detection, when a transition is declared while it is not the case. 2) Missed detection and it happens when the method is not able to detect a transition. It is obvious that the second type is very hard to correct. Therefore, in our perception, we tried first to find all the true transition (i.e. zero missed shot), and then correct the false alarm detection. The idea is to design a video shot boundary detection technique in two steps. If the first one is based on histogram differences, the second feature should be insensitive against illumination changes. Figure 5.1 outlines the different steps of our proposed approach. In [30], authors underline the importance of the luminance component in SBD features. In our work, we implement, in different ways, the work proposed in [30] and noticed that we can usually get the same results for different color spaces when manually adjusting the threshold. We also notice that the luminance component from different color spaces shares quite different results. In our

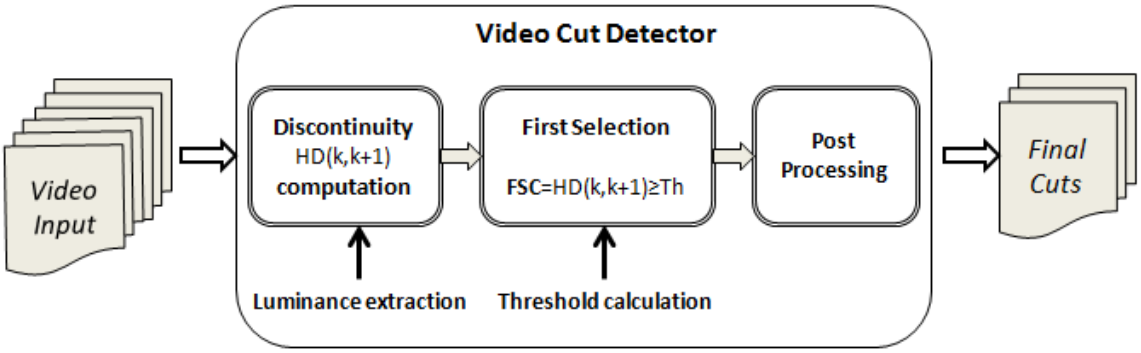


FIGURE 3.1: Different steps of our approach for video cut detection

experiments, we try several combination of components from different color spaces as global or local features. We found that a combination between the luminance Y from the YCbCr color space and the value or brightness V from the HSV color space, works good. This combined color space YV is less sensitive to different illumination changes than using only the Y or the HSV, and performs well for shot cut detection. The concatenation of the RGB color space also works good and gives quite similar results in some cases to the YV color space. However, our main challenge was to find the most appropriate threshold. For this purpose, statistical analysis on the distances between histograms are performed. Following this, we propose two algorithms for cut detection using quite similar thresholds based on the same analysis, while using different features and metrics. The first idea was the selection of the minimum threshold that reaches a perfect rate of the recall criterion (i.e. $R = 100\%$) and proceed to a second verification of the potential false alarms. In the same way, we define a second threshold which reaches a perfect precision rate (i.e. $P = 100\%$), and examine the set of candidate frames.

3.1.1 Minimal threshold selection

The Y luminance and V brightness components are extracted and the histogram differences between every two consecutive frames are calculated to form the vector HD using:

$$HD_k = \sum_{j=1}^{B} |Y_k(j) - Y_{k-1}(j)| + |V_k(j) - V_{k-1}(j)|\,, \tag{3.1}$$

where $Y_k(j)$ and $V_k(j)$ denotes respectively the luminance and brightness histogram value in the j^{th} bin for the k^{th} frame. B represents the number of bins. Such parameters are set experimentally. The distances are then compared against a predefined threshold T. For all distances greater than T, their corresponding frames will compose a set, noted S, of frames that potentially may contain a cut.

$$\text{if } \begin{cases} HD_k > T, \text{ then } f_k \in S \\ HD_k \leq T, \text{ then } f_k \notin S \end{cases} \tag{3.2}$$

Two consecutive frames belonging to the same shot will have a much smaller distance than two consecutive frames belonging to different shots. The idea here is to use the histogram differences as a first filter, so to have a set of frames considered to be potential cuts. In this first selection, there will be some false detections. However, the choice of the threshold T is made in such a way to avoid missed shots. Ideally, the set S should contain all the true shot cuts SC (i.e. with no missed shot) and some false detections FD that we eliminate thereafter. If the number of shot cuts in a video is n, then after the first step, we will have $S = \{SC_1, SC_2, ...SC_n, FD_1, FD_2, ...FD_m\}$. The number of false detections m should be the smallest possible.

Many video cut detection algorithms have been proposed in the past, where several parameters and different thresholds are used to detect the transitions. The common challenge of such methods is the selection of the appropriate threshold that can determine the level of variation between distances. Our initial goal was to define a threshold that best characterizes the shot changes in a video sequence. Such a choice highly dependents on the HD distances vector, which represents the similarity between frames. If we consider the observations vector HD, it can be seen from Figure 3.2, that the distribution of its values have the allure of a log-normal distribution[1]. When a random variable X is normally distributed with $N(\mu, \sigma)$, then the interval of confidence $[\mu - 2\sigma, \mu + 2\sigma]$ covers a probability of 95.5% of the observations [103], which means that only 4.5% of the observations are left in the interval $]0, \mu - 2\sigma] \cup [\mu + 2\sigma, +\infty[$. Generally in SBD, frames depecting shot cuts represent only a small rate (i.e. between 1% to 3%) of the whole number of frames in the video. Since the distances are positive, we can restrict the threshold selection to the interval $I_c = [\mu + 2\sigma, +\infty[$, which contains 2.25% of the observations. Our main objective is to define an appropriate threshold that detects all the shot cuts. Following this, we fixed the minimum value of the interval I_c as an initial threshold, as in equation 3.3. Afterward, we tested several thresholds, to select the best one, according to 3.4:

1. Initial Threshold

$$T = \min(I_c) = \mu_x + 2.\sigma_x, \tag{3.3}$$

2. General Threshold

$$T = \mu_x + \alpha.\sigma_x. \tag{3.4}$$

[1] If a random variable X is log-normally distributed, then $Y = log(X)$ has a normal distribution [103].

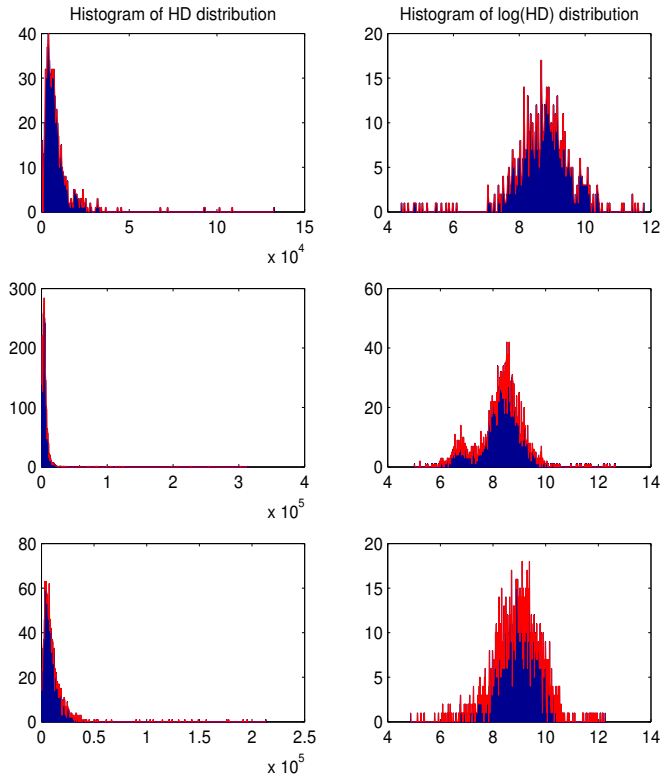


FIGURE 3.2: Samples of the HD distribution on left and its log distribution on right.

where μ_x and σ_x are the mean and the standard deviation, respectively, and α is a fitting parameter. According to our experiment, the number of missed shot increases linearly with the value of α. However, when α is small, the number of false positive is high. By tuning the parameter α, we can select the minimum threshold that detects all the shot cuts with a moderate number of false detections. We noticed that when choosing a small value for α, the results are good for some videos, while the number of false detections is higher for others; but at least in both cases, the smaller α is, the lesser the number of missed shots is. A large value of α decreases the number of false detections, but it results in some missed shot.

Since the threshold strongly depends on the parameter α, the main challenge is to find the optimal value that gives the best result for each video. In SBD, the evaluation is based on statistical error measures (i.e. P, R and F1) and there exists two classes of detection errors. The first error type is the quantity of false detections and is related to the precision criterion P. The second class of error is due to the amount of missed shots and is related to the recall criterion R. This last can not be corrected in post processing. This statement was an important point to consider while attempting to select the appropriate parameter α in our approach. Our idea was to reduce the number of missed shots to zero. When evaluating the results according to the parameter α, we noticed that a score of 100% for the recall criterion is always reached for

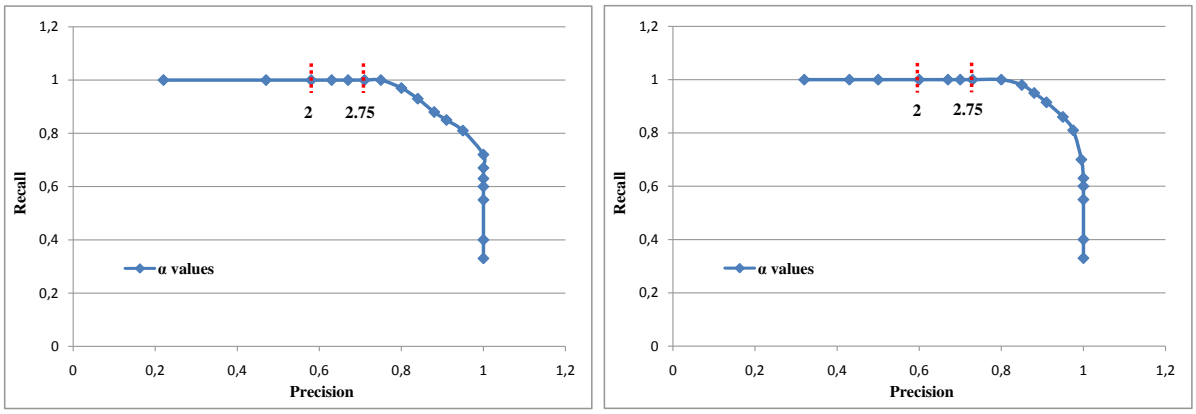


FIGURE 3.3: Precision-recall curve for different values of α over several videos.

any video from $\alpha = 2$. However, an optimal parameter is expected to result into few false detections. Several values of α where tested over various videos as shown in Figure 3.3 and select the maximum value that gives the minimum number of false detections and zero missed shots. It is clear that each video reaches its maximum for a different value. Otherwise, we notice that when $\alpha \in [2, 2.75]$, we get a perfect score for the recall, while the precision accuracy remains satisfactory. Thus, we experimentally set the value of α to 2.75.

In our approach, the first step ensures that the number of missed shots is reduced to zero. Obviously, the number of false detections may be high in some cases. Most of them are due to illumination change. Methods based on histogram differences are sensitive to illumination changes that occur in a scene. Therefore, the second feature in the post-processing should be invariant to these changes. A lighting changes between two frames sharing the same visual content can be seen as a linear combination between them:

$$f_1 = \alpha f_2 \tag{3.5}$$

In some works, the mutual information is used to decrease the influence of the different changes in illumination. A special case of the mutual information is the correlation. More specifically, the correlation is a particular case in which the dependence relationship between two variables is strictly linear, which is the case for the illumination changes in SBD. The correlation coefficient between each frame $f_k \in S$ and its previous one is calculated. If this coefficient is significantly higher than 0.5, this means that the frames are quite similar, and the current frame f_k will be considered as a false detection. Otherwise, the frame will be considered as a true cut and thus maintained in the set S.

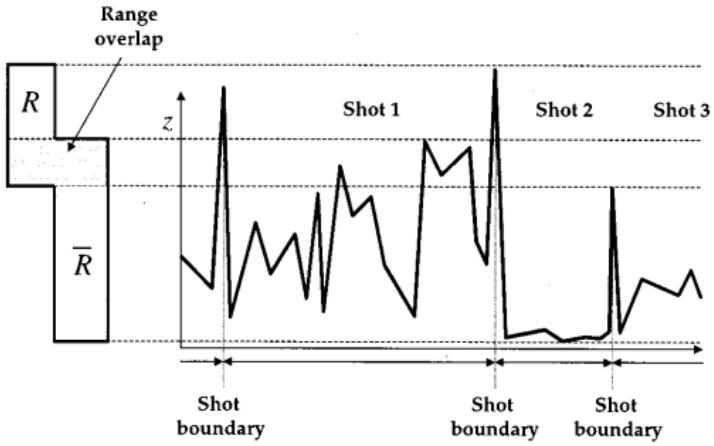


FIGURE 3.4: Illustration of the overlap range between shot cuts and no cuts. (Taken from [3])

3.1.2 Double thresholds algorithm

Another idea based on the same approach and analysis, as explained in Figure 5.1, is to use two thresholds: (i) the first one controls the recall criterion while (ii) the second should control the precision criterion. In this algorithm, we prove the effectiveness of our approach, since the results remains high even when using different features, similarity measures and other post processing. The proposed method is also based on histogram differences, but this time in the RGB color space according to:

$$HD_k = \sum_{j=1}^{B} HDR_k(j) + HDG_k(j) + HDB_k(j), \tag{3.6}$$

with

$$\begin{aligned} HDR_k(j) &= |R_k(j) - R_{k-1}(j)|\,, \\ HDG_k(j) &= |G_k(j) - G_{k-1}(j)|\,, \\ HDB_k(j) &= |B_k(j) - B_{k-1}(j)|\,. \end{aligned} \tag{3.7}$$

where $R_k(j)$, $G_k(j)$ and $B_k(j)$ denote respectively the red, green and blue histogram values for the k^{th} frame; and B is the number of bins. It represents the sum of the pixels belonging to the color bin (r, g, b) in the frame k. After calculating the vector of distances HD, normally, it should be compared against a predefined threshold to locate shot cuts. In his study [3], author mentioned the existence of an overlap interval between clear shot cuts and frames within a same shot as illustrated in Figure 3.4. Thus, the features and metrics used should be as discriminating as possible to be able to clearly detect the transitions. Distances belonging to the

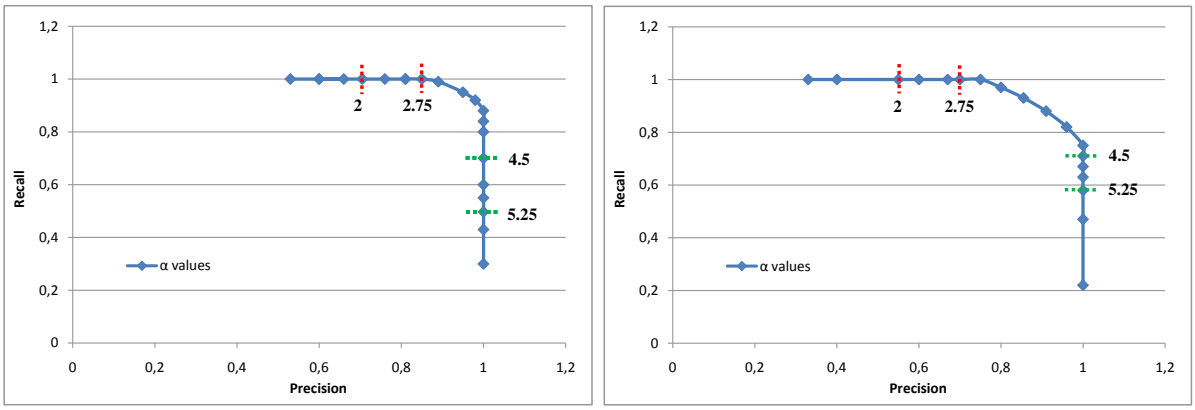


FIGURE 3.5: Precision-recall curve to determine the thresholds.

overlap area make the decision about the presence or absence of shot cuts difficult, particularly with the several factors that may lead to detections mistakes, i.e. missed or false detections. However, in our approach, using a simple threshold, it is easy to recognize the non shot cuts. In a same way and by tuning the parameter α via cross validation, we are able to identify the real shot cuts using a second tougher threshold. Following this, we can define three classes of frames: (i) For distances lower than T_{min}, this means that frames are within the same shot, (ii) if a distance is higher than T_{max}, then the frame surely represents a cut, (iii) when a distance is between T_{min} and T_{max}, its corresponding frame will be considered as a candidate frame, which may contain a shot cut. Thereby, we can isolate the overlap interval as follows:

$$\begin{aligned} &\text{if} && HD_k < T_{min} && \text{then} && f_k \text{ is not a cut,} \\ &\text{else if} && HD_k > T_{max} && \text{then} && f_k \text{ is a cut,} \\ &\text{else if} && HD_k < T_{max} && \text{then} && f_k \in \text{CF.} \end{aligned} \tag{3.8}$$

Once the distances calculated, compared against thresholds and candidate frames set CF identified, we proceed to another post processing to find the final set of shot cuts. As previously explained for the first set S, the set CF will contain several false detections, but will not include all the true cuts. Here, we notice the performance of the second threshold T_{max}, which decreases the number of frames to be processed in the second step. This allows to save computational time. The choice of T_{min} and T_{max} is similar than in the first algorithm, as it can be see from Figure 3.5.

