Nº d'ordre : 3456

THÈSE

PRÉSENTÉE A

L'UNIVERSITÉ BORDEAUX I

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

Par **Jan NESVADBA**

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : INFORMATIQUE

Segmentation Sémantique des Contenus Audio-Visuels

Soutenue le : 5 Novembre 2007

Après avis de rapporteurs :

Jan Biemond …….	Professeur
Riccardo Leonardi ……..	Professeur

Devant de la commission d'examen compose de :

Myriam De Sainte-Catherine	Professeur ……..	President
Jan Biemond …….	Professeur ……..	Rapporteur
Riccardo Leonardi …….	Professeur ……..	Rapporteur
Jenny Benois-Pineau ……	Professeur …….	Examinateur
Ioannis Patras …….	Ass. Professor……..	Examinateur
Nozha Boujemaa …….	Dir. Recherche INRIA	Examinateur

- 2007 -

Acknowledgments:

It was my dream to further elaborate my technical knowledge and to have the honour to elaborate this thesis. Many people supported me to achieve this aim and I would like to thank all of them for their kind and constant support, but also in their trust in me.

I would like to thank all my relatives, especially my parents, my wife, sister, grandparents, uncles, aunts and all other family members for their trust and their support.

Furthermore, I would like to thank my PhD mentor Prof. Jenny Benois-Pineau for her constant and highly appreciated support.

I did this work next to my normal work at Philips Research, The Netherlands. Hence, I would also thank to all of my colleagues, students, and collaborating academic partners for their kind collaboration, which I appreciated very much and hope to continue in the future.

Résumé:

Compte tenu des évolutions techniques concernant les nouveaux appareils de stockage personnels et de l'émergence des connexions hauts débit facilitant l'accès à une grande quantité de contenus multimédia sur Internet, les consommateurs veulent de plus en plus pouvoir gérer de façon intuitive et ordonnée ces très grandes collections de vidéos. Les données vidéo brutes contenues dans ces contenus ne fournissent pas d'informations appropriées pour un accès sémantique (métadonnées). Or l'utilisateur a besoin de données de plus haut niveau afin de pouvoir caractériser au mieux sa base de données vidéo, pour cela il faut ainsi générer des métadonnées qui décrivent chaque élément de contenu au niveau non seulement de ses propriétés mais aussi de sa sémantique. Les consommateurs, par exemple, désirent accéder rapidement à des sous-évènements individuels d'un contenu multimédia. Cette thèse présente une méthode de segmentation du contenu audiovisuel en sections significatives, appelées scènes. Par analogie avec les chapitres d'un livre, les scènes contiennent une partie de l'histoire et représentent elles-mêmes un événement à part entière.

Dans cette thèse, nous présentons d'abord une architecture de système distribué qui accueille chaque module de connaissance du contenu. Cela se fait par la création de composants dans une architecture plus large qui exploite efficacement les caractéristiques et les possibilités d'un grand nombre de périphériques intelligents (*smart devices*) (tels que les organiseurs).
Dans une seconde partie, nous donnons un aperçu de l'état de l'art que nous utilisons comme base de notre travail.
Plusieurs modules de bas et moyen niveau permettant de créer une connaissance du contenu sont présentés dans une troisième partie, notamment un détecteur de changements de plans très robuste.
Comme la segmentation en entités sémantiques est porteuse d'information ne repose pas uniquement sur des décisions objectives, nous présentons la technique de production de films et les règles de grammaire habituellement appliquées. Grâce à la connaissance de ces règles, nous décrivons un détecteur permettant d'identifier des

séquences narratives liées, telles que le cross-cutting et les dialogues. Ce détecteur est appelé détecteur de plans parallèles; il permet de regrouper jusqu'à 70% des plans de contenu narratif en plans parallèles.
Enfin, nous présentons diverses méthodes de détection de changement de scènes pour classifier les plans qui n'ont pas été fusionnés par la méthode précédente. Les résultats obtenus sont encourageants (prometteurs) et atteignent des pourcentages de détection, c'est-à-dire des résultats de rappel et précision, de l'ordre de 80% pour les séries et de 66% pour les films.