In SBD, histogram based methods are limited and produce several detection errors. Therefore, it is mandatory to combine them with other features to refine the obtained results. However, using both T_{min} and T_{max} on histogram differences, we are able to identify a reduced set of uncertain shot cuts. Thus, we can circumvent the limits of such features with the use of

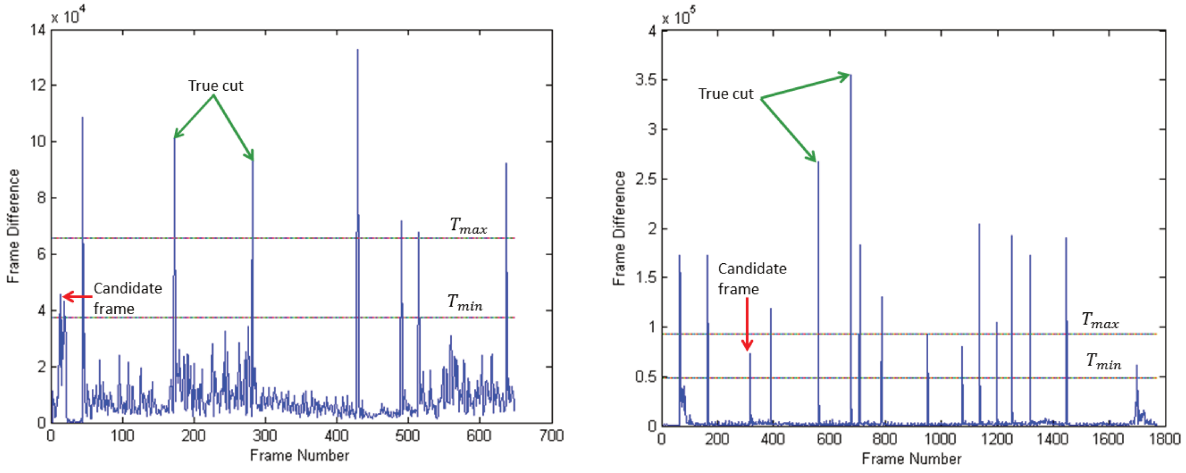


FIGURE 3.6: Dissimilarity measures for two video sequences where the candidate frames are between the thresholds.

Algorithm 1 Surf matching for correction

Input: Candidate frames CF, videofile, T_S
Output: Cuts CTs
1: V = read(videofile)
2: **for** i = 1:length(CF) **do**
3: frame1 = read(V,CF(i))
4: frame2 = read(V,CF(i)-1)
5: sim(i) = surf_matching(frame1,frame2)
6: **end for**
7: CTs = CF(sim>T_S)

another features for decision. Such choice depends on the frames composing the set CF. Generally, frames in the CF set can be divided into two types: (i) frames representing difficult shot cuts, where both previous and next shots are in the same scene or with similar visual content or color distribution. (ii) false detections due to illumination changes. Histogram differences fall in discriminating such situations. In these cases, distances are similar and quiet high, however, they remain usually lower than T_{max}, as it can be seen in Figure 3.6. Local image descriptors are generally robust against the illumination changes and different transformations such as rotations or translations, which can be produced by camera or object motions. Both sift [104] and surf [105] algorithms were tested. In term of accuracy, they share similar results, however, in term of speed, surf is faster than sift. The different steps of our post processing are explained in algorithm 1. Samples of frames from the CF set processed using surf matching for CT decision are illustrated in Figure 3.7.

FIGURE 3.7: False detection excluded by matching surf descriptors on top. Frames representing a bad matching classified as a cut on bottom.

TABLE 3.1: Description of the video dataset

Video sequences	# of frames	duration (s)	# of cuts
Comm_02 (V1)	235	8	0
Bor_03 (V2)	1770	59	14
Indi_001 (V3)	1687	57	15
SITC_tv (V4)	2375	95	41
VidAbs (V5)	2900	116	17
Lisa_CN (V6)	649	21	7
Total	**9616**	**356**	**94**

3.1.3 Experimental Results

In this section, we evaluate the effectiveness of the proposed approach. Various simulations and tests were carried out on standard video databases, especially against a set of videos used in [39] available at [106]. Other videos are from the Open-Video Project [107], as described in Table 3.1. Figure 3.8 illustrates some frames related to the video sequences used. For evaluation, the precision (P), the recall (R) and the combined measure (F1) are used:

$$\text{Precision P} = \frac{N_C}{N_C + N_F} \tag{3.9}$$

$$\text{Recall R} = \frac{N_C}{N_C + N_M} \tag{3.10}$$

$$\text{Combined Measure F1} = \frac{2 \times \text{P} \times \text{R}}{\text{P} + \text{R}} \tag{3.11}$$

FIGURE 3.8: Sample of frames taken from video database used.

TABLE 3.2: The difference between the two steps

	Algorithm 1				**Algorithm 2**			
	Step 1		**Step 2**		**Step 1**		**Step 2**	
	N_F	N_M	N_F	N_M	N_F	N_M	N_F	N_M
V1	0	0	0	0	0	0	0	0
V2	7	0	0	0	5	0	1	0
V3	4	0	0	0	6	0	0	0
V4	3	0	1	0	2	0	1	0
V5	2	0	0	0	1	0	0	0
V6	3	0	0	0	3	0	0	0
Total	**19**	**0**	**1**	**0**	**17**	**0**	**2**	**0**

where N_C, N_F and N_M are the number of true detected cuts, false detections and missed shots, respectively. The precision measure is defined as the ratio of the number of correctly detected cuts to the sum of correctly and falsely detected cuts. The recall is defined as the ratio of true detected cuts to the sum of the detected and undetected ones. The higher these ratios are, the better the performance.

In our approach, we design algorithms in two steps: (i) the first has for objective to detect simple cuts and filter non boundary frames while the second is used to decide for shot cuts and remove potential false detection. Table 3.2 shows the difference between the results of the first step and the second one, where N_F and N_M are defined in (3.9) and (3.10), respectively. It can be seen in both steps, that the number of missed shots is zero. Also, in the first step, we take the result provided by the best threshold each time for each video. The two proposed algorithms perform well and give high rates of precision and recall as illustrated in Table 4.2.

TABLE 3.3: Experimental results

	Algo 1			Algo 2			INT3D [30]			PWD [5]		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
V1	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00 ;	1.00	1.00	1.00
V2	1.00	1.00	1.00	0.94	1.00	0.97	1.00	0.93	0.96	0.93	1.00	0.96
V3	1.00	1.00	1.00	1.00	1.00	1.00	0.93	0.93	0.93	0.92	0.80	0.85
V4	0.98	1.00	0.99	0.98	1.00	0.99	0.98	0.98	0.98	0.84	0.84	0.84
V5	1.00	1.00	1.00	1.00	1.00	1.00	0.94	1.00	0.97	0.75	0.71	0.73
V6	1.00	1.00	1.00	1.00	1.00	1.00	0.88	1.00	0.94	1.00	0.86	0.92
Average	**0.99**	**1.00**	**0.99**	**0.99**	**1.00**	**0.99**	**0.95**	**0.97**	**0.96**	**0.91**	**0.87**	**0.88**

Several comparisons with various techniques were performed. Table 4.2 highlights the results of the comparison with the pixel wise differences (PWD) [5] and the histogram intersection (INT3D) [30]. Figure 3.9 depicts the comparison with the feature tracking method (FTrack) [39] and the block based histogram method (BBHD) [31]. Both Table 4.2 and Figure 3.9 show the effectiveness of our approach which works well and outperforms existing methods.

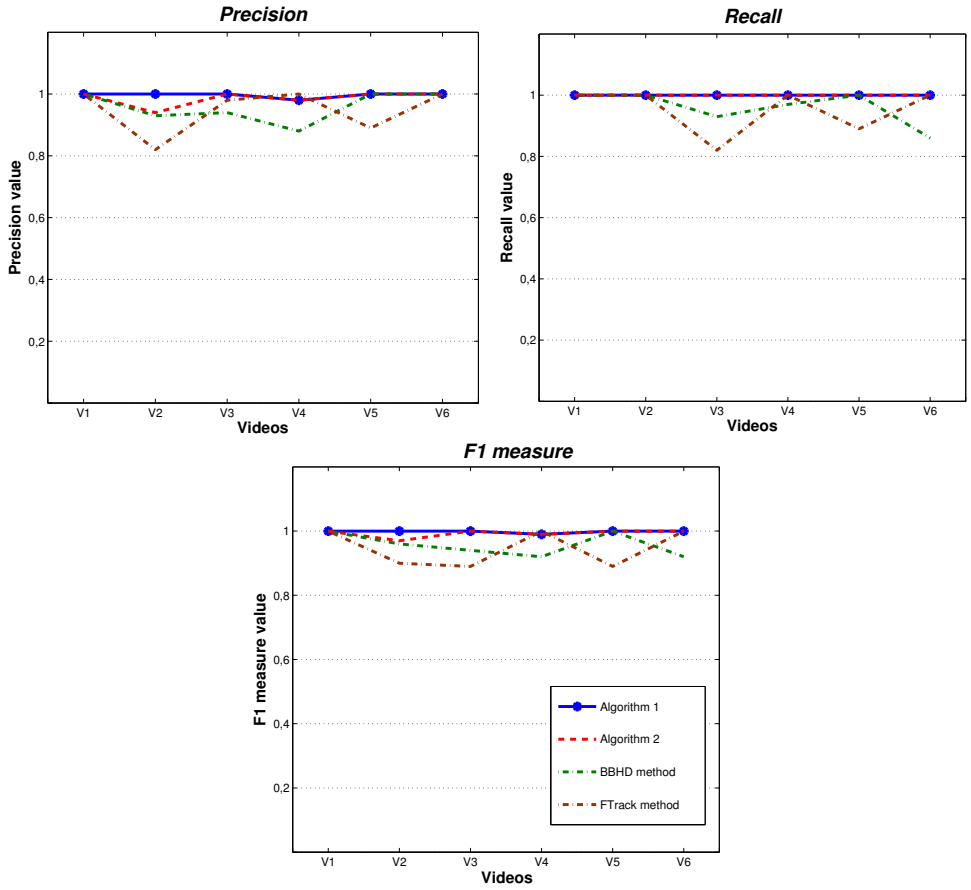


FIGURE 3.9: Comparisons of P, R and F1 criteria of various cut detection methods.

We can notice that among existing methods, our approach leads to a perfect score for the recall criterion $R = 100\%$. Using a post processing, the precision criterion is well increased.

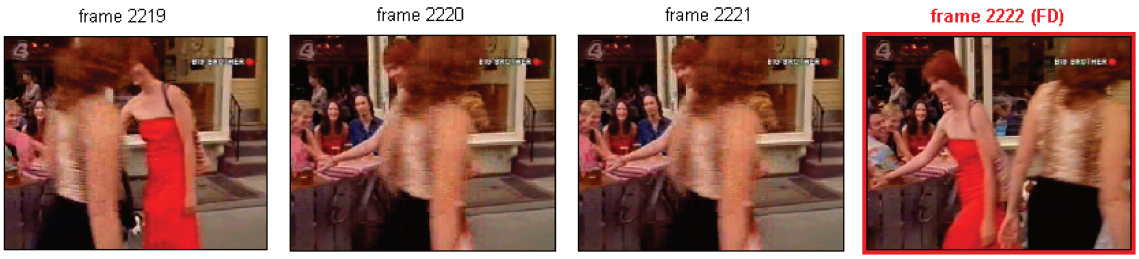


FIGURE 3.10: False detection in red.

Maximum cuts are identified and the average value for the combined measure is 0.99, while for other methods the averages are 0.95, 0.96 and 0.88. The first algorithm provides only one false detection, shown in Figure 3.10. The red surrounded frame represents the false detection. At first glance, this is due to a rapid movement in the scene, but the fact that the frames 2220 and 2221 are similar means that an error of acquisition or transmission has occurred. Between frame 2220 and 2222, we should have another frame than the actual 2221, where the movement is not as significant. This will probably not cause a false detection.

In this first proposed approach, the cut detection is evaluated and two algorithms are proposed based on statistical analysis. Even if the results are satisfactory, processing the video frame by frame increases the time computation. In addition, the use in real-time application is not possible, since the algorithm needs to calculate the distances for all video frames to compute the thresholds. For this purpose, a second approach is proposed, where the video is handled segment by segment to save the time processing. Moreover, to allow the real-time use, features are transformed into a reduced space.

3.2 Video cut detection using linear algebra

With the growth of real-time interactive applications, many fields of research require the elaboration of fast but still effective methods. Great efforts have been made to develop accurate cut detection methods, however, the high calculation cost is a block for real-time applications. Motivated by the urgent need for rapid algorithms, we propose a fast shot cut detection method in this section. Static segment verification and singular value decomposition SVD are adopted to speed up the detection. The main objective of the proposed approach is to reduce the computational cost and to classify segments into static or dynamic. This also can be useful for subsequent applications. Mainly, the information contained in a video sequence aims to display an event or to tell a story during time. Thus, in a video, frames representing non-boundary

are much more numerous than boundary frames. In addition, consecutive frames within a short temporal segment are usually high correlated. Consequently, while processing the video by segment, static ones can be avoided to decrease the computing time. Using a frame step $20 < l < 40$, with l the length of each segment, if the first and last frames are similar, the segment is declared as static. Only non static segments that may contain shot cuts are preserved for further processing. The SVD is then performed to reduce the feature dimension. In the present section, we first introduce the dimensionality reduction provided by the SVD and its interpretation toward shot boundary. The static segment verification, features extraction and cut detection are addressed thereafter. We conclude this section by illustrating the experimental results and comparisons.

3.2.1 Singular value decomposition

As a reminder, we present some properties of the SVD, which is a useful technique in linear algebra. Given an $m \times n$ matrix $H = [h_1, h_2, ...h_n]$, where $m \gg n$ and $h_i \in \mathbb{R}^m$ is a vector column, the singular value decomposition of the matrix $H \in \mathbb{R}^{m \times n}$ is performed using the following equation:

$$H = U\Sigma V^T, \tag{3.12}$$

where $U \in \mathbb{R}^{m \times m}$ and $V^T \in \mathbb{R}^{n \times n}$ are orthogonal matrices whose columns represent the left and right singular vectors, respectively. $\Sigma \in \mathbb{R}^{m \times n}$ is a diagonal matrix whose diagonal elements are the non-negative singular values of H sorted from the highest to the lowest: $\Sigma = \text{diag}(\sigma_1, \sigma_2, ..., \sigma_n)$. This decomposition is called the *full* SVD [108]. A more commonly used form is the *thin* or the *economy* SVD by choosing only the first r-largest singular values, with $r = \text{rank}(H)$. In this second version, the matrix U is reduced to $U_r \in \mathbb{R}^{m \times r}$, V^T to $V_r^T \in \mathbb{R}^{n \times r}$ and Σ to $\Sigma_r \in \mathbb{R}^{r \times r}$. The SVD can reveal important and reduced information about the structure of a matrix as illustrated in the following theorems. For proof, see [108].

Theorem 1. *Let the SVD of H be given by* (3.12) *and*

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r \geq \sigma_{r+1} = ... = \sigma_n = 0, then :$$

1. *Dyadic decomposition:*

$$H = U_r \Sigma_r V_r^T = \sum_{i=1}^{r} u_i.\sigma_i.v_i^T, \tag{3.13}$$

2. *Frobenius Norm:*

$$\|H\|_F^2 = \sigma_1^2 + ... + \sigma_r^2. \tag{3.14}$$

One can see the usefulness of calculating *the economy* SVD from the theorem 1, as long as the singular values from $r+1$ to n are zero. Another useful property is the possibility to calculate the Frobenius norm, using the sum of the r first squared singular values, which contains temporal characteristics.

Theorem 2. *Let the SVD of H be given by (3.13) with $r = rank(H) \leq p = min(m,n)$ and define*

$$H_k = U_k \Sigma_k V_k^T = \sum_{u=1}^{k} u_i.\sigma_i.v_i^T, \tag{3.15}$$

with $k \leq r$, then

$$\min_{rank(B)=k} \|H - B\|_F = \|H - H_k\|_F = \sqrt{\sigma_{k+1}^2 + ... + \sigma_p^2}.$$

On the other hand, the second theorem highlights the utility of the SVD in producing the closest k-rank matrix H_k of the matrix H, which is calculated from the k-largest singular triplets of H according to (3.15). In fact, this low rank approximation H_k, also called the *truncated SVD*, represents a reduced space by choosing only the k-largest singular values, and the k-first elements of singular vectors from V_r^T and U_r. This dimension reduction is not a loss of pertinent information; on the contrary, it often can be more suitable. In video shot boundary, this can be useful in eliminating noises and neglecting small changes.

3.2.2 Shot cut transition identification

The proposed approach processes the video segment by segment, each one of length n, and is composed of two main parts: static segment verification and shot cut identification. The length of each segment S_i is experimentally set to $n = 25$ frames since it represents less than 1 second in a video, where usually the visual content does not change dramatically. This leads to the improvement of time processing as long as only few non static segments will be processed. A segment is classified as static if its first and last frames share a similar visual content. Contextual information may be used to perform this first step. If a segment is declared static, the next one is processed. Otherwise the shot transition procedure is performed. It consists, first, in features extraction, matrix construction, SVD calculation and best low rank selection. After that, each frame will be mapped into a k-dimensional vector. Then, depending on each case, a CT or GT transition identification will be required. For detecting CT transitions, a fast localization algorithm is used. Figure 3.11 outlines the different steps of our CT detection.

Static segment verification

The purpose of this step is to classify the video into static and non-static segments. This allows saving the computation cost, since a video contains more static than dynamic segments. Such classification can also be very useful for further applications such as video summarization. Generally, a segment S_i belongs to the same shot and rarely to two consecutive ones. In the first case, S_i can be either static or dynamic, while in the other case, it may contain a CT or a GT transition. To verify whether a segment is static or not; the concatenated block based histograms (CBBH) $h_{i,1}$ and $h_{i,n}$ are extracted respectively from $f_{i,1}$ and $f_{i,n}$, the first and last frames of S_i. To do this, each frame f_i is divided into 3×3 blocks. Then for each block, Y, R, G and B histograms are extracted from each component separately. The concatenation of the histograms of the nine blocks represents the CBBH of the frame f_i, noted h_i. It can be seen as local features composed by block histograms, and that gives more information with better precision than global frame features. Moreover, extracting histograms is generally simple and faster than extracting complex global or local features. Multiple features were tested, and the CBBH gives the best tradeoff between speed and performance. This is also due to the fact that after the SVD decomposition, the features will be more discriminant. To measure the similarity between first and last frames, the correlation coefficient, noted $d_{i,j}$, is calculated between their CBBH features using:

$$d_{i,j} = d(f_i, f_j) = \rho(h_i, h_j), \tag{3.16}$$

with

$$\rho(h_i, h_j) = \frac{\langle \tilde{h_i}, \tilde{h_j} \rangle}{\|\tilde{h_i}\| \|\tilde{h_j}\|}, \tag{3.17}$$

where $\langle \cdot, \cdot \rangle$ represents the inner product and $\tilde{h_i}$ is the centered CBBH of the frame f_i. If this distance is higher than the predefined threshold T_s, the segment is declared static and the next one is processed. Since a single threshold may not be adequate for a fair classification, contextual information between adjacent features may be performed to enhance the obtained results [12]. We found that the correlation is robust against illumination changes and sufficient for good detection. Two frames sharing the same visual content under different illumination conditions will have a high correlation, as long as they represent a linear combination[2].