Discipline: Informatique

Mots-clés: indexation multimeda, segmentation temporelle des flux multimedia, classification multimedia, analyse des contenus audiovisuals numérique, analyse sémantique des scènes

LaBRI
Université Bordeaux 1,
351, cours de la libération
33405 Talence Cedex (France)

Philips Research
High Tech Campus 36
5656 Eindhoven (The Netherlands)

Abstract:

Technical attributes of new consumer storage devices, but also broadband connectivity to Internet video portals, increased consumers' desire to be able to manage intuitively and hassle free large private or commercial video archives. Raw video data of those archives do not provide the appropriate semantic access data (metadata). Hence, the market requires context-awareness creating metadata describing the individual content items not only on feature level, but preferably also at a semantic level. Consumers, for example, desire to have fast access to the individual sub-events embedded in a content item. This thesis presents a method to segment the audiovisual content item into meaningful chapters, also called scenes. The latter have an analogy to chapters of a printed book. They both contain a part of the story, but in itself they contain one story event.

In this work we present first a distributed system architecture to host individual modular content-awareness creating components in a larger framework exploiting efficiently the attributes and capabilities of a larger set of smart devices.

Hereafter, an overview of the state-of-the-art is given, which is used as basis for our own work.

Subsequently several low- and mid-level content-awareness-creating modules are presented, such as a very robust shot boundary detector.

Knowing that the segmentation into meaningful semantic entities not only relies on objective decisions, the art of film production is introduced and commonly applied film grammar rules are presented. Exploiting the knowledge of film grammar we present a detector identifying interleaved narrative sequences, such as cross-cuttings and dialogues, further called parallel shot detector. The latter clusters up to 70% of all shots of narrative content into such parallel shots.

Finally we present in this work various methods to identify semantic audiovisual scene boundaries within the remaining parts, i.e. after parallel-shot-based clustering. The results achieved for automatic semantic segmentation are encouraging reaching detection rates, i.e. recall and precision, of 80% for series and 66% for movies.

Discipline: Computer Science

Keywords: multimedia indexing, temporal segmentation of multimedia streams, multimedia classification, digital audiovisual content analysis, semantic scene analysis

LaBRI
Université Bordeaux 1,
351, cours de la libération
33405 Talence Cedex (France)

Philips Research
High Tech Campus 36
5656 Eindhoven (The Netherlands)

Résumé prolongé:

L'évolution constante des caractéristiques techniques des appareils de stockage domestique tels que les magnétoscopes ou bien caméscopes permettent aujourd'hui aux utilisateurs de générer une grande quantité de contenus multimédia. En conséquence de quoi les consommateurs d'aujourd'hui veulent avoir la possibilité de gérer leurs archives vidéo autant que possible intuitivement et sans grand effort. Les données vidéo brutes des contenus multimédia ne contiennent pas suffisamment d'informations sémantiques nécessaires à leur gestion de manière intuitive, de la même manière qu'un DVD propose un accès interactif à chaque chapitre d'un film. A l'aide de ces accès le consommateur peut, ainsi, facilement et intuitivement naviguer à travers le film. Pour offrir la même possibilité de gestion des contenus privés la génération de métadonnées descriptives au niveau sémantique est nécessaire.

Cette thèse présente un système et une méthode de segmentation des données audiovisuelles en chapitres sémantiques, appelées scènes. Ces scènes sont assimilables aux chapitres d'un livre. Les scènes représentent une partie de vidéo homogène au sens de la sémantique.

Dans ce travail nous décrivons d'abord les tendances techniques et commerciales, qui influencent le marché des hautestechnologies destiné au grand public. Parmi ces innovations technologiques nous avons : l'expansion des capacités de calcul, la croissance des capacités d'enregistrement ou bien encore l'élargissement des bandes de transmission. La connaissance approfondie de ces évolutions techniques ainsi que la demande du marché sontà la base de la création d'une plate-forme d'analyse et d'indexation des contenus multimédia.

Dans ce travail nous avons opté pour l'utilisation d'un *Service Oriented Architectur* SOA, à savoir une architecture distribuée d'analyse du contenu composé de plusieurs modules appelés *System Units* SU. Cette architecture contient de nombreux composants individuels nécessaires à la connaissance du contenu i.e. les algorithmes d'analyse du contenu pourront être utilisés de manière indépendante et spécifiques (traitement audio et vidéo) et exploiter, ainsi, pleinement les capacités du système.