[2] $I_1 = \alpha I_2$, with $\alpha \in \mathbb{R}$, means that the frames I_1 and I_2 have a same visual content with different illumination.

Features construction

For each non static segment of length n, an $m \times n$ frame-feature matrix $H = [h_1, h_2, ...h_n]$ is constructed, with $m \gg n$ and where the column $h_i \in \mathbb{R}^m$ represents the CBBH of the frame f_i. The economy SVD of H is then performed using (3.13). In previous works [33, 34], the truncated SVD is calculated for a fixed k. In fact, this dimension reduction is not a loss of meaningful information; on the contrary, removing the $r - k$ smallest singular values is equivalent to removing various noises and disturbances that may arise, as well as neglecting different unimportant changes that can occur in a scene, where the majority of the visual content remains the same. In other words, keeping only the k-largest singular values is the same as keeping only the relevant information of a scene. That said, the choice of the parameter k is problematic, as for a static scene, a small value (e.g., $k = 4$) would be sufficient for a correct classification, whereas for a more dynamic scene where there is much information, a larger value (i.e., $k > 8$) would be required.

Obviously, it would be better adapted to vary k according to each segment than to set it from the start. Let $\tilde{k}$ be the most appropriate parameter for feature extraction. To make it as representative as possible, the parameter $\tilde{k}$ should preserve the useful information while neglecting various disturbances. Among all possible values of k, we highlight here how an appropriate parameter $\tilde{k}$ for feature extraction can be chosen. According to (3.14), it is easy to see that the relative error when one approximates $\|H\|_F^2$ by $\|H_k\|_F^2$, for $1 \leq k \leq r$, is given by:

$$|1 - R_k|, \tag{3.18}$$

with

$$R_k = \frac{\|H_k\|_F^2}{\|H\|_F^2} = \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2}. \tag{3.19}$$

It should be noted that the matrices H_k are never calculated. The ratio R_k can be seen as a projection of the pertinent information in a scene on the entirety of the information contained in that scene. From a more concrete standpoint, the norm $\|H\|_F^2$ represents all the information in the scene, while the norm $\|H_k\|_F^2$ would represent the relevant one. The selected $\tilde{k}$ we are looking for, is defined as the smallest integer belonging to $\{1, 2, ..., r\}$ such that the corresponding relative error satisfies:

$$\left|1 - R_{\tilde{k}}\right| < \varepsilon, \tag{3.20}$$

where the constant ε stands for different noises and little changes in a given scene. The ratio $R_{\tilde{k}}$ will represent an approximate value of 1 accurate within ε. Remark that the discrete function $|1 - R_k|$, of $k \in \{1, 2, ..., r\}$, is decreasing on $[0, 1]$, has r possible values, reaches its maximum value, M [3], at $k = 1$ and its minimum value, $m = 0$, at $k = r$. Thus, $\tilde{k}$ always exists and is unique. Using (3.19), it is straightforward that (3.20) can be rewritten

$$\sum_{i=\tilde{k}+1}^{r} \sigma_i^2 < \frac{\varepsilon}{1-\varepsilon} \sum_{i=1}^{\tilde{k}} \sigma_i^2. \tag{3.21}$$

It can be seen from (3.21) that the $\tilde{k}$ would at the same time conserve the relevant information contained in a minimum number of significant first singular values, while it would discard a maximum number of small insignificant singular values. In our implementation, we iterate k until we reach $k = \tilde{k}$ which satisfies (3.20) and (3.21). After estimating the appropriate parameter $\tilde{k}$, each column h_i will be mapped into the singular space and represented with a reduced projected vector $\Psi_i \in \mathbb{R}^{\tilde{k}}$ according to the matrix $V_{\tilde{k}}^T = [\Psi_1, \Psi_2, ...\Psi_n]$. Each frame $f_i \in S_i$ will be then characterized by a $\tilde{k}$-dimensional vector β_i:

$$\beta_i = \Sigma_{\tilde{k}} \Psi_i. \tag{3.22}$$

This turns out to be very useful since the selected $\tilde{k}$ can also adapt itself according to the transition that occurs. We noticed that for a CT transition, the adapted $\tilde{k}$ is often smaller than the one selected during a GT transition. In addition, when two successive shots belong to the same scene, a $k = 6$ may result in a missed shot, while for $\tilde{k} = 8$ in this case, the change will be detected. Based on these views, it can be concluded that the adaptive low rank significantly improves the obtained results.

Decision and classification

Once all frames within a same segment are mapped into the singular vector space, the similarities d_i^l between consecutive frames are calculated for a frame step $l > 1$ using:

$$d_i^l = d(f_i, f_{i+l}) = \rho(\beta_i, \beta_{i+l}). \tag{3.23}$$

Here, the frame step $l > 1$ aims to divide a segment into small partitions, which allows to better distinguish between static and dynamic segments in a reduced time computation.

[3] M = max $|1 - R_k| = \sum_{i=2}^{r} \sigma_i^2 / \sum_{i=1}^{r} \sigma_i^2$.

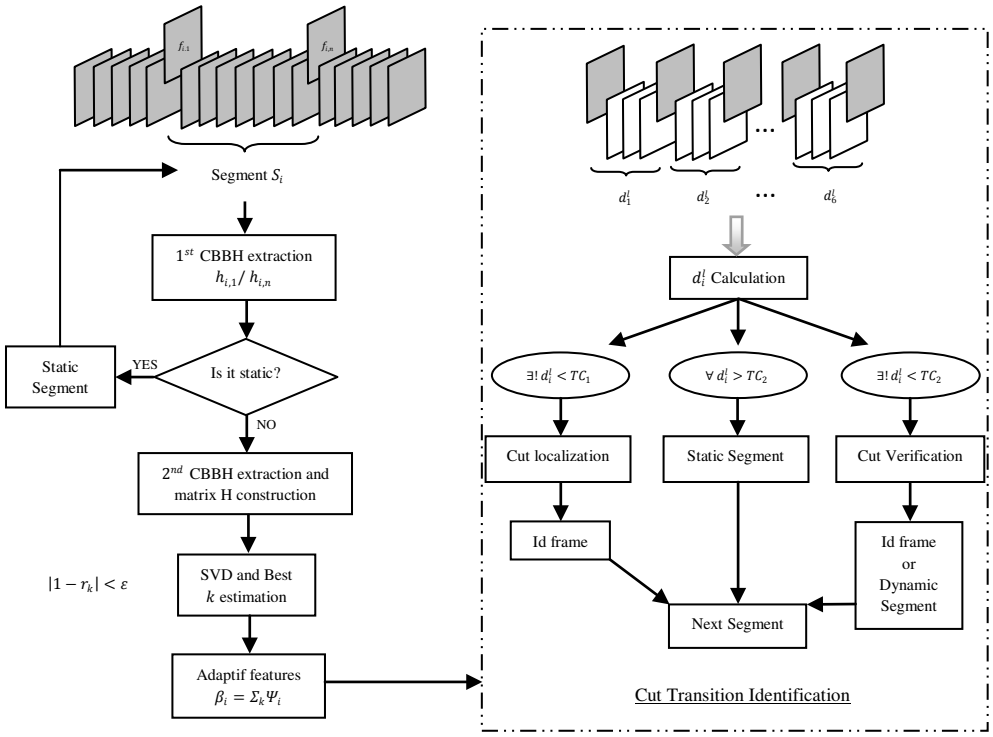


FIGURE 3.11: Different steps for CT detection process including the cut localization and the cut verification.

Moreover, this leads to a simultaneous CT and GT processing, since a pairwise comparison will not display distinct distances in the presence of a GT transition. Once the continuity signal is constructed for a segment S_i, the double thresholding is then performed for the classification of continuity values. If a distance d_i^l is smaller than the first threshold T_{C1}, there is no doubt that a CT transition has occurred, thus the cut localization starts. This is achieved by comparing each two consecutive frames contained between f_i and f_{i+l} according to (3.23) using the same threshold T_{C1}. The different steps of this procedure using $l = 4$ are explained in Figure 3.11. Now if all the distances d_i^l exceed T_{C1}, a second thresholding verification is needed; where if one and only one distance d_i^l is lower than T_{C2}, according to (3.24), a CT transition may be declared after the cut verification step.

$$\exists!\, d_i^l :\ d_i^l < T_{C2}. \tag{3.24}$$

As illustrated in Figure 3.11, the cut verification step is different from the cut localization in that it may result in either a CT transition (i.e., id frame) or a dynamic segment. There is no certainty about getting a CT transition. Now if all the distances d_i^l are higher than T_{C2} and none of them satisfies the equation (3.24), the segment will be declared as static. Otherwise, (e.g., more than one d_i^l satisfies (3.24)) there will be three possibilities. Firstly, the presence of a

TABLE 3.4: Video dataset description

Database 1

Videos	Frames	Cut	Source
Anniversary005 (V1)	11363	39	TRECVID 01 [107]
Anniversary006 (V2)	16588	42	
Anniversary009 (V3)	12306	39	
Anniversary010 (V4)	31391	98	
Airline safety (V5)	12510	45	
Global watcher (V6)	13650	40	
Crew activities (V7)	10267	11	
Landing FCR (V8)	10750	13	
Total	**118825**	**327**	

Database 2

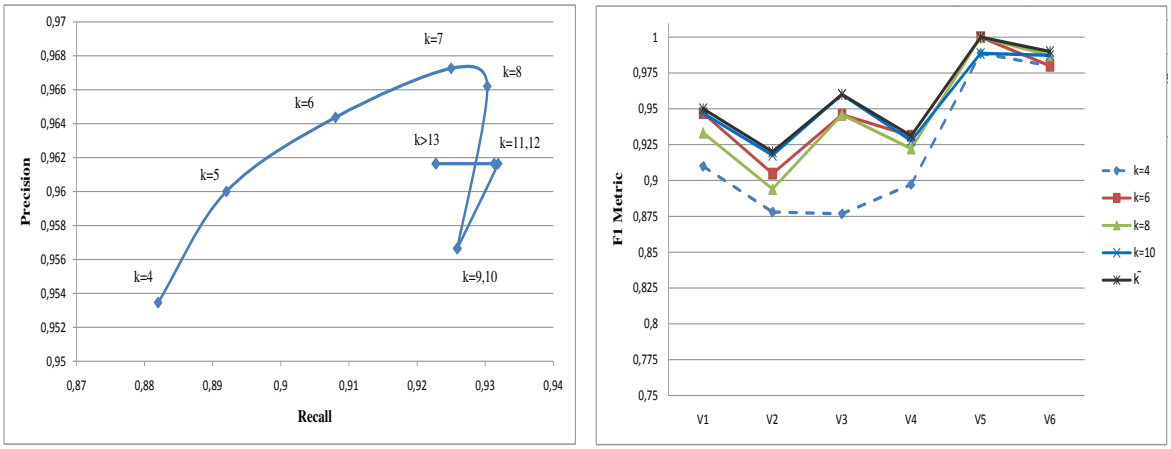
Videos	Frames	Cut	Source
Lisa_CN (V9)	650	7	OTHERS [106]
Comm (V10)	500	18	
Comm_2 (V11)	236	0	
TV_News (V12)	479	4	
VidAbs (V13)	5132	38	
SITC_tv (V14)	2632	34	
BOR_03 (V15)	3183	18	
INDI_001 (V16)	1687	15	
Total	**14499**	**134**	

subshot, if the first and last frames of a segment share a different visual content and where all the consecutive frames are quite similar. This can be due to a camera motion. Secondly, it could be a dynamic segment with a high object or camera motion. And finally it may be a gradual transition GT segment. In order to differentiate one from another, GT transition identification is required. In this section, only CT detection is addressed. An extension of the proposed method which detects GTs is proposed in the following chapter.

3.2.3 Results and discussions

We conclude the present chapter by illustrating our experimental results. To prove the effectiveness of the proposed method, several tests and simulations were carried out using various video sequences taken from the "Open-Video Project" [107] and other sources [106], as described in Table 3.4. The performance is evaluated using the well known precision (P), recall (R) and the combined measure (F1), described in the previous section. The first dataset illustrated in Table 3.4 is used as it contains a number of difficult cuts and was used in recent good works [34, 37]. The second video dataset is selected to perform a comparison with the fuzzy-rule method [36]. Sequences V7, V8 and V16 are used for training and parameters selection via cross-validation. The threshold T_s has a great impact on classifying a segment as static or not. From our experiments, we found that a high value is required to be sure that all segments classified as static are really static, so it was set to $T_s = 0.96$. Another important parameter in our approach is the parameter ε, defined in (3.20), which controls the selection of the best k. As long as the singular values are high, the value of ε will be very small–around 10^{-4}. The remaining two parameters, used for cut identification, T_{C1} and T_{C2} are set to 0.55 and 0.85, respectively.

The proposed algorithm is able to detect 309 cuts among the 327 present in the first database. Few misclassifications are produced, with only 11 false detections and 18 missed shots. Such



(A) Average Precision-Recall over k (B) F1 rate for different videos over k

FIGURE 3.12: Various simulations for different criteria over different values of k.

results are achieved through careful and sharp analysis. The double thresholding with the notion of uniqueness in a refined feature space turns out to be efficient. This strategy allows to minutely detect the most difficult cuts, thus decreasing the number of missed shots. It is also able to recognize sudden motions that may lead to false alarms. The CBBH local histograms and the use of adaptive features $\tilde{k}$ have also a significant impact toward the obtained results. To demonstrate their relevance, the same algorithm was implemented using different values of k for comparisons. The overall average rate of both precision and recall for different k are calculated and represented in Figure 3.12. We notice that for a small value (i.e. $k < 6$), the information is not relevant, as opposed to a larger value where, from $k > 13$, the information remains unchanged. According to Figure 3.12, one can see the drawback of setting the parameter k from the start, namely the difficulty to find a compromise between both criteria P and R, since the precision P reaches its maximum for $k = 6$ while the recall R for $k = 11$. Moreover, a comparison of the combined measure $F1$ when varying the parameter k against the estimated one $\tilde{k}$ is performed, as illustrated in Figure 3.12. As can be seen, the proposed approach reaches high rates for several values of k, however, one can see that the appropriate $\tilde{k}$ gives every time the best result.

In order to demonstrate the competitiveness of the proposed method, a comparison with recent related works is illustrated in Table 3.5. This comparison is performed using the results reported directly from [34] and [37]. The best results are written in bold, unavailable ones are represented with a dashed line ('- -'). The method in [32] reaches a high rate of 0.99 for the precision P, however it gives a poor rate of 0.67 for the recall R. Hence, the overall rate for the F1 measure is only about 0.79, which is not sufficient. The performance of an SBD method is

TABLE 3.5: Comparison with recent related methods

	Related state-of-the-art methods									Method		
	FFRM [32]			**SVD [34]**			**WHT [37]**			**Our VCD**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
V1	---	---	---	---	---	---	0.95	**0.97**	**0.96**	**0.97**	0.92	0.95
V2	**1.00**	0.57	0.73	0.90	0.90	0.90	0.85	**0.97**	0.91	0.91	0.93	**0.92**
V3	**1.00**	0.46	0.63	0.86	0.66	0.75	0.86	0.82	0.84	**1.00**	**0.92**	**0.96**
V4	**0.99**	0.75	0.86	0.89	0.88	0.89	0.90	0.88	0.89	0.94	**0.92**	**0.93**
V5	---	---	---	---	---	---	0.93	0.95	0.94	**1.00**	**1.00**	**1.00**
V6	**1.00**	0.90	0.95	0.97	0.95	0.96	0.97	0.95	0.96	**1.00**	**0.97**	**0.99**
Avg.	**0.99**	0.67	0.79	0.90	0.85	0.87	0.91	0.92	0.91	0.97	**0.94**	**0.96**

TABLE 3.6: Comparison with the AVCD-FRA method

	AVCD-FRA [36]			**Proposed**		
	P	**R**	**F1**	**P**	**R**	**F1**
V9	0.87	**1.00**	0.93	**1.00**	**1.00**	**1.00**
V10	**1.00**	**0.89**	**0.94**	0.94	**0.89**	0.91
V11	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
V12	0.66	**1.00**	0.80	**0.80**	**1.00**	**0.89**
V13	**1.00**	0.92	**0.96**	0.95	0.95	0.95
V14	0.97	**1.00**	**0.98**	**1.00**	0.97	**0.98**
V15	**0.95**	**1.00**	**0.97**	**0.95**	0.95	0.95
Av.	0.92	**0.97**	0.94	**0.95**	0.96	**0.95**

strongly related to the F1 measure since it represents the harmonic average of the recall and precision. As can be seen from Table 3.5, our approach outperforms recent state-of-the-art methods with an average of 0.96 for the F1 metric. Precision and recall also achieve high detection rates with 0.97 and 0.94, respectively. Such results would have been better if the videos did not contain slight cuts and very fast motions, thereby producing various challenging misclassification. To show the difficult transitions, some false detections and missed shots returned by our algorithm are exposed in Figure 3.13. One of the major challenges of an SBD method is the false positives caused by camera flashes. The correlation coefficient between refined features seems to be immune to most of them, however, the coarsest ones lead to a detection error as seen in Figure 3.13. Sample of a gradual segment classified as dynamic by our system is also illustrated in Figure 3.13. The last comparison against the AVCD based method is illustrated in Table 3.6. It can be seen from this comparative table that the AVCD performs very well in detecting CTs with a very high average of $0,97$ for the recall criterion. However, the limitation of the method is that it only detects the abrupt transitions. Almost all reported works since 2010 must detect both hard and gradual transitions. Our CT detection method was thereafter extended toward detecting also GT transitions, as explained in the next chapter.

FIGURE 3.13: Row 1: false detections in red, caused by rapid air-screw motion. Row 2: missed shots, represented in green, due to very similar color distribution. Row 3: Dynamic segment. Row 4: Gradual transition classified as dynamic.

3.3 Summary

In this chapter, two different approaches were presented to solve the video cut detection problem. In the first one, the video is processed frame by frame and is histogram based. Specific thresholds are used to detect distinct shot cuts. Limitations of histograms require the use of post processing to refine the results. Local image descriptors such as surf or sift are insensitive to illumination changes, camera and object motions. For example, when histogram is inaccurate, local descriptors are more discriminating and the combination of the two works good as it was shown in the experiments. Otherwise, the second approach deals with the video by segment of frames and is SVD based. Motivated by the real-time requirement, we speed up the method using a static verification step to reduce the number of segments to process. High performance is reached by the last approach in term of both accuracy and speed. Experimental results concerning time processing are discussed in the next chapter.

Although the obtained results are satisfactory, a robust SBD technique should also detect gradual transitions. The shot cut detection can be considered as a solved problem, where almost perfect results were obtained. Following the work based on SVD features, we present in the next chapter our solution for GT detection. The notion of SVD-updating is introduced and pattern matching is performed for decision and classification.

Chapter 4

Gradual transition detection using SVD updating and pattern matching

As previously mentioned, GT identification is a much more complex task than the CT detection. This is due to: (i) the diversity of existing types of gradual transitions, (ii) various additional difficulties during the presence of a GT segment. Motivated by the rapid progress of this research subject, which is a still open topic, we propose a new and more general model that can detect several types of transitions simultaneously. In this chapter, we present our solution for GT detection [109]. Following the work discussed in Chapter 3 for CT detection, our proposed method for GT detection is also SVD based. Several other properties provided by the SVD proved to be suitable for shot boundary detection, as explained in next sections.

4.1 GT detection via SVD-updating

4.1.1 Folding-in and transition modeling

In addition to the aforementioned properties in the previous chapter, the SVD gives also the possibility to incorporate additional incoming terms in the singular vector space without recomputing the whole decomposition. This is given by the folding-in terms technique, known as SVD-updating. Let $g(h_i) = \Psi_i^P$ denote the function which projects new incoming vectors h_i into the singular space Ω and be defined as:

$$\begin{aligned} g: \quad & \mathbb{R}^m \longrightarrow \Omega \\ & h_i \longrightarrow h_i^T U_k \Sigma_k^{-1}. \end{aligned} \tag{4.1}$$

This technique can be very useful for detecting gradual transition and has never been used in shot boundary detection. It aims to integrate further incoming frames from the feature space used $\mathbb{R}^m$ into the singular space Ω. This allows for performing new processing in the singular

space without recomputing the whole SVD, which significantly reduce the time processing while maintaining the same accuracy.

A gradual transition represents a time varying combination of two shots where:

$$f_t = \alpha(t) f_p + (1 - \alpha(t)) f_n, \tag{4.2}$$

Based on this, we assume that estimating gradual frame features is one of the best ways to solve GT detection problem. However, in some cases, the visual content may change in a same shot even in the presence of a gradual transition. Thereby, the definition in (4.2) stands true only when during a dissolve, both shots are quiet static. In addition, wipe transitions are not included. A more general modeling should consider all types of transitions and must take into consideration also moving objects, camera motion and special effects. For this purpose, a transform function $\psi(f_p)$, respectively $\psi(f_n)$ may be introduced instead of the true f_p, respectively f_n. This may represent small modifications in global or local visual content of adjacent shots, due to camera (i.e., global changes) or objects (i.e., local changes) motion for example. Meanwhile, a more convenient model for transitions can be proposed according to:

$$f_t^{(e)} = \alpha_t \psi(f_p, t) + \gamma_t \psi(f_n, t) + \xi(t), \tag{4.3}$$

where $\psi(x, t)$ represents an estimated transform function based on motion analysis depending on the t^{th} adjacent frame of previous or next shots. This definition considers all kinds of transition including also wipe transitions, where for $\alpha_t = \gamma_t = 1$, $\psi(.)$ may return partial frames of previous and next shots and $\xi(t)$ stands for added special effects. Generally, for dissolve or fade-in/out detection $\gamma_t = 1 - \alpha_t$. Such modeling provides better accuracy which significantly improves the obtained results. However, estimating $\psi(.)$ each time depending on each case requires additional processing, which costs in computational speed. Hence, the major drawback of (4.3) resides in the real-time constraints. Furthermore, in our conception, several possibilities are expected to be eliminated (i.e., static and cut) at this level of the algorithm, which makes the task easier. Thereby, an alternative to the aforementioned definitions is to use a particular feature able to remove different small changes and disturbances in a reduced time processing. In other words, representing an estimated frame f_t^e in the singular space would be better adapted. Moreover, this is necessary to perform comparison since all frames are already mapped in the singular space. This leads to a superior classification in a reduced computational time. Let $h(f_i) = h_i$ be the function that calculates the CBBH of a frame f_i. Then, we define $\Phi(f_i)$ as a mapping function from the image space F to the singular space Ω:

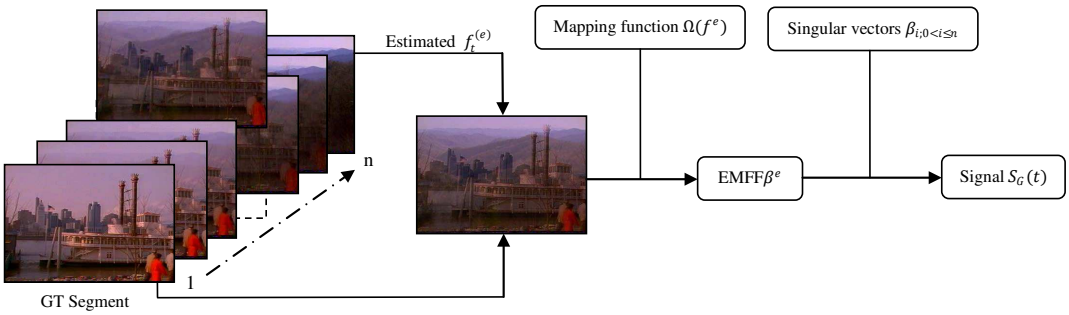


FIGURE 4.1: Discontinuities signal $S_G(t)$ calculation

$$\begin{aligned} \Phi: \quad & F \longrightarrow \Omega \\ & f_i \longrightarrow \beta_i^P \end{aligned} \tag{4.4}$$

where $\beta_i^P \in \mathbb{R}^{\tilde{k}}$ is the projection of the frame f_i into the singular space Ω. In fact, this mapping function Φ represents a composition of the two functions $\tilde{g}$ and h, in which $\tilde{g}$ is a modification of the folding in function g defined in (4.1). Since a frame f_i is characterized by a $\tilde{k}$-dimensional vector $\beta_i = \Sigma_{\tilde{k}} \Psi_i$, which represents the projected vector Ψ_i weighted by the singular values; as a result the mapping function Φ can be defined as follow:

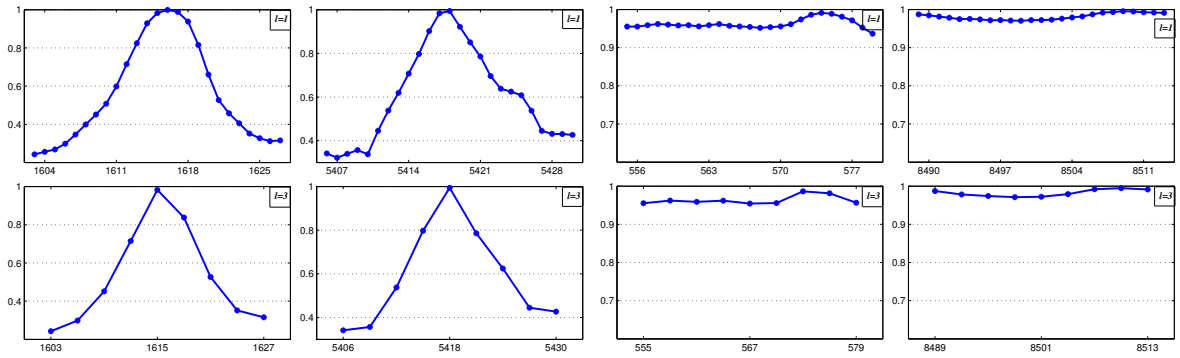
$$\Phi(f_i) = (\tilde{g} \circ h)(f_i) = \tilde{g}[h(f_i)] = \tilde{g}(h_i) = h_i.U_{\tilde{k}}^{'} \tag{4.5}$$

4.1.2 Pattern matching using estimated middle frame

Following this, it is now possible to calculate a continuity signal $S(t)$ between estimated gradual features and the current segment features (i.e., $\beta_{i,\, 1\leq i\leq n}$) in the singular space Ω. In the ideal case, starting from the assumption that for f_n and f_p known, gradual features can be estimated from f_n to f_p and compared with actual segment features; then, the presence of a gradual transition implies the existence of a signal $S(t)$:

$$S(t) = d(f_t^e, f_t) = \rho(\beta_t^P, \beta_t) = 1, \quad p \leq t \leq n, \tag{4.6}$$

where f_t^e is the t^{th} estimated frame and f_t stands for the t^{th} video frame. Due to a lack of ideality in some cases, this may create some confusion. Furthermore, this logical implication remains insufficient for GT detection, since $(S(t) = 1) \Rightarrow$ (GT) is not always true. The only information we can conclude is that: if $S(t) \neq 1$, then, no GT is present. Moreover, estimating each time a number of frames will increase the time processing. To overcome those difficulties, our strategy



(A) GT segments (B) Dynamic segments

FIGURE 4.2: Signal $S_G(t)$ for different segments taken from V5 : (a) segments within GT transition and (b) dynamic segments.

is based on two steps. First, to reduce the computational time, only the middle frame, noted f_m^e, is estimated using (4.2) for $\alpha_t = 0.5$. Its projected feature β_m^P is then calculated using the mapping function Φ in (4.5). A continuity signal $S_G(t)$ is finally constructed as follows:

$$S_G(t) = \rho(\beta_m^P, \beta_t), \ 1 \leq t \leq n. \tag{4.7}$$

The second point is based on scrupulous constraints and fast pattern matching for decision. It strongly depends on the first step, since for a static gradual transition, the projection into the singular space becomes optional for GT detection. However, working on the singular space gives discriminating features, especially with the adaptive low rank $\tilde{k}$ which allows to distinguish between noises, disturbances, insignificant and important changes than a fixed k. It can be seen as a combination of both definitions in (4.2) and (4.3), in which the adjustment is done after estimating the middle frame f_m^e. This highly improves both detection and speed. During a GT transition, the curve of the signal $S_G(t)$ will share specific and unique characteristics. With the assumption that the estimated middle frame feature (EMFF) β_m^P will be similar to the middle singular vector $\beta_{t/2}$ during a gradual transition, $S_G(t)$ will be symmetric and admit a maximum global in the middle. In addition, the curve should linearly increase before the middle and decrease after. On the contrary, during a dynamic segment, $S_G(t)$ will be nearly constant. These unique properties give equivalence between the curve of the signal $S_G(t)$, from (4.7), and the presence of a GT, in contrast to the signal $S(t)$, from (4.6), which represents only a mere implication. It may thus be concluded that the signal in (4.7) highly improves both detection and speed than the signal in (4.6). Examples of different cases of GT transitions and dynamic segments are illustrated in Fig. 4.2. It is visually easy to recognize GT transitions from dynamic segments. The gradual detection is then converted into a pattern recognition

task, in which an isosceles triangle should be identified. For this purpose, three constraints were used to detect an inverted triangle in previous works [32, 34]. First, the distance between the maximum and the minimum should be distinct. Second, the maximum value should be located in the center. Third, there must be very few outliers (i.e. abnormal points). This latter is not significant and can be avoided using a sampling. In other words, a frame step $l > 2$ is sufficient to remove abnormal points, as illustrated in Figure 4.2. Consequently, the constraint is not required anymore. In addition, this leads to save computational time. On the contrary, the first constraint is mandatory, and must be satisfied in the presence of a GT transition. Based on this, a gradual transition is declared if the following constraints are satisfied:

$$\text{Max}(S) - \text{Min}(S) > T_D \ \cap \ \text{Max}(S) > T_G \tag{4.8}$$

$$\exists ! \, S(T) : \begin{cases} S(T) > S(T-1) \\ S(T) > S(T+1) \\ S(T) > T_G \\ T_m - q < T < T_m + q \end{cases} \tag{4.9}$$

If the constraint in (4.8) is not satisfied, the current segment is declared as dynamic and the next one is processed. Otherwise, we verify if there is only one maximum which is higher than the predefined threshold T_G in (4.9). This maximum must be located near to the center T_m to declare a gradual transition. Meanwhile, if the maximum is not located nearly at the center, a correction is generally required by adding frames at last. This case occurs when the GT is located on the edge of the current segment. In our work, instead of recomputing the whole SVD with a shift of L-frame, to save the computational time, the folding-in technique can be performed using the mapping function Φ in (4.4). Only few incoming frame can be added before the GT identification starts. Similarity to the EMFF, their feature vectors are mapped into the singular space to perform comparisons.

4.2 Simulations and discussions

In this section, various simulations and tests are carried out to prove the efficiency of our approach. The selection of video data and parameters used are discussed. As new data are used, CT detection is also evaluated in this experiments. The performance is evaluated using the well defined precision (P), recall (R) and the combined measure (F1). As a reminder:

$$\text{P} = \frac{N_C}{N_C + N_F}, \quad \text{R} = \frac{N_C}{N_C + N_M}, \quad \text{F1} = \frac{2 \times \text{P} \times \text{R}}{\text{P} + \text{R}}, \tag{4.10}$$

TABLE 4.1: Video dataset description

Dataset	Frames	Cut	Gradual	Total
TRECVid 2001	132 407	333	385	718
TRECVid 2002	209 069	362	338	700
TRECVid 2005	744 604	2758	1465	4223
Total	**1 086 080**	**3 453**	**2 188**	**5 641**

where N_C, N_F and N_M are the number of true detected shots, false detections and missed shots, respectively. The closer those criteria are to 100%, the better the performance is. Experimental results on computational time are addressed to assess speed. Comparative studies with recent related works are performed to validate the competitiveness of our approach.

4.2.1 Video data and parameters

Different videos taken from different databases are used to perform comparisons with recent related techniques. The richest video databases for the SBD task are provided by the National Institute of Standards and Technology (NIST) [110]. Each year from 2001 to 2009, researchers set up a TRECVid test collection, containing several videos, for simulations and experimental results. Therefore, specific video sequences related to SBD from three different annual TRECVid dataset were selected for tests and evaluations, as described in Table 5.1. Our detector is first tested on TRECVid 2001 and 2002 videos taken from the "Open-Video Project" [107]. This dataset contains a number of difficult transitions and was used in recent good works [34, 37]. To allow comparison with other SBD methods, the proposed algorithm was tested on the TRECVid 2005 dataset [111]. This choice is motivated by the work in [38], where authors present an overview of the seven years of the annual TRECVid benchmarking exercise, in which they focus their comparisons on TRECVid 2005. Examples of frames composing the TRECVid 2005 are illustrated in Figure 4.3. To sum up, our video dataset is selected as follows:

- TRECVid 2001 [107]: The NASA 25th Anniversary Show including: Segment 05 (V1), Segment 06 (V2), Segment 09 (V3) and Segment 10 (V4). Airline Safety (V5), Global Watcher (V6), The Rio Grande (V7) and Senses Sensitivity (VT1).

- TRECVid 2002 [112]: Exotic Terrane (V8), Hidden Fury (V9), Computer Animation (V10), Wrestling with Uncertainty (V11) and The Dynamic American City (VT2).

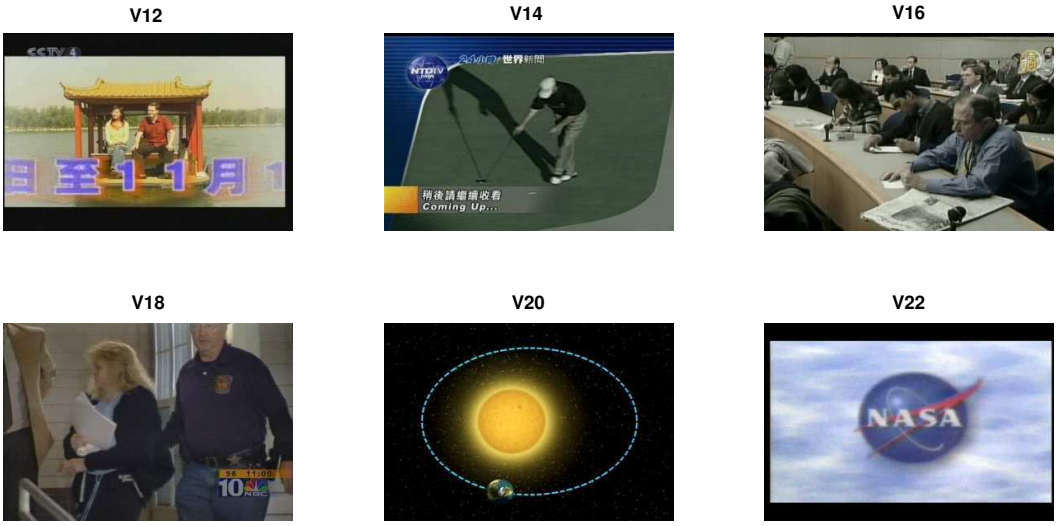


FIGURE 4.3: Frames belonging to TRECVid 2005 database

- TRECVid 2005 [113, 111]: All videos including: Arabic (LBC), Chinese (CCTV-4 and NTDTV), English (CNN, NBC and MSNBC) and broadcast TV news. In the experiments, sequences were denoted (V12-V23) in the order in which they appear in the groundtruth.