L'architecture que nous avons développée peut être assimilée à un ensemble d'appareils domestiques. Chaque composant analysant le contenu contient un algorithme spécifique pour une modalité, par exemple pour l'audio. Chaque composant, contenant un algorithme est intégré dans une unité *Service Unit*, qui peut communiquer avec le système à l'aide d'interfaces d'entrée/sortie standardisées. Le système considère les *Services Units* comme une boite noire, c'est-à-dire que le système ne connaît pas les processus internes de l'unité *Service Unit*, mais il peut l'utiliser avec l'aide des interfaces. De plus, le système contient une composante additionnelle qui permet d'utiliser efficacement les capacités de calcul et d'enregistrement. Cette composante est appelée *Connection Manager*. Une autre composante, le *Health Monitor*, est lui responsable de la robustesse du système. Avec cette composante l'état individuel du *Service Unit* est surveillé et réparé en cas de nécessité.

Dans le premier chapitre, nous présentons un ensemble des techniques de segmentation sémantique puis nous présentons une étude des travaux actuellement en cours. Cette dernière inclus les méthodes que nous avons développées dans cette thèse.

Ensuite, nous exposons les descripteurs de bas et moyen niveau développés dans ce travail. De plus, dans le but de proposer un système robuste de détection des *Commercial Block* (publicités), nous avons développé de nombreux descripteurs vidéo de bas et moyen niveau tels que: *Monochrome Frame Detector*, *Progressive-Interlaced Classifier*, un *Letterbox Detector* et un *Shot Boundary Detector*. Explicitement pour le *Shot Boundary Detector* nous avons analysé d'abord les profiles des différents groupes de consommateurs afin de proposer un corpus de test adapté composé d'un ensemble d'extraits vidéo de genre divers et provenant de plusieurs pays. Après quoi, nous avons analysé et développé trois types de *Shot Boundary Detectors*, à savoir les *Marcoblock Correlation Cut Detector*, le *Field Difference Cut Detector* et finalement le *Colour Segmentation Cut Detector*. Ces trois détecteurs ainsi qu'un détecteur issu de la littérature ont été testé avec notre corpus afin d'en évaluer les performances. Les tests ont montré, que le *Field Difference Cut Detector* donnait de meilleurs résultats. En utilisant des algorithmes d'analyse supplémentaires nous avons pu améliorer ces résultats, pour cela, nous avons appliqué les technologies *Feature Points* et les analyses rétrospectives. Les résultats, en termes de rappel et précision, sont de 98,3 % et sont supérieurs aux détecteurs publiés dans la littérature. En conséquence, nous avons utilisé les détecteurs que nous avons développés dans cette thèse. Une amélioration finale consiste en l'intégration d'un détecteur décrit dans la littérature, le

Gradual Transition Detector, en y ajoutant quelques étapes d'analyse supplémentaires nous avons obtenu une meilleure précision dans les résultats.
Dans l'étape suivante, nous avons examiné plusieurs paramètres audio de bas et moyen. Notre but est de combiner les paramètres audio et vidéo pour créer un détecteur de publicité plus robuste que ceux présentés dans la littérature. Nous avons construit un détecteur de silences détecteur qui utilise des données d'entrée audio compressées au format MPEG-1 layer 2 et AC3. Nous avons étudié les performances propres de chacun des divers descripteurs audio et vidéo développés au cours de notre travail. Avec les résultats que nous avons obtenus nous avons décidé de combiner notre *Monochrome Frame Detector* avec notre détecteur de silence de publicité. Utilisant notre corpus pour nos tests, nous avons obtenu une précision de 99,93 % et un rappel de 91,4 %. Nous avons implémenté ce détecteur comme un *Service Unit* et nous l'avons intégré dans notre système global, nous avons ainsi éliminé un des segments de vidéo tels que les passages contenant des réclames publicitaires qui étaient sources de fausses alarmes.

Il est nécessaire de prendre en considération que la segmentation en unités sémantiques doit respecter non seulement les règles objectives mais aussi les règles appliquées dans la production de films (post-production, montage, collage, …). La connaissance de ces règles nous a permis d'analyser des méthodes différentes qui permettent de classifier des séquences complexes et ensuite de les classifier en séquences cohérentes. On appelle ces séquences les *Parallel Shots*, chaque unité représente une partie singulière sémantique. Elles ont la caractéristique de ne contenir, par définition, aucun changement de scène. Ainsi, une grande partie du contenu du film peut donc être classifié dans différents sous-genres de *Parallel Shots*. Cette classification précède la classification des scènes sémantiques.

Les *Parallel Shots* peuvent être répartis en deux groupes : les *Cross-Cuttings* et les *Shot-Reverse-Shots*. Les plans de montage vidéo sont utilisés comme unité de base de la segmentation. En général environ 70 % de tous les plans de montage vidéo des contenus audiovisuels tels que les films ou bien les séries, peuvent être classifiées en *Parallel Shots*. Pour la pré-segmentation nous avons étudié et développé plusieurs techniques d'analyse de similarité des images de la vidéo. Parmi ces méthodes, nous avons les méthodes a) HSV ressemblance b) HY ressemblance c) ScaFT ressemblance et d) SIFT ressemblance. Une comparaison des résultats de classification a montré que les méthodes les plus simples – HSV et HY – donnent des résultats meilleurs, comparé avec les méthodes de calcul plus complexes que sont les méthodes ScaFT et SIFT. Les

résultats de la méthode HSV et de la méthode HY étaient plus ou moins identiques, nous avons donc décidé d'utiliser la méthode HSV dans la suite de ce travail de thèse, cette méthode se caractérise aussi par une implantation plus simple. Les résultats de pré-segmentation sont de 75 % pour le rappel et presque 90 % pour la précision.

Après avoir éliminé tous les segments de vidéo qui ne correspondent pas à notre contenu grâce à notre détecteur de publicité et ayant classifie presque 70 % de tous les *Shots* en *Parralel Shots* nous avons essayé de chercher dans cette partie restant du film les changements de scène sémantique. Tout d'abord nous avons construit un détecteur de non ré-assemblage basé sur la méthode HSV. Avec ce détecteur nous avons obtenu des résultats de détection des changements sémantiques de scène avec une qualité de 85 % pour le rappel et de l'ordre de 50 % pour la précision. Pour améliorer ces performances nous avons construit des détecteurs supplémentaires pour le flux audio et de durée du *Shot*. La combinaison HSV et durée du *Shot* a donné des résultats très acceptables. Ce détecteur combiné de classification de changement de scène a donné des résultats à hauteur de 80 % pour les séries et 66 % pour les films - rappel et précision sont identiques en termes de valeurs. Prenant en considération les problèmes subjectifs et donc la difficulté de cette classification avec un détecteur de changement de scène ont peut accepter ces résultats comme représentatifs et aptes à l'intégration dans des appareils domestiques.

CONTENTS

TABLE OF FIGURES

CHAPTER 1

1 INTRODUCTION

1.1 Introduction (in English)

Our, the consumer's, live can be divided in general in our work live and our private leisure live, during which we aim to enjoy and to relax as much as possible. In interaction terms we talk either about lean-forward, i.e. work-based, or lean-backward, i.e. entertainment-based, activities. *Consumer Electronics* CE manufacturers are mainly aiming to provide solutions for the entertainment related part of our lives and, hence, strive to invent lean-backward oriented applications and services. Various technology breakthroughs in the domain of processing, memory and connectivity eroded the price levels for these particular technologies in the last decade changing them into commodity solutions. As expected, shortly after new processing- and memory-powerful consumer electronic devices conquered the market, such as *Digital Versatile Disc* DVD recorders and *Hard Disk* HDD recorders. Moreover, more and more consumer electronics devices embed connectivity units resulting in In-Home networks i.e. interconnected CE devices, and CE devices with broadband connection to the Internet.

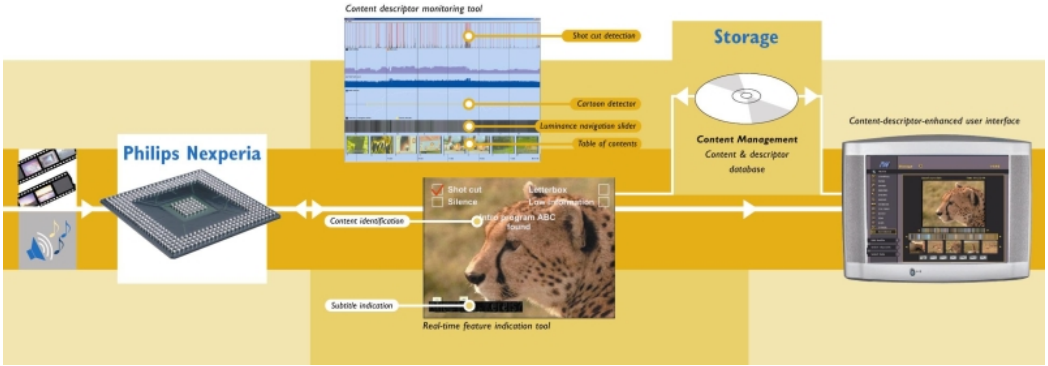


Figure 1. Content analysis for CE devices at Philips Research[1].