As in CT detection, our approach for GT detection stands out by adjusting few parameters, which were also set based on cross-line validation. In fact, the present method is a continuity of the CT method based on SVD. They were both grouped in a single work, and can process simultaneously. As previously mentioned in CT detection scheme, if more than one distance d_i^l satisfies (3.24), GT detection is required. It allows to recognize GT segments and thus to classify the rest as segments within a same shot with high motions or other changes, called dynamic segments. Following the algorithm chaining already designed, two parameters are added for GT identification: $T_D = 0.25$ and $T_G = 0.9$. Several others combinations and tuning can be used. Another idea is to call GT detection when more than a distance is lower than T_{C1} or a quiet higher threshold. This can be useful in some cases, where the GT length is between 8 and 14 frames. The visual content may change considerably during the transition, and sometimes it can be classified as a shot cut. Partitioning a segment S_i allows to better locate a shot cut, and additional information can be retrieved for GT detection.

Another significant tool in our SBD approach is the use of adaptive features, where the relation between Frobenius norm and singular values turns to be useful. The dynamic selection of $\tilde{k}$ shows to be efficient also for GT detection. Generally, the $\tilde{k}$ used for GT detection should be higher than the one used for CT detection.

4.2.2 Results and comparisons

Experimental results for CT and GT detection are illustrated in Table 4.2. The overall detection rates are also reported in the table. It is calculated by summing all present transitions (i.e. CT and GT). Both false detections related to CT and GT are added to get the overall number of false detection provided by the algorithm. It is different from the average value computed by summing half of both values. The performance of an SBD method is strongly related to the F1 measure since it represents the harmonic average of recall and precision. Therefore, if one value is needed for evaluation, it would be the average value of the overall rate of the F1 criterion. The proposed method provides solid performances in detecting transitions, thus a value of F1$= 0.93$ is reached for the most important score. This is due to the high rates achieved by the other criteria P$= 0.93$ and R$= 0.94$.

TABLE 4.2: Experimental results for TRECVid 2001, 2002 and 2005 SBD tasks.

	Results of the Proposed Method											
	Cut Transition				**Gradual Transition**				**Overall Transition**			
	P	**R**	**F1**	**T(%)**	**P**	**R**	**F1**	**T(%)**	**P**	**R**	**F1**	**T(%)**
V1	0.97	0.92	0.95	7.51	0.89	0.96	0.92	6.33	0.94	0.94	0.94	19.6
V2	0.91	0.93	0.92	6.89	0.91	0.97	0.94	6.22	0.91	0.95	0.93	18.5
V3	1.00	0.92	0.96	6.34	0.88	0.88	0.88	7.29	0.92	0.89	0.90	20.1
V4	0.94	0.92	0.93	8.67	0.81	0.96	0.88	7.93	0.90	0.93	0.91	21.2
V5	1.00	1.00	1.00	7.41	0.85	0.88	0.87	6.88	0.94	0.96	0.95	19.3
V6	1.00	1.00	1.00	7.21	0.87	0.93	0.90	7.98	0.93	0.96	0.95	22.2
V7	—	—	—	—	0.98	0.96	0.97	9.54	0.98	0.96	0.97	13.4
Avrg.	**0.97**	**0.95**	**0.96**	**7.33**	**0.88**	**0.93**	**0.91**	**7.45**	**0.93**	**0.94**	**0.93**	**19.1**
V8	0.98	0.95	0.97	6.23	0.89	0.90	0.89	8.92	0.93	0.93	0.93	19.5
V9	0.98	0.95	0.96	8.51	0.92	0.94	0.93	8.21	0.93	0.95	0.94	21.3
V10	0.97	0.93	0.95	4.34	—	—	—	—	0.97	0.93	0.95	10.4
V11	0.96	0.96	0.96	5.12	0.93	0.95	0.94	7.35	0.94	0.95	0.94	17.6
Avrg.	**0.97**	**0.95**	**0.96**	**6.05**	**0.91**	**0.93**	**0.92**	**8.16**	**0.94**	**0.94**	**0.94**	**17.1**
V12	0.96	0.95	0.95	9.67	0.90	0.91	0.90	8.12	0.93	0.93	0.93	22.6
V13	0.90	0.92	0.91	8.49	0.88	0.88	0.88	6.83	0.93	0.91	0.92	25.4
V14	0.97	0.96	0.96	7.54	0.85	0.89	0.87	7.00	0.94	0.94	0.94	21.3
V15	0.96	0.95	0.95	6.88	0.89	0.86	0.87	4.72	0.94	0.93	0.93	17.4
V16	0.97	0.95	0.96	5.79	0.92	0.89	0.90	7.35	0.95	0.93	0.94	24.1
V17	0.98	0.95	0.97	8.67	0.95	0.93	0.94	8.53	0.97	0.94	0.96	31.7
V18	0.97	0.96	0.96	9.02	0.93	0.92	0.92	9.27	0.96	0.95	0.95	36.5
V19	0.95	0.94	0.94	4.53	0.93	0.95	0.94	5.30	0.94	0.94	0.94	22.4
V20	0.95	0.97	0.96	2.65	0.90	0.92	0.91	12.0	0.92	0.94	0.93	30.8
V21	0.97	0.96	0.96	7.33	0.90	0.94	0.92	5.56	0.95	0.95	0.95	23.5
V22	0.94	0.93	0.93	6.03	0.88	0.90	0.89	9.17	0.91	0.92	0.91	19.7
V23	0.96	0.93	0.95	9.93	0.93	0.92	0.92	2.76	0.96	0.93	0.95	20.7
Avrg.	**0.96**	**0.95**	**0.95**	**7.21**	**0.90**	**0.91**	**0.90**	**7.22**	**0.94**	**0.93**	**0.94**	**24.6**

As a similarity measure, the cosine distance, the Euclidean distance and the correlation coefficient were tested using the same algorithm steps and the results are reported in Table 4.3.

TABLE 4.3: Results of the approach using the cosine and the Euclidean distances

	Overall transition using several distances											
	Correlation				**Cosine**				**Euclidean**			
	P	**R**	**F1**	**T(%)**	**P**	**R**	**F1**	**T(%)**	**P**	**R**	**F1**	**T(%)**
V1	0.94	0.94	0.94	19.6	0.92	0.92	0.92	18.1	0.90	0.91	0.90	21.3
V2	0.91	0.95	0.93	18.5	0.90	0.94	0.92	17.8	0.88	0.89	0.88	20.1
V3	0.92	0.89	0.90	20.1	0.91	0.89	0.90	19.4	0.85	0.87	0.86	21.7
V4	0.90	0.93	0.91	21.2	0.90	0.93	0.91	20.3	0.87	0.85	0.86	22.8
V5	0.94	0.96	0.95	19.3	0.93	0.94	0.93	19.1	0.90	0.92	0.91	21.2
V6	0.93	0.96	0.95	22.2	0.93	0.95	0.94	21.7	0.91	0.92	0.91	24.5
V7	0.98	0.96	0.97	13.4	0.96	0.96	0.96	12.9	0.94	0.93	0.93	14.4
Avrg.	**0.93**	**0.94**	**0.93**	**19.1**	**0.92**	**0.93**	**0.92**	**18.4**	**0.89**	**0.90**	**0.89**	**21.1**
V8	0.93	0.93	0.93	19.5	0.91	0.91	0.91	18.1	0.88	0.87	0.88	20.3
V9	0.93	0.95	0.94	21.3	0.92	0.94	0.93	20.5	0.90	0.91	0.90	23.1
V10	0.97	0.93	0.95	10.4	0.93	0.90	0.91	9.83	0.91	0.89	0.90	11.9
V11	0.94	0.95	0.94	17.6	0.94	0.95	0.94	17.1	0.92	0.92	0.92	18.3
Avrg.	**0.94**	**0.94**	**0.94**	**17.1**	**0.93**	**0.92**	**0.92**	**16.3**	**0.90**	**0.90**	**0.90**	**18.4**
V12	0.93	0.93	0.93	22.6	0.92	0.92	0.92	21.2	0.89	0.90	0.89	23.2
V13	0.93	0.91	0.92	25.4	0.90	0.90	0.90	22.5	0.87	0.85	0.86	26.8
V14	0.94	0.94	0.94	21.3	0.89	0.91	0.90	20.2	0.88	0.86	0.87	22.5
V15	0.94	0.93	0.93	17.4	0.92	0.89	0.91	14.6	0.87	0.87	0.87	18.2
V16	0.95	0.93	0.94	24.1	0.93	0.91	0.92	23.5	0.90	0.89	0.90	24.9
V17	0.97	0.94	0.96	31.7	0.95	0.92	0.93	29.6	0.92	0.91	0.91	34.6
V18	0.96	0.95	0.95	36.5	0.94	0.94	0.94	32.3	0.93	0.92	0.92	37.3
V19	0.94	0.94	0.94	22.4	0.94	0.93	0.93	21.8	0.91	0.90	0.91	23.2
V20	0.92	0.94	0.93	30.8	0.90	0.91	0.90	27.7	0.88	0.89	0.88	31.2
V21	0.95	0.95	0.95	23.5	0.93	0.94	0.93	22.6	0.93	0.92	0.92	24.2
V22	0.91	0.92	0.91	19.7	0.91	0.92	0.91	17.2	0.87	0.88	0.87	20.8
V23	0.96	0.93	0.95	20.7	0.93	0.92	0.92	18.5	0.91	0.90	0.91	21.1
Avrg.	**0.94**	**0.93**	**0.94**	**24.6**	**0.92**	**0.91**	**0.92**	**22.6**	**0.90**	**0.89**	**0.89**	**25.6**

From experiments, we noticed that the correlation slightly improves the obtained results in term of accuracy. However, in term of speed, the cosine distance is faster. Also, it can be seen that the Euclidean distance gives the worst results among used distances. Simulations on time processing for CT, GT and overall detection are also reported. Generally, the computational time taken is around 20% of the video duration, which represents an average frequency of 150 fps (i.e. frame par second). Compared to real-time requirement, our algorithm yields less computation time and hence it can be used in real-time applications. In addition, speed can be enhanced by considering optimal implementations of the approach. However, our main objective was to design a competitive method for detecting various types of difficult transitions, particularly those present in the video data used. Samples of misclassifications returned by our algorithm, showing complex transitions, are exposed in Figure 4.4-4.7.

Generally, false detections are returned when abnormal events happen or can be caused by noises and special effects. Most of missed shots are due to similar visual information. One of

FIGURE 4.4: CT false detections, represented in red, caused by high speed motion and camera flashes

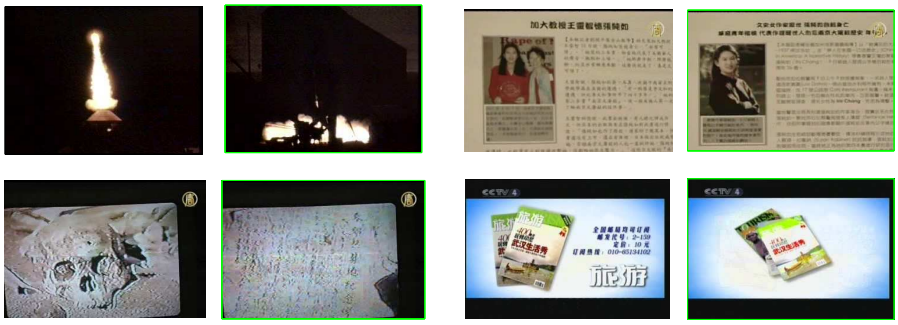


FIGURE 4.5: CT missed shots, in green, due to similar color distribution and background

FIGURE 4.6: GT false detections caused by zoom, special effects and noises

FIGURE 4.7: GT missed shots due to very similar visual content

the major challenges of an SBD method is the false positive caused by camera flashes. The correlation coefficient between features in refined space seems to be immune to most of them, and the coarsest ones (i.e. camera flashes) returned as false detections are illustrated in Figure 4.4. Sometimes it is difficult to detect a gradual transition composed by two shots with very similar visual content. In this case, distances exceed all thresholds using various features. Samples of such cases are illustrated in Figure 4.7.

To prove the efficiency of our method, comparisons with state-of-the-art methods are performed independently for both CT and GT detection. The WHT method [37], proposed in 2014, is selected for comparison as it recently achieved high performances. This last is considered the best according to TRECVid 2007 dataset. Since our approach depends on the SVD, we also measure its performance with the SVD-based method [34], presented in 2013. The results of these comparisons are illustrated in Table 5.3 and Table 5.2, respectively. The highest scores for each evaluation criterion are represented in bold. The method reaches high detection rates, particularly for CTs, with P= 0.97, R= 0.94 and F1= 0.96, which significantly exceeds both SVD [34] and WHT [37]. The obtained results for GTs are also superior, with an average value of 0.90 for the F1 criterion, while for the same metric, the SVD and WHT achieve only 0.81 and 0.87, respectively. It can be seen from Table 5.3, that except for precision criterion P, our method usually surpasses the results obtained by the WHT method. One can notice from Table 5.2, that our algorithm enhances the obtained results of the SVD-based method with an improvement of more than 8% in the overall values of all criteria for both CT and GT detection. Table 4.6 summarizes a last comparison of the overall detection rates, including both CT and GT detection, with both methods. It is worth mentioning that the proposed approach is able to achieve high detection rates in terms of different criteria in detecting several transitions. One can conclude, from this comparative study, that the competitiveness of our technique outperforms existing methods and always gives the best results.

TABLE 4.4: Comparison with the Walsh-Hadamard Transform method

	WHT method [37]						Proposed algorithm					
	CT Transition			**GT Transition**			**CT Transition**			**GT Transition**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
V1	0.94	**0.97**	**0.96**	**0.95**	0.92	**0.93**	**0.97**	0.92	0.95	0.89	**0.96**	0.92
V2	0.85	**0.97**	0.91	0.90	0.87	0.88	**0.91**	0.93	**0.92**	**0.91**	**0.97**	**0.94**
V3	0.86	0.82	0.84	**0.88**	0.85	0.87	**1.00**	**0.92**	**0.96**	**0.88**	**0.88**	**0.88**
V4	0.90	0.88	0.89	**0.84**	0.80	0.82	**0.94**	**0.92**	**0.93**	0.81	**0.96**	**0.88**
V5	0.93	0.95	0.94	**0.87**	0.87	**0.87**	**1.00**	**1.00**	**1.00**	0.85	**0.88**	**0.87**
V6	0.97	0.95	0.96	**0.88**	0.90	0.89	**1.00**	**1.00**	**1.00**	0.87	**0.93**	**0.90**
Av.	0.91	0.92	0.91	**0.88**	0.87	0.87	**0.97**	**0.94**	**0.96**	0.87	**0.93**	**0.90**

TABLE 4.5: Comparison with the SVD based method

	SVD-based method [34]						Proposed algorithm					
	CT Transition			**GT Transition**			**CT Transition**			**GT Transition**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
V2	0.90	0.90	0.90	0.72	0.93	0.81	**0.91**	**0.93**	**0.92**	**0.91**	**0.97**	**0.94**
V3	0.86	0.66	0.75	**0.94**	0.73	0.82	**1.00**	**0.92**	**0.96**	0.88	**0.88**	**0.88**
V4	0.89	0.88	0.89	0.74	0.72	0.73	**0.94**	**0.92**	**0.93**	**0.81**	**0.96**	**0.88**
V6	0.97	0.95	0.96	**0.92**	0.84	0.88	**1.00**	**1.00**	**1.00**	0.87	**0.93**	**0.90**
Av.	0.90	0.84	0.87	0.83	0.80	0.81	**0.96**	**0.94**	**0.95**	**0.87**	**0.93**	**0.90**

TABLE 4.6: Comparison of the overall transition rates

	SVD [34]			**WHT [37]**			**OURS**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
V1	—-	—-	—-	**0.95**	0.92	0.93	0.94	**0.94**	**0.94**
V2	0.80	0.91	0.85	0.87	0.93	0.90	**0.91**	**0.95**	**0.93**
V3	0.91	0.70	0.79	0.87	0.84	0.86	**0.92**	**0.89**	**0.90**
V4	0.84	0.83	0.83	0.88	0.85	0.87	**0.90**	**0.93**	**0.91**
V5	—-	—-	—-	0.91	0.90	0.90	**0.94**	**0.96**	**0.95**
V6	0.95	0.89	0.92	0.92	0.90	0.91	**0.93**	**0.96**	**0.94**
Av.	0.87	0.83	0.84	0.90	0.89	0.89	**0.92**	**0.94**	**0.93**

An efficient approach for video shot detection based on multiple SVD properties has been proposed. The Frobenuis norm and low rank approximation are used to construct $\tilde{k}$-dimensional frame feature vectors, each time depending on each segment. A double thresholding technique is performed to detect CT transitions. This procedure allows a better classification and an effective detection in a reduced time computation. Despite, our main contribution lies in detecting GT transitions. The SVD-updating is used to incorporated the estimated middle frame feature EMFF vector in the singular space. A discontinuity signal which shares a specific curve when a GT segment occurred is then calculated. To distinguish between dynamic segments and GT gradual transitions, a pattern matching is performed. Experimental results show the effectiveness of our algorithm, which gives fulfilling results. Moreover, the proposed method outperforms recent related works in terms of different criteria in both CT and GT detections.

4.3 Summary

In this chapter, a gradual shot detection method is proposed based on SVD and pattern matching. A different modeling is also proposed for different shot transitions. An elaboration of a wipe transition detection method is actually our main interest. The recognition of several transitions in a video sequence is very useful in video and scene understanding. Also, when searching for an image query in a video, generally, the target image will not represent a gradual

transition. Therefore, the search will be performed just for well defined shots. In some cases, key-frame extraction requires shot detection as a first step. Following this, we can use the proposed method to segment the video into shots, then to extract $l \geq 1$ frames from each shot to represent a storyboard. An elimination of redundant frames is sequentially performed to avoid repetition. As our frames are represented in the singular space by the β_i vector, another idea is to group each shot into one class and to calculate the centroid vector β_c. The frame corresponding to the closest vector β_i to β_c is select as a key-frame if it differs from actual key-frames. In several area, features are important and their elaboration differ from application to another. Defining new features allow the possibility to use them in different topics. In our work, we find that the SVD is a powerful tool for features construction in term of accuracy and speed. However, designing a new approach is more interesting and important than defining new features. Generally, good approaches perform well whatever features and metrics used. The results will depend on the architecture of the proposed idea to solve the problematic. For this purpose, a new and different idea for key-frame extraction is designed, as explained in the next chapter.

Chapter 5

Static video summarization based on important scenes

Most of applications resulting from SBD are designed for the video summarization; and as a result, the one that requires the most a good temporal video segmentation. A bad shot detection may jeopardize the expected results for video summarization. The usefulness of this area of research is that it enables fast browsing of large video data and efficient information access. Different from detection, human created summaries do not ensure to reach the best results. Moreover, a common method or a standard procedure for assessing the results of a video summary system is not yet available. Designing a competitive system, without knowing how to evaluate it, was our main challenge.

As mentioned earlier, a video summary (VS) is defined as a sequence of still or moving frames, presenting the video content in reduced and concise information. The essential message of the original video should be preserved. According to [114], two fundamental video summarization methods may be categorized: static video summary, composed by a set of key-frame extracted from the original video, and dynamic video skimming, constructed by set of shots. A systematic review and classification of video abstraction is proposed in [114]. In this penultimate chapter, an idea for representing the essential information contained in a video sequence is discussed. The proposed method consists of three main steps: features extraction via SVD decomposition, dictionary selection, and important scenes selection.

5.1 Key-frame extraction

In this section, we propose a simple and efficient approach for automatic video summarization. The method is based on SVD features and deals with the video by segments. Generally, in a video sequence, consecutive frames within short temporal segments are highly correlated and

sometimes the visual content may not considerably change during a long temporal segment, as in documentaries. Therefore, a video sampling is first performed to eliminate similar frames and to reduce the time processing. In the literature, a frame rate of 5 or 6 frames per second (FPS) is usually used. Also to save computational time, monochromatic frames are removed. In [69], it was referred to meaningless frames as the information contained is not important. A segment of N frames is then formed for features extraction. To verify whether it contains considerable changes, the first and last frames are compared using the cosine distance of their respective feature vectors h_i and h_j. If this distance is greater than T_S, the segment is considered to contain similar visual content, and thus, the next one is processed. Otherwise, a frame feature matrix $H = [h_i]_{1 \leq i \leq N}$ is formed for the current segment by assembling the feature vectors. The SVD is then performed on H to obtain a matrix $A = [\beta_i]_{1 \leq i \leq N}$, in which each column vector β_i represents one frame in the refined space.

5.1.1 Feature representation and SVD decomposition

In a video summary system, features and metrics are significant and should be invariant to illumination changes, camera and object motion. In this work, we adopt the Centrist [115] (Census Transform histograms), which is a visual descriptor for recognizing places or scene categories. Centrist encodes structural properties within a frame while suppressing textural details, and contains rough geometrical information in the scene. In addition, it is easy to implement, has nearly no parameter to tune and evaluates extremely fast. To incorporate spatial information, each frame is first divided into 3×3 blocks, and the centrist features are extracted from each block. These nine features are then concatenated together to form a first part of the feature vector h_i. As Centrist does not capture color information, which is important for VS, three central color moments are also used to compose the second part of our feature vector. Distribution of color in a frame are viewed as a probability distribution, thus, the mean, standard deviation and skewness can be defined:

$$\begin{aligned} &\text{Mean :} && \mu_i = \sum_{j=1}^{NP} \frac{1}{NP} p_{ij} \\ &\text{Standard Deviation :} && \sigma_i = \sqrt{\frac{1}{NP} \sum_{j=1}^{NP} (p_{ij} - \mu_i)^2} \\ &\text{Skewness :} && s_i = \sqrt{\frac{1}{NP} \sum_{j=1}^{NP} (p_{ij} - \mu_i)^3} \end{aligned} \tag{5.1}$$

where p_{ij} is j^{th} pixel of the i^{th} color channel and NP is the number of pixels. The RGB color space is used and color moments are calculated from each image block. In our work, the standard deviation is also used to detect the monochromatic frames. A value of zero or very close to zero means that the frame shares a unique color information and thus it will be removed. Afterward, both Centrist and color moments are normalized, independently, and stacked together to form one combined feature vector, denoted h_i.

Once the feature set is generated, visual content of the current segment is measured. For non static segments, feature vectors of its composing frames are extracted to constitute the matrix $H = [h_1, h_2, ..., h_N]$, where $h_i \in \mathbb{R}^M$ and N is the length of the segment. As in SBD, the matrix $H \in \mathbb{R}^{M \times N}$ is mapped into reduced space via singular value decomposition technique:

$$H = U \Sigma V^T, \tag{5.2}$$

where $U \in \mathbb{R}^{M \times M}$ and $V^T \in \mathbb{R}^{N \times N}$ are composed by left and right singular vectors, respectively. The diagonal matrix $\Sigma = \text{diag}(\sigma_1, \sigma_2, ..., \sigma_N) \in \mathbb{R}^{M \times N}$ contains the singular values. By definition, there are only $r = \text{rank}(H)$ singular values that are nonzero. Therefore, we can obtain $U_r \Sigma_r V_r^T$ as a refined form of H, in which $U_r \in \mathbb{R}^{M \times r}$, $V_r^T \in \mathbb{R}^{r \times N}$ and $\Sigma \in \mathbb{R}^{r \times r}$. The magnitude of a singular value is closely related to the importance of the corresponding singular vectors in the reduced matrices U_r and V_r^T. Generally, the first singular values are much greater than the following ones, thus keeping the first $k \ll r$ largest singular values is equivalent to keeping the important information of the matrix H. The dyadic decomposition is given by:

$$H \simeq H_k = \sum_{i=1}^{k} \sigma_i . u_i . v_i^T = \sum_{i=1}^{k} u_i . \beta_i = U_k A_k. \tag{5.3}$$

Such dimensionality reduction (5.3), called *truncated* SVD, was successfully employed in several previous works including shot boundary detection [109]. The Projection of concatenated features into a refined k-singular space allows to merge them together, thus the new features will be more discriminating. It has been shown in our experiments that the Centrist features can perform well alone. However, mapping such features in singular space turns out to be useful, especially with the inclusion of color information. This makes new features more discriminating, therefore, frames belonging to a same scene are easier to detect. Adaptive features can be used to improve accuracy, as described in [109]. Criteria for selecting the optimal $\tilde{k}$ in VS are different than for SBD since more information are required for scene classification than for detecting transitions.

5.1.2 Fast iterative reconstruction

The purpose of any video summarization system is to extract the pertinent information contained in the original video, where the size of selected data is as small as possible. Moreover, the original video should be well restored from extracted data. Such requirements meet the properties of dictionary selection algorithms. Therefore, VS can be formulated as a dictionary selection problem, where the target dictionary will represent the key-frame set.

Given the matrix A, defined in (5.3), where frames are represented by singular features β_i, we seek for a dictionary X allowing a good reconstruction of the original matrix from a sparse one $B = AX$. Supporting such idea, the actual task is how to select, from the initial set A, the optimal subset $\hat{B} = \{f_{r_1}, f_{r_2}, ..., f_{r_k}\}$, where $\hat{B} \subset A$ and $r_k \in \{1, 2, ..., n\}$ is the number of selected key-frames. Several algorithms have been proposed in the literature to solve such problem, also known as the sparse dictionary selection and defined as follows:

$$\min_X : \frac{1}{2} \|A - AX\|_F^2 + \lambda \|X\|_{1,2} \tag{5.4}$$

where $X \in \mathbb{R}^{N \times N}$ is the pursuit coefficient matrix; $\|.\|_F$ is the Frobenius norm; $\|X\|_{1,2} = \sum_{i=1}^{n} \|X_{i,.}\|_2$, with $\|X_{i,.}\|_2$ is the l_2 norm of the i^{th} row of X, and λ is a regularization parameter. This formulation was originally proposed for abnormal event detection [116]. In previous works [85, 59], the model in (5.4) was developed for key-frame extraction, where sparsity constraints are used to support predefined assumptions toward the optimal set selection. At each iteration of the off-line MSR method [59], the percentage of reconstruction (POR) of all frames is calculated according to the current key-frame set. The frame with the minimum POR is selected as a new key-frame if its POR is lower than a threshold T_{POR}. After each key-frame selection, the POR are updated and the algorithm is repeated until the POR of all frames is higher than T_{POR}. This proves to be very expensive in computational time. In another faster version, once the POR of a frame is calculated, it is compared against the threshold T_{POR}. Depending on the number of key-frames required, T_{POR} can be updated after each selection of a new key-frame. Only one iteration is required which reduces the time processing. Therefore, our optimal set $\hat{B}$ selection is performed based on the following steps:

1. Select the first frame as initial key-frame $\hat{B} = [\beta_1]$.

2. Calculate the POR of the current frame using:

$$POR_i = \frac{R_i}{\|\beta_i\|_2}, \tag{5.5}$$

with R_i represents the reconstruction related to frame f_i:

$$R_i = \|\hat{B}(\hat{B}^T\hat{B})^{-1}\hat{B}^T\beta_i\|_2. \tag{5.6}$$

3. Add a key-frame according to:

$$\text{if } POR_i \begin{cases} \geq T & B \text{ remains unchanged} \\ < T & \text{add key-frame } \hat{B} = [\hat{B}, \beta_i] \end{cases} \tag{5.7}$$

4. Update T_{POR} and repeat until no more frames.

To allow a better reconstruction, depending on T_{POR}, the optimal set $\hat{B}$ may contain quite similar vectors, representing frames belonging to a same scene. This results in selecting insignificant redundant key-frames during a short temporal sequence. As it contains redundant information, the optimal subset $\hat{B}$ will not be considered as our final key-frame set. A clustering algorithm is used to select the most representative key-frames form the optimal set $\hat{B}$ according to their importance, as explained in the next subsection. This gives a diversity to the final set, in which only redundant frames belonging to similar scenes, at different locations in time space, are allowed.

5.1.3 Importance score calculation

The main of dictionary selection algorithms is only to keep vectors which can not be reconstructed well from the current set of vectors already selected. This can be useful as, from key-frame set, we are able to reconstitute the original video. However, this will not ensure to catch the essential information. Maintained features are those different from actual ones, thus, a key-frame is selected only if its visual content differs from other key-frames. Resulting key-frames may not be the most important. Moreover, insignificant redundant successive key-frames may be selected in some cases.

In our approach, after the minimum reconstruction step, k-means algorithm is performed where k the number of selected key-frames. In the literature [117], the standard algorithm consists of three main steps: the initialisation step in which the number k is set, and k initial means are randomly generated within the data. The assignment step, in which k clusters are created by associating every observation with the nearest mean. Finally, the updating step where the centroid of each clusters becomes the new mean. These two last steps are repeated until convergence. One of the limitations of the standard algorithm is the initialization step, in which how to set k the number of clusters and how to generate the first initial means are

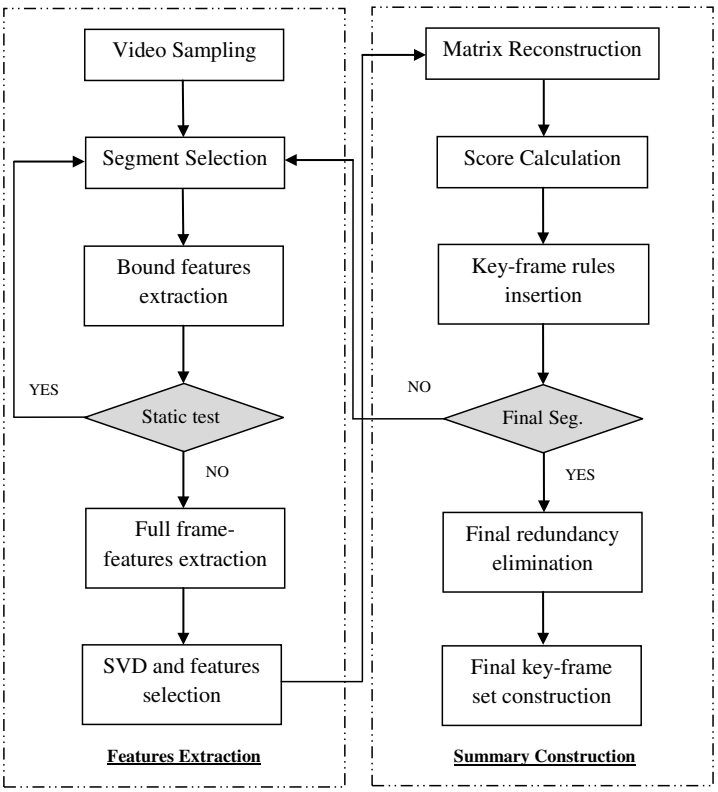


FIGURE 5.1: Processus of the proposed video summary algorithm.

crucial. This may give different results. Therefore, in our work, we consider the k column vectors composing the key-frame set $\hat{B}$ as the k initial entries. Then, only one iteration of the assignment step is performed and the size of each cluster is divided by the size of all data (i.e. length of a segment N) to attribute a score to each cluster or to its corresponding key-frame. A potential candidate key-frame is then characterized by its feature vector, its location and its score s_i. In addition, for each segment, we have:

$$\sum_{i=1}^{k} s_i = 1. \tag{5.8}$$

A candidate key-frame represents the centroid of a cluster, thus, the size of this cluster will reflect the number of similar frames to that key-frame. One can select key-frames with highest scores, however, selecting a fixed number of key-frames is not representative and may lead either in losing pertinent information or in producing redundancy. To avoid this, scores are sorted in descending order: $sm_1 \geq sm_2 \geq ... \geq sm_k$, then we keep the $r < k$ key-frames when the sum of their scores exceeds T_{max}. Key-frames with lowest scores are removed according to:

$$\begin{aligned} \textstyle\sum_{i=1}^{r} sm_i \geq T_{max} \\ sm_i < T_{min} \end{aligned} \tag{5.9}$$

When the sum of scores is higher than a predefined threshold, this means that the current segment can be represented only by those key-frames. The final set depends also on the number of apparition of each key-frame and to its location in time space. If a scene is repeated in short temporal segment, this means that it represents an event or an action. No redundancy is preferred in this case, and thus the current key-frame is not selected. Now, if a scene is repeated several times, at different time location, the scene can be considered as important. Therefore, before adding a key-frame, it is compared not with all current data (i.e. actual key-frames set), but only with recent extracted key-frames, according to their location. A new key-frame is inserted if it differs from its nearest key-frames in the space time.

Based on the discussion above, our key-frame extraction algorithm can be summarized in Figure 5.1. The video sampling and the static segment verification can be considered as post-processing. They compose the first part of our method with the centrist extraction and the SVD decomposition, in which the appropriate features β_i, from A related to (5.3), are calculated. The second part starts from the dictionary selection and constitute the steps of our storyboard construction.

5.2 Discussions and simulations

Although it has long existed and that many methods have been proposed, the major obstacle of this area of research was to find the most appropriate evaluation criteria. In this section where several experiments are performed, we start by introducing the evaluation criteria used.

5.2.1 Evaluation methods

For a long time, a consistent evaluation framework was seriously missing for video summarization research and the evaluation task was usually objective. According to Truong and Venkatesh [2], evaluation techniques can be grouped into three different categories:

- Summary description where generally the impact of parameters used on the resulted key-frames are discussed. Such form of evaluation does not allow comparison with existing techniques.

- Objective metrics can be used to allow comparison of different techniques. However, it is not sure that the metric maps well to human judgement regarding the quality of the summary.
- User evaluation where independent experts judge the quality of resulted summary. This is the most used form of evaluation particulary for methods designed for specific applications where prior information are given.

In 2011, Avila et al. [69], proposed a less objective approach for evaluating static storyboard. In their method, called Comparison of User Summaries (CUS), several users build manually a number of key-frames composing the video summary. The user-created summaries are then considered as the ground-truth. These summaries are finally compared to the results of different techniques, which allows a ranking for competing summarization systems. The quality of a video summary was estimated using two metrics: accuracy rate CUS_A and error rate CUS_E, defined as follows:

$$CUS_A = \frac{N_{ma}}{N_{us}} \tag{5.10}$$

$$CUS_E = \frac{N_{nma}}{N_{us}} \tag{5.11}$$

where N_{ma} is the number of matching key-frame, N_{nma} the number of non-matching key-frame from automatic summary and N_{us} is the number of key-frames from user summary. A value of 0 for the CUS_A means that no key-frames are matched and is considered as the worst case. A value of 1 means that all key-frames in a user summary are generated by the algorithm. Even if it can be considered as the best case (i.e. $CUS_A = 1$), this not ensure that all returned key-frames are present in the user summary. For the second criterion CUS_E, the best value is 0 and occurs when this occurs when $N_{nma} = 0$, which means that all key-frames are present in the user summary. This gives a complementarity to CUS_A and CUS_E highest performance is reached when $CUS_A = 1$ and $CUS_E = 0$.