[1] Personal image of the author.

Nevertheless, more storage space on a consumer storage device, such as on an HDD recorder, not necessarily leads automatically to a pure advantage. If enabling consumers, i.e. all of us, to store and archive whatever we like onto the hard disk of our HDD recorder, naturally this leads to a 'searching for the needle in the haystack' retrieval problem. More seems to be less in this case. This problem in combination with the available processing power gave the content analysis community another outlet for their work. Many consumer electronics manufacturers realized this as a threat, but at the same time as an opportunity and started with own research projects, as e.g. the project Cassandra at Philips Research, with its abstract semantic visualized in Figure 1. One of the project's aims is to enrich unstructured recorded content stored e.g. on an HDD recorder with content awareness, i.e. content descriptors, by applying content analysis. These content descriptors, also called metadata, should be created by dedicated content analysis software components or integrated circuits. These descriptors should be stored with the content in a standardized way onto the storage medium, e.g. DVD, as claimed by us in [1] and [2]. But why are these descriptors of such relevance one may ask?

Today's consumers are used to browse and navigate through purchased commercial DVDs in a non-linear fashion using embedded chapter information. These chapter markers have been provided by the content providers and these markers segment the content into semantic meaningful entities. No need to say that consumers would like to have a comparable semantic chaptering solution as well for their recorded broadcast content. Hence, one of the expressed consumer desires is to browse and navigate intuitively and in a non-linear way through individual recorded content items and archives. With this market value proposition we decided to research new automatic audiovisual content segmentation solutions applicable for consumer devices, such as HDD recorder, but also Internet services using AV archives.

Hence, the main aim of this work was to research an automatic and semantic chaptering solution, i.e. converting recorded content-unaware broadcast of private content into content-aware audiovisual content augmented by semantic segmentation. The solution had to be suited for the implementation into e.g. DVD or HDD based CE storage devices. One of the first challenges was identifying the technology and consumer trends to secure providing the market with the appropriate technology and consumer solution at the appropriate time. In the second chapter of this work we present these underlying fundamental consumer and technology trends, which served as basis when choosing appropriate technologies throughout our work.

Because of the semantic nature of our task we faced soon the issue that we needed syndicating many single modality content analysis solutions into one overall system. For efficiency, complexity and transparency reasons each single modality analysis solution had to be embedded into one component. The trend towards distributed system architectures and modularity based solutions motivated us to select a *Service-Oriented Architecture* SOA as preferred choice. In the second half of chapter two we describe our approach towards an SOA based distributed content analysis prototyping framework. Each analysis solution was embedded into a container component called *Service Unit*, each communicating with the system through standardized interfaces. Nevertheless, we faced some robustness and maintenance issues, which we researched in more detail. As a result of this, we elaborated dedicated components such as a *Health Monitor* and a *Connection Manager*, which helped us to solve some of the identified issues. Finally, in the end of chapter two we introduced a set of selected individual content analysis *Service Units*, which syndicated together, formed our envisioned semantic content segmentation application.

In order to avoid redundant work we studied thoroughly a selected group of state-of-the-art technologies, which we expected to become relevant for our semantic segmentation application. For each technology block we collected the available information and evaluated the state-of-the-art solutions on their suitability for our application, on their maturity and on their robustness. In chapter three we described the studied state-of-the-art technologies and summarized our analysis results. Special attention was given in this analysis to segmentation-related works such as shot and scene segmentation. The results of this state-of-the-art study served as basis for the decision, which technologies required further research and development to achieve our aim of a semantic segmentation solution.

In our analysis we unveiled that broadcast content items contain non-content related inserts, i.e. commercials, which could deteriorate the results of any automatic chaptering solution. In the state-of-the-art analysis we retrieved several interesting solutions for commercial block detection, but none of them satisfied our requirements. Hence, we researched, starting at the bottom of the content analysis pyramid as sketched in Figure 2, several compressed domain video low-level and mid-level features specially designed for a dedicated video compression hardware unit, which was at our availability. These features ranged from *Monochrome Frame Detector*, *Progressive-Interlaced Detector*, *Letterbox Detector* to *Shot Boundary Detector*, as described in the first part of chapter four. Especially for the latter we faced the challenge that none of the benchmark corpora