In their work [69], authors also provide video data and summaries from five users for each video. This was a motivation for many researchers to develop more and more video abstraction techniques [118, 119, 120]. Considering user summaries as a ground-truth, several other quantitative assessments can be employed. A key-frame generated by the algorithm that is not found in any user summary can be considered as a false key-frame. A key-frame composing

TABLE 5.1: Description of the video dataset used

	Dataset 1	Dataset 2	Overall
Number of videos	50	50	100
Total duration (s)	8685	4956	13641
Average duration (s)	174	100	137
Total frames	234495	143724	378219
Resolution	320×240	320×240	320×240
Frame rate (fps)	25-29	29	25-29
Frame sampling (fps)	5	5	5

user summaries and not returned by the algorithm is considered as missed key-frame. Following this, and similarly to SBD, our evaluation for VS is based on three metrics including precision, recall and F1-score as defined:

$$\text{Precision P} = \frac{N_{match}}{N_{AS}}, \tag{5.12}$$

$$\text{Recall R} = \frac{N_{match}}{N_{US}}, \tag{5.13}$$

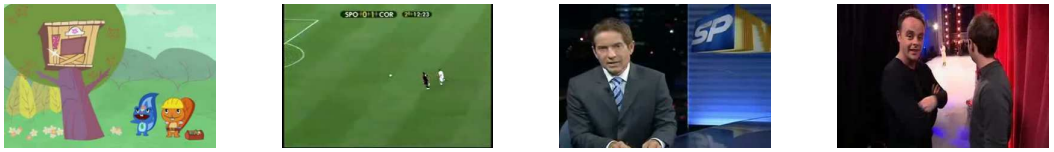
$$\text{Combined measure F1} = \frac{2 \times \text{P} \times \text{R}}{\text{P} + \text{R}}, \tag{5.14}$$

where N_{match} is the number of common key-frames (i.e. matching key-frames) between automatic and user summaries, N_{AS} is the number of key-frames in the automatic summary and N_{US} is the number of key-frames in a user summary. By definition, the precision is the percentage of matched key-frames from the generated ones. It reveals the false returned key-frames, and it is high when the number of false alarm is low. The recall shows the percentage of matched key-frames from the ground truth. It reveals the number of missed key-frames from the automatic summary. Lesser the missed key-frames, higher the recall. The combined measure F1, considered as the harmonic average of both metrics (i.e. precision and recall), allows to evaluate the overall performance of the video summary. The higher these ratio are, the better the performance.

5.2.2 Results and comparisons

Various simulations and tests over different video sequences are carried out to prove the effectiveness and diversity of our video summary. Our experiments are performed on two benchmark datasets: A first dataset taken from the Open Video Project [107] and a second one provided by [69]. Each one contains 50 videos including several types (news, cartoons, commercials, sports and tv-shows). Their durations vary from 1 to 10 minutes, with a total video

(A) Frames belonging to the first dataset



(B) Frames belonging to the second dataset

FIGURE 5.2: Example of frames contained in the video database used

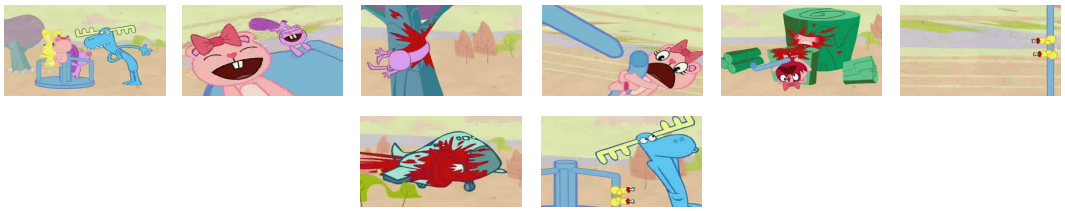
duration of 4h as described in Table 5.1. Figure 5.2 illustrates frames belonging to the video sequences used. The proposed approach is compared with sparse dictionary (SD) [85], VSUMM [69], Open Video Project storyboard (OVP) [107], Delaunay Clustering (DT) [121], STIMO [122] and the Minimum Sparse Reconstruction (MSR) [59] approaches. For comparison purposes, all video sequences are sampled at 5 fps and reformed into the uncompressed MPEG-1 format with a resolution of 320×240 pixels. The average results of the first dataset are reported in Table 5.2. The obtained results and comparisons with the state-of-the-art methods are ranked by the overall F1-metric. Recorded performance results of these approaches are adopted from [69, 59]. It can be seen from Table 5.2, that the results are relatively close, however our approach achieves the best performance among all compared methods according to the F1 evaluation criterion. The obtained results and comparisons of the second dataset are illustrated in Table 5.3. It may be noted that the VSUMM reaches a very high value of the recall criterion, however, the precision rate is low, and thus resulting in a weak average of the F1 measure. Both Table 5.2 and 5.3 show the competitiveness of the proposed method.

TABLE 5.2: Comparisons with related works for the first video dataset

	SD	DT	STIMO	OVP	VSUMM	MSR	OUR VS
Precision	40	47	39	43	48	58	**59**
Recall	61	50	**65**	64	63	58	61
F1-score	48	48	49	51	54	58	**60**

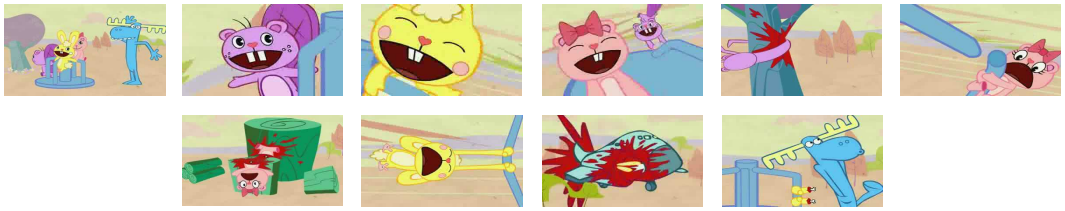
TABLE 5.3: Comparisons with related works using the second video dataset

	SD	MSR	VSUMM	OUR VS
Precision	37	52	38	**55**
Recall	53	45	**72**	56
F1-score	44	48	50	**55**



(A) One user video summary

(B) VSUMM



(C) Our video summary

FIGURE 5.3: Sample video summarization results of one user, the VSUMM and our method for the 1st video of the first dataset.

For objective evaluation, several video summaries from different video categories are illustrated to show the diversity of our summaries. The user and VSUMM summaries are added for visual comparisons. Figure 5.3-5.7 illustrate extracted key-frames from carton, news, sport, commercial and tv-show categories. It can be observed that the VSUMM provide redundant key-frames in a short period of time, which is not the case of our algorithm.

In Figure 5.3b, the two first key-frames are quite similar. The sixth and seventh frames in Figure 5.4b also share a similar visual content. The only similar key-frames generated by our method are relatively far from each other in their locations. In Figure 5.7c, the second key-frame is quite similar to the fifth one, however, that scene is repeated at different temporal locations. In our perception, when a scene is repeated several times at different time positions, it can be considered as important. In addition, the obtained results show the diversity of our summaries which produce quite unique information and very few redundant key-frames. Almost all key-frames proposed by users are generated by our method. Also, all additional key-frames share unique information. Based on this, it can be concluded that the proposed algorithm generates summaries of good quality for different type of video categories.

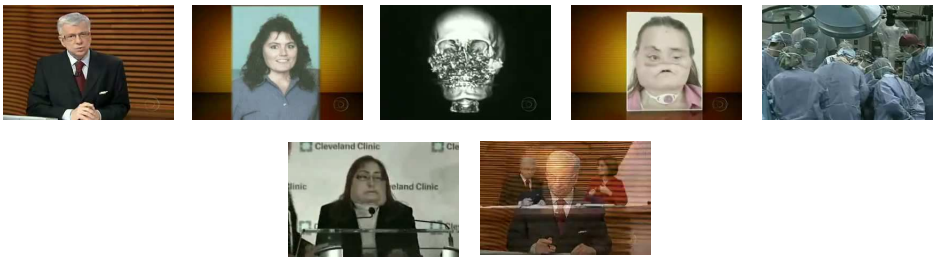
(A) One user video summary

(B) VSUMM

(C) Our video summary

FIGURE 5.4: One user, the VSUMM and our VS for commercial video.

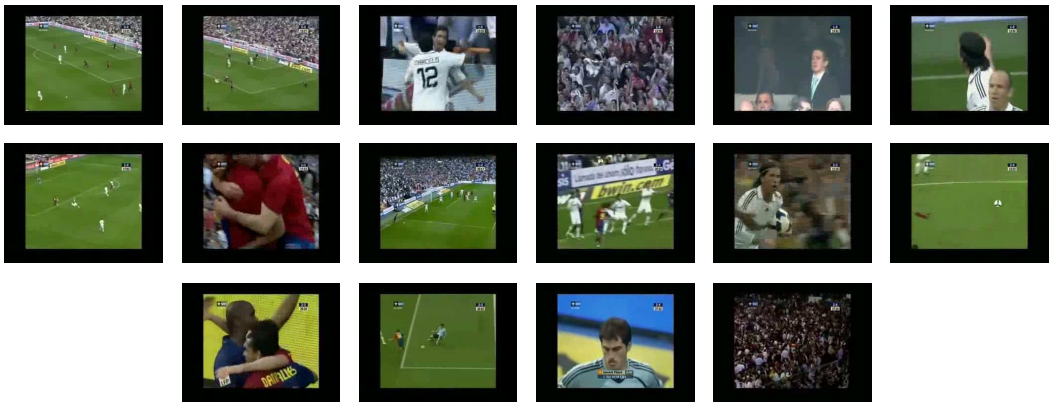
(A) One user video summary



(B) VSUMM

(C) Our video summary

FIGURE 5.5: Video summarization results of one user, VSUMM and our method for a news video from the second dataset.



(A) One user video summary

(B) VSUMM

(C) Our video summary

FIGURE 5.6: One user summary, VSUMM and our VS for sport video from the second dataset.

A new approach for static video summarization based on important scene was proposed. The singular value decomposition of the centrist gives more refined features allowing a better scene classification. Our global key-frame construction is based on local representative video frames extraction. A first selection of representative frames is achieved by minimum reconstruction. Then, while most existing methods remove similar frames to share unique information, we favor selecting the most important scenes. Therefore, before removing redundant key-frames, a score is calculated for each scene to define the most important ones via modified k-means clustering algorithm. The final video summary is regularized using the score of importance and the location of each candidate key-frames. Experimental results and comparisons have shown the effectiveness of our method and the diversity of our summaries.

(A) One user video summary

(B) VSUMM

(C) Our video summary

FIGURE 5.7: Sample video summarization results of one user, the VSUMM and our method for the 46th video of the first dataset.

5.3 Summary

Many ideas were proposed for video summarization using shot boundary detection first. Therefore, several enhancement can be produced using SVD features and proposed solutions for transitions identification. A video summarization algorithm is generally designed for a specific application and may work for a limited type of video. The main challenge faced by video summarization systems is the evaluation of generated summaries. Although no standard framework exists for evaluation, great efforts were made during this decade leading to several ideas and propositions. In our study, we follow the work presented in [69], in which manual ground truth summaries created by different users is proposed. When manually creating our summaries, we can usually get moderate results according to evaluation criteria. Also, we have noticed that a user may include repetitive frames from a same scene. Intuitively, when seeing information several times, our brain may judge it as important. Generally, when composing summaries, users can remove redundant information. From our point of view, the more a scene is repeated, the more important it will be. Based on this, we designed a new approach, in which extracting important scenes is favored.

Following those works and progressions on the field, we are considering to present o similar model using other different and significant constraints. The first one will focus on temporal video segmentation into scenes and then to summarize each scene independently. Subsequently, a second modeling will aim to bring together the most important scene to constitute the summary. In addition, we project to solve the dictionary selection problem via other iterative algorithms. The optimization problem can be converted into a proximal gradient problem, and thus solved using a fast iterative shrinkage thresholding. Another approximate iterative algorithm can be developed using the K-SVD or the sparse K-SVD. Doing so may be interesting as it comes in continuity with our research, where the SVD has been already used for temporal video partitioning.

Chapter 6

Conclusions

The main motivation of this study was to allow content-based video retrieval and to enable effective indexing of visual information stored in large video data. Such area of research are very useful in saving both memory cost and time processing. This thesis presented work in the fields of video shot boundary detection and key-frame extraction using mathematical tools.

6.1 Thesis summary

Shot boundary detection (SBD) has the longest and richest history in the area of CBVR. This is not surprising that, sometimes, it was difficult to find a better solution than those proposed in the literature. Our first objective was to propose algorithms for cut detection only, which is considered as the first step in CBVR. Good performances were achieved when comparing to state-of-the-art methods. However, as discussed in Chapter 2, a robust SBD method needs to fulfill the following two criteria:

- It should detect different types of transitions.
- Works for any arbitrary video sequence.

Gradual shot detection, which is a more challenging and difficult task, was missing to our first works. For this purpose, a unified approach for detecting both cut and gradual transitions, based on adaptive features, was designed. Singular value decomposition (SVD) is a powerful tool for features selection and may lead to major improvement. Experiments and comparisons with recent and accurate methods were performed to show the robustness of our system.

During our work, several ideas could have been used for local video summarization based on proposed SBD methods. A challenging task was to present a global key-frame extraction based on SVD features. Representative frames are first selected via dictionary selection algorithm. Before removing redundant elements, a score is calculated for each key-frame to define

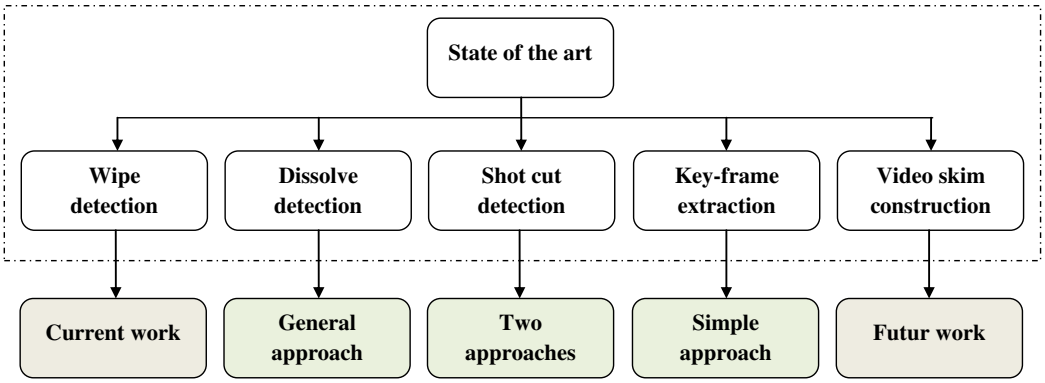


FIGURE 6.1: Summary of proposed work, ongoing studies and future research.

the most important. The final storyboard is regularized using the location of each candidate key-frames. Our current work is focused on developing a new video summarization scheme based on iterative algorithms giving optimal solutions to present both static and dynamic video abstraction. Concerning SBD, we intent to set up a technique for wipe transition detection. Figure 6.1 summarizes the works presented in this thesis.

6.2 Concluding remarks

When working on shot boundary detection area, researchers will usually identify the key-frame extraction topic. In this thesis, our first interest was to examine the video summarization. While reviewing the different classifications of existing methods, we have notice the existence of temporal video segmentation and the several types of transitions. It took time to identify the SBD problem, but at least, it remains a subject where the evaluation criteria are objective. The field of video abstraction lacks of objective evaluation and methods are, most of the time, evaluated by humans.

During our research, we noticed that several existing techniques can be improved, however the main of our study was originality. Now the question if this work has been successful or not can be answered only after the algorithms and ideas proposed are integrated into an automatic video indexing system. The author hopes that the techniques designed here will be incorporated into commercial application for video researchers doing real production work.

Bibliography

[1] R. Lienhart. Reliable dissolve detection. *in Proc. SPIE Storage and Retrieval for Media Databases 2001*, 4315.

[2] T. B. Truong and S. Venkatesh. Video abstraction: A systematic review and classification. *ACM Trans. Multimedia Comput. Commun. Appl.*, 3(1), February 2007.

[3] A. Hanjalic. Shot-boundary detection: unraveled and resolved? *Circuits and Systems for Video Technology, IEEE Transactions on*, 12(2):90–105, 2002.

[4] H. Yoo, H. Ryoo, and D. Jang. Gradual shot boundary detection using localized edge blocks. *Multimedia Tools and Applications*, 28(3):283–300, 2006.

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

[6] J. Yu and M. D. Srinath. An efficient method for scene cut detection. *Pattern Recognition Letters*, 22:1379–1391, 2001.

[7] P. S. Mittalkod and G. N. Srinivasan. Shot boundary detection : an improved algorithm. *International journal of enginnering Sciences Research*, 4:8504–8512, 2013.

[8] G. Lupatini, C. Saraceno, and R. Leonardi. Scene break detection : A comparison. *International Workshop on Research Issues in Data Engineering, IEEE Computer Society*, 0:34, 1998.

[9] R. Zabih, J. Miller, and K. Mai. A feature-based algorithm for detecting and classifying scene breaks. *ACM Multimedia 95*, pages 189–200, 1995.

[10] R. Lienhart. Comparison of automatic shot boundary detection algorithms. *Proc. IS&T/SPIE Storage and Retrieval for Image and Video Databases VII*, 3656:290–301, 1999.

[11] P. Alvaro. Simple and robust hard cut detection using interframe differences. *In Progress in Pattern Recognition Image Analysis and Applications Springer Berlin Heidelberg*, pages 409–419, 2005.

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

[13] I. Koprinska and S. Carrato. Temporal video segmentation: A survey. *Signal Processing: Image Communication*, 16(5):477 – 500, 2001.

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

[15] K. Otsuji, Y. Tonomura, and Y. Ohba. Video browsing using brightness data. *Proc. SPIE*, 1606:980–989, 1991.

[16] B. Shahraray. Scene change detection and content-based sampling of video sequences. volume 2419, pages 2–13, 1995.

[17] A. Nagasaka and Y. Tanaka. Automatic video indexing and full-video search for object appearances. *Proceedings of the Second Working Conference on Visual Database Systems*, pages 113–127, 1991.

[18] H. Ueda, T. Miyatake, and S. Yoshizawa. Impact: An interactive natural-motion-picture dedicated multimedia authoring system. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '91, pages 343–350, New York, NY, USA, 1991. ACM.

[19] F. Arman, A. Hsu, and M. Chiu. Feature management for large video databases. volume 1908, pages 2–12, 1993.

[20] J. Meng, Y. Juan, and S. F. Chang. Scene change detection in a mpeg compressed video sequence. *Digital Video Compression: Algorithms and Technol.*, (6):14–25, 1995.

[21] H. C. Liu and G. L. Zick. Scene decomposition of mpeg compressed video. *Digital Video Compression: Algorithms and Technol.*, pages 26–37, 1995.

[22] I. K. Sethi and N. Patel. A statistical approach to scene change detection. *Storage and Retrieval for Image and Video Databases III*, pages 329–338, 1995.

[23] H. Zhang, C. Y. Low, and S. W. Smoliar. Video parsing and browsing using compressed data. *Multimedia Tools and Applications*, 1(1):89–111, 1995.

[24] N. Patel and I. K. Sethi. Compressed video processing for cut detection. *IEE Proceedings - Vision, Image and Signal Processing, volume=143, number=5, pages=315-323, year = 1996*.

[25] H.W. Yoo, H.J. Ryoo, and D.S. Jand. Gradual shot boundary detecion using localized edge blocks. *Multimedia Tools Appl*, 28(3):283–300, 2006.

[26] D. Adjeroh, M.C Lee, N. Banda, and U. Kandaswamy. Adaptive edge-oriented shot boundary detection. *EURASIP J. Image Video Process*, 2009:5:1–5:13, 2009.

[27] A. Kankanhalli H. Zhang and S. W. Smoliar. Automatic partitioning of full-motion video. *ACM Multimedia Systems*, 1(1):10–28, 1993.

[28] A. Hampapur, T. Weymouth, and R. Jain. Digital video segmentation. In *Proceedings of the Second ACM International Conference on Multimedia*, MULTIMEDIA '94, pages 357–364, New York, NY, USA, 1994. ACM.

[29] A. Hampapur, R. Jain, and T. E. Weymouth. Production model based digital video segmentation. pages 111–153, 1996.

[30] U. Gargi, R. Kasturi, and S. H. Strayer. Performance characterization of video shot change detection methods. *Circuits and Systems for Video Technology, IEEE Transactions on*, 10(1):1–13, 2000.

[31] G. G. L. Priya and S. Domnic. Video cut detection using block based histogram differences in rgb color space. *International Conference on Signal and Image Processing (ICSIP)*, pages 29–33, 2010.

[32] Y. N. Li, Z. M. Lu, and X. M. Niu. Fast video shot boundary detection framework employing pre-processing techniques. *Image Processing, IET*, 3(3):121–134, 2009.

[33] Z. Cernekova, C. Kotropoulos, and I. Pitas. Video shot-boundary detection using singular-value decomposition and statistical tests. *Journal of Electronic Imaging*, 16(4):043012, 2007.

[34] Z. M. Lu and Y. Shi. Fast video shot boundary based on svd and pattern matching. *IEEE Transactions on Image Processing*, 22(12):5136–5145, 2013.

[35] O. Küçüktunç, U. Güdükbay, and Ö. Ulusoy. Fuzzy color histogram-based video segmentation. *Computer Vision and Image Understanding*, 114(1):125–134, 2010.

[36] R. Dadashi and H. R. Kanan. Avcd-fra: A novel solution to automatic video cut detection using fuzzy-rule-based approach. *Computer Vision and Image Understanding*, 117:807–817, 2013.

[37] G.G.L. Priya and S. Dominic. Walsh-hadamard transform kernel-based feature vector for shot boundary detection. *Image Processing, IEEE Transactions on,*, 23(12):5187–5197, 2014.

[38] A. F. Smeaton, P. Over, and A. R. Doherty. Video shot boundary detection: Seven years of {TRECVid} activity. *Computer Vision and Image Understanding*, 114(4):411 – 418, 2010.

[39] A. Whitehead, P. Bose, and R. Laganiere. Feature based cut detection with automatic threshold selection. *CIVR*, pages 410–418, 2004.

[40] Sarah Victoria Porter. Video segmentation and indexing using motion estimation. *PhD Thesis*, University of Bristol, 2004.

[41] R. Lienhart and A. Zaccarin. A system for reliable dissolve detection in videos. *Proceedings 2001 International Conference on Image Processing*, 3:406–409, 2001.

[42] B. T. Truong, C. Dorai, and S. Venkatesh. New enhancements to cut, fade, and dissolve detection processes in video segmentation. *Proceedings of the Eighth ACM International Conference on Multimedia*, pages 219–227.

[43] W. Zheng, J. Yuan, H. Wang, F. Lin, and B. Zhang. A novel shot boundary detection framework. *Visual Communications and Image Processing 2005*, 5960:181–191, 2005.

[44] C. N. Canagarajah W. A. C. Fernando and D. R. Bull. Fade-in and fade-out detection in video sequences using histograms. *IEEE International Symposium on Circuits and Systems*, pages 709–712, May 28-31 2000.

[45] C. Ngo, T. Pong, and T. R. Chin. Video partitioning by temporal slice coherency. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(8):941–953, 2001.

[46] U. Naci and A. Hanjalic. Tu delft at trecvid 2005: Shot boundary detection. *Proc. TRECVID 2005 Workshop*, 78:80–82, 2005.

[47] S. Li and M. Lee. Effective detection of various wipe transitions. *IEEE Trans. Cir. and Sys. for Video Technol.*, 17(6):663–673, June 2007.

[48] H. H. Yu and W. Wolf. A hierarchical multiresolution video shot transition detection scheme. *Computer Vision and Image Understanding*, 75(1):196 – 213, 1999.

[49] A. M. Alattar. Detecting and compressing dissolve regions in video sequences with a dvi multimedia image compression algorithm. 1:13–16, May 1993.

[50] A. M. Alattar. Detecting fade regions in uncompressed video sequences. 4:3025–3028, Apr 1997.

[51] J. Nam and A.H. Tewfik. Detection of gradual transitions in video sequences using b-spline interpolation. *Multimedia, IEEE Transactions on*, 7(4):667–679, 2005.

[52] R.A. Joyce and B. Liu. Temporal segmentation of video using frame and histogram space. *Multimedia, IEEE Transactions on*, 8(1):130–140, 2006.

[53] P.P. Mohanta, S.K. Saha, and B. Chanda. A model-based shot boundary detection technique using frame transition parameters. *Multimedia, IEEE Transactions on*, 14(1):223–233, 2012.

[54] D. Geerts, P. César, and M. Obrist. Interaction design for online video and television. In *CHI Extended Abstracts*, pages 959–960. ACM, 2016.

[55] C. Zhan, Y. Cai, N. S., C. Qiu, Y. Cui, and X. Gao. Saliency based wireless capsule endoscopy video abstract. *CISP-BMEI*, pages 1423–1428, 2016.

[56] K. Darabi and G. Ghinea. User-centred personalised video abstraction approach adopting SIFT features. *Multimedia Tools Appl.*, 76(2):2353–2378, 2017.

[57] H. S. Chang, S. Sull, and S. U. Lee. Efficient video indexing scheme for content-based retrieval. *IEEE Trans. Circuits Syst. Video Techn.*, 9(8):1269–1279, 1999.

[58] A. Hanjalic and H. Zhang. An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis. *IEEE Trans. Circuits Syst. Video Techn.*, 9(8):1280–1289, 1999.

[59] S. Mei, G. Guan, Z. Wang, S. Wan, M. He, and D. D. Feng. Video summarization via minimum sparse reconstruction. *Pattern Recogn.*, 48(2):522–533, February 2015.

[60] B. Shahraray and D. C. Gibbon. Automatic generation of pictorial tanscripts of video programs. *in: Proc. SPIE*, 2417:512–518, 1995.

[61] C. Panagiotakis and G. Tziritas A. Doulamis. Equivalent key frames selection based on iso-content principles. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(3):447–451, 2009.

[62] M. M. Yeung and B. Liu. Efficient matching and clusering of video shots. *IEEE International Conference on Image Processing (ICIP)*, 1:338–341, 1995.

[63] H. J. Zhang, J. Wu, D. Zhong, and S. W. Smoliar. An integrated system for content-based video retrieval and borwsing. *Pattern Recognition*, 30(4).

[64] A. Das R. Panda and A. K. Roy-Chowdhury. Embedded sparse coding for summarizing multi-view videos. *IEEE International Conference on Image Processing (ICIP)*, pages 191–195, 2016.

[65] F. Essannouni Y. Hadi and R. O. Thami. Video summarization by k-medoid clustering. *Proceedings of the ACM Symposium on Applied Computing (SAC)*, pages 1400–1401, April 2006.

[66] J. Calic and E. Izuierdo. Efficient key-frame extraction and video analysis. In *Proceedings. International Conference on Information Technology: Coding and Computing*, pages 28–33, April 2002.

[67] X. Zhang, T. Liu, K. Lo, and J. Feng. Dynamic selection and effective compression of key frames for video abstraction. *Pattern Recognition Letters*, 24(9-10):1523–1532.

[68] P. Mundur, Y. Rao, and Y. Yesha. Keyframe-based video summarization using delaunay clustering. *International Journal on Digital Libraries*, 6(2):219–232, April 2006.

[69] S. E. F. Avila, A. P. B. Lopes, A. Luz, and A. A. Araujo. Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, 32(1):56–68, 2011.

[70] Y. Zhuang, Y. Rui, T. S. Huang, and S. Mehrotra. Adaptive key frame extraction using unsupervised clustering. *International Conference on Image Processing (ICIP)*, 1:866–870, 1998.

[71] C. Ngo, Y. Ma, and H. Zhang. Video summarization and scene detection by graph modeling. *IEEE Trans. Circuits Syst. Video Techn.*, 15(2):296–305, 2005.

[72] A. Divakaran, K. A. Peker, and R. Radhakrishnan. Motion activity-based extraction of key-frames from video shots. In *Proceedings of the International Conference on Image Processing (ICIP)*, pages 932–935, September 2002.

[73] N. D. Doulamis A. D. Doulamis and S. D. Kollias. A fuzzy video content representation for video summarization and content-based retrieval. *Signal Processing*, 80(6):1049–1067, 2000.

[74] H. Lee and S. Kim. Iterative key frame selection in the rate-constraint environment. *Sig. Proc.: Image Comm.*, 18(1):1–15, 2003.

[75] T. Liu, X. Zhang, J. Feng, and K. Lo. Shot reconstruction degree: a novel criterion for key frame selection. *Pattern Recognition Letters*, 25(12):1451–1457, 2004.

[76] A. Hanjalic and H. Zhang. An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(8):1280–1289, 1999.

[77] Y. Gong and L. Xin. Video summarization using singular value decomposition. *Proceedings in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2:174–180, 2000.

[78] A. Nasreen, K. Roy, K. Roy, and G. Shobha. Key frame extraction and foreground modelling using k-means clustering. In *7th International Conference on Computational Intelligence, Communication Systems and Networks, CICSyN 2015, Riga, Latvia, June 3-5, 2015*, pages 141–145, 2015.

[79] S. K. Kuanar, R. Panda, and A. S. Chowdhury. Video key frame extraction through dynamic delaunay clustering with a structural constraint. *J. Visual Communication and Image Representation*, 24(7):1212–1227, 2013.

[80] A. M. Ferman and A. M. Tekalp. Two-stage hierarchical video summary extraction to match low-level user browsing preferences. *IEEE Trans. Multimedia*, 5(2):244–256, 2003.

[81] Z. Sun, K. Jia, and H. Chen. Video key frame extraction based on spatial-temporal color distribution. In *4th International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2008), Harbin, China, 15-17 August 2008, Proceedings*, pages 196–199, 2008.

[82] L. Liu and G. Fan. Combined key-frame extraction and object-based video segmentation. *IEEE Trans. Circuits Syst. Video Techn.*, 15(7):869–884, 2005.

[83] X. Song and G. Fan. Joint key-frame extraction and object segmentation for content-based video analysis. *IEEE Trans. Circuits Syst. Video Techn.*, 16(7):904–914, 2006.

[84] J. Calic and E. Izquierdo. Efficient key-frame extraction and video analysis. In *2002 International Symposium on Information Technology (ITCC 2002), 8-10 April 2002, Las Vegas, NV, USA*, pages 28–33, 2002.

[85] Y. Cong, J. Yuan, and J. Luo. Towards scalable summarization of consumer videos via sparse dictionary selection. *IEEE Transactions on Multimedia*, 14(1):66–75, 2012.

[86] P. M. Fonseca and F. Pereira. Automatic video summarization based on MPEG-7 descriptions. *Sig. Proc.: Image Comm.*, 19(8):685–699, 2004.

[87] Z. Li, G. M. Schuster, A. K. Katsaggelos, and B. Gandhi. Rate-distortion optimal video summary generation. *IEEE Trans. Image Processing*, 14(10):1550–1560, 2005.

[88] I. Otsuka, K. Nakane, A. Divakaran, K. Hatanaka, and M. Ogawa. A highlight scene detection and video summarization system using audio feature for a personal video recorder. *IEEE Trans. Consumer Electronics*, 51(1):112–116, 2005.

[89] Y. Gao, W. Wang, and J. Yong. A video summarization tool using two-level redundancy detection for personal video recorders. *IEEE Trans. Consumer Electronics*, 54(2):521–526, 2008.

[90] M. Tavassolipour, M. Karimian, and S. Kasaei. Event detection and summarization in soccer videos using bayesian network and copula. *IEEE Trans. Circuits Syst. Video Techn.*, 24(2):291–304, 2014.

[91] C. M. Taskiran, Z. Pizlo, A. Amir, D. B. Ponceleon, and E. J. Delp. Automated video program summarization using speech transcripts. *IEEE Trans. Multimedia*, 8(4):775–791, 2006.

[92] Y. Gong. Summarizing audiovisual contents of a video program. *EURASIP J. Adv. Sig. Proc.*, 2003(2):160–169, 2003.

[93] G. Evangelopoulos, A. Zlatintsi, G. Skoumas, K. Rapantzikos, A. Potamianos, P. Maragos, and Y. S. Avrithis. Video event detection and summarization using audio, visual and text saliency. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP, Taipei, Taiwan*, pages 3553–3556, April 2009.

[94] C. M. Taskiran, J. Chen, A. Albiol, L. Torres, C. A. Bouman, and E. J. Delp. Vibe: a compressed video database structured for active browsing and search. *IEEE Trans. Multimedia*, 6(1):103–118, 2004.

[95] A. Girgensohn and J. S. Boreczky. Time-constrained keyframe selection technique. *Multimedia Tools Appl.*, 11(3):347–358, 2000.

[96] Y. Geng, D. Xu, and S. Feng. Hierarchical video summarization based on video structure and highlight. In *Structural, Syntactic, and Statistical Pattern Recognition, Joint IAPR International Workshops, SSPR 2006 and SPR 2006, Hong Kong, China, August 17-19, 2006, Proceedings*, pages 226–234, 2006.

[97] G. Ciocca and R. Schettini. Supervised and unsupervised classification post-processing for visual video summaries. *IEEE Trans. Consumer Electronics*, 52(2):630–638, 2006.

[98] R. M. Jiang, A. H. Sadka, and D. Crookes. Hierarchical video summarization in reference subspace. *IEEE Trans. Consumer Electronics*, 55(3):1551–1557, 2009.

[99] R. Lu, H. Yang, J. Zhu, S. Wu, J. Wang, and D. Bull. Hierarchical video summarization with loitering indication. In *2015 Visual Communications and Image Processing, VCIP 2015, Singapore, December 13-16, 2015*, pages 1–4, 2015.

[100] Y. Bendraou, F. Essannouni, D. Aboutajdine, and A. Salam. Video shot boundary detection method using histogram differences and local image descriptor. *2nd World Conference on Complex Systems, WCCS 2014, Agadir, Morocco, November 10-12, 2014*, pages 665–670, 2014.

[101] Y. Bendraou, F. Essannouni, D. Aboutajdine, and A. Salam. Video cut detection method based on a 2d luminance histogram using an appropriate threshold and a post processing. *Wseas Transactions on Signal Processing*, 11:99–106, 2015.

[102] Y. Bendraou, F. Essannouni, D. Aboutajdine, and A. Salam. Video cut detector via adaptive features using the frobenius norm. pages 380–389, 2016.

[103] W. A. Stahel E. Limpert and M. Abbt. Log-normal distributions across the sciences: Keys and clues. *BioScience*, 51(5):341–352, 2001.

[104] D. G. Lowe. Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 2:1150–1157, 1999.

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

[106] *Video dataset*, [Online]:Available at : http://www.site.uottawa.ca/∼laganier/videoseg/.

[107] *Open-Video Project*, [Online]:Available at : http://www.open–video.org/.

[108] G. H. Golub and C. F. Loan. *Matrix Computations*, volume 3rd ed. The Johns Hopkins University Press, 1996.

[109] Y. Bendraou, F. Essannouni, D. Aboutajdine, and A. Salam. Shot boundary detection via adaptive low rank and svd-updating. *Computer Vision and Image Understanding*, 161:20–28, August 2017.

[110] National Institute of Standards and Technology (NIST). *TRECVid Dataset*, [Online]:Available at : http://trecvid.nist.gov/.

[111] P. Over, T. Ianeva, W. Kraaij, and A. F. Smeaton. Trecvid 2005 - an overview. 2005.

[112] *The Internet Archive*, [Online]:Available at : https://archive.org/details/movies.

[113] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, pages 321–330, 2006.

[114] B. T. Truong and S. Venkatesh. Video abstraction: A systematic review and classification. *ACM Trans. Multimedia Comput. Commun. Appl.*, 3(1), February 2007.

[115] W. Jianxin and M. R. James. Centrist: A visual descriptor for scene categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1489–1501, 2011.

[116] Y. Cong, J. Yuan, and J. Liu. Sparse reconstruction cost for abnormal event detection. *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3449–3456, June 2011.

[117] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.

[118] N. Ejaz, T. B. Tariq, and S. W. Baik. Adaptive key frame extraction for video summarization using an aggregation mechanism. *Journal of Visual Communication and Image Representation*, 23(7):1031 – 1040, 2012.

[119] J. Almeida, N. J. Leite, and R. S. Torres. Vison: Video summarization for online applications. *Pattern Recognition Letters*, 33(4):397–409, 2012.

[120] N. Ejaz, I. Mehmood, and S. W. Baik. Efficient visual attention based framework for extracting key frames from videos. *Signal Processing: Image Communication*, 28(1):34 – 44, 2013.

[121] P. Mundur, Y. Rao, and Y. Yesha. Keyframe-based video summarization using delaunay clustering. *International Journal on Digital Libraries*, 6(2):219–232, 2006.

[122] M. Furini, F. Geraci, M. Montangero, and M. Pellegrini. Stimo: Still and moving video storyboard for the web scenario. *Multimedia Tools and Applications*, 46(1):47–69, 2010.