available from academia or industry served our purpose, i.e. consisting of a variety of genres and of a variety of cultures. To establish a representative corpus we researched the consumer behaviour of our target consumer group and established accordingly our benchmark corpus, which we applied to test and benchmark our solutions, such as the *Shot Boundary Detector*. Especially the latter was of importance, because shots are an important atomic entity in video content and the robustness of the latter is of utmost importance for all subsequent analysis steps. Because the state-of-the-art solutions did not achieved the detection results required we researched three *Shot Boundary Detectors*, i.e. *Marcoblock Correlation Cut Detector*, *Field Difference Cut Detector* and *Colour Segmentation Cut Detector*, and benchmarked them using our own corpus against each other and against one detector derived from academia. The latter we used as objective reference because it participated in the TRECVid benchmark. The winner of our benchmark, i.e. *Field Difference Cut Detector*, was selected and we further researched improvements to even further boost the robustness by e.g. applying processing steps using *Feature Points* and *Backward Analysis*. Furthermore, we implemented an inherited, from academia, Gradual Transition Detector and improved it's robustness with post-processing steps, as described in the first part of chapter four.

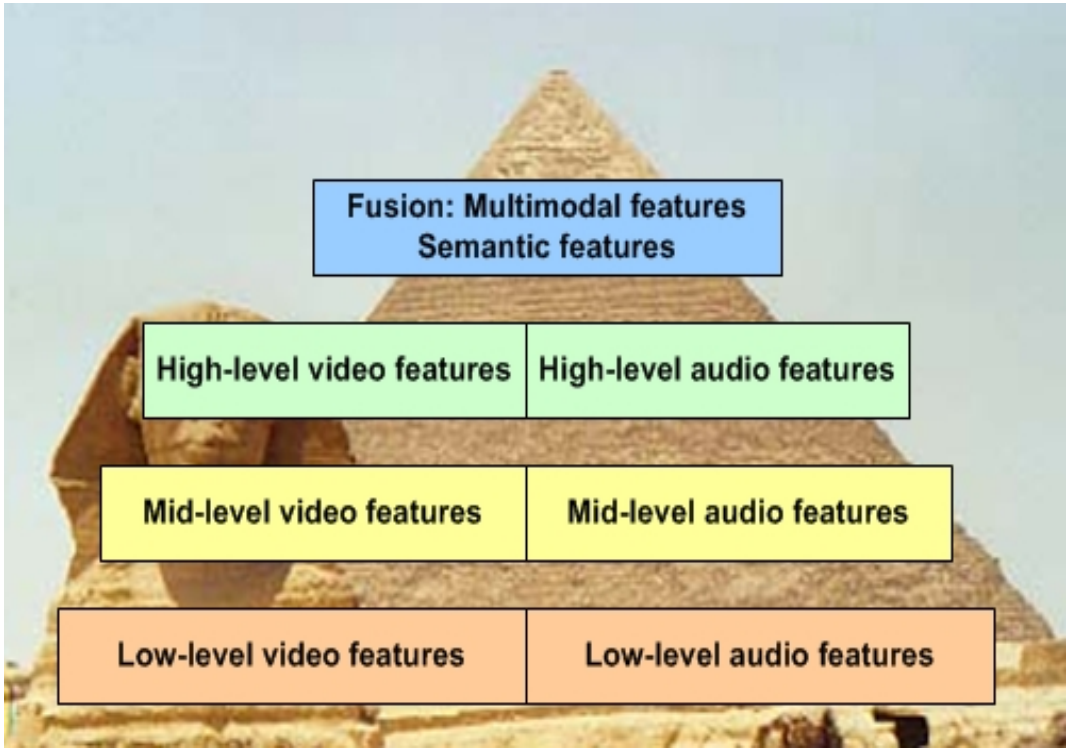


Figure 2. Content analysis feature pyramid – layers of semantics[1].

Due to the multi-modal nature of our audiovisual content we researched, subsequently, strong audio low-level and mid-level features, such as a dedicated compressed domain commercial silence detector. During the state-of-the-art analysis we retrieved only general purpose silence detection solutions, which did not fulfil our requirements.
The results of our compressed domain commercial silence detector were summarized in the second section of chapter four.

The latter we then combined with a variety of selected video low-level and mid-level features to derive a dedicated genre detector, i.e. a commercial block detector. Our research unveiled that the combination of our *commercial silence detector* and *Monochrome Frame Detector* performed best and outperformed robustness-wise those, which we analysed during the state-of-the-art analysis. We summarized our research on this detector in the fourth section of chapter four. Finally, we implemented this *Commercial Block Detector* as *Service Unit* into our analysis framework, which here after automatically eliminated all non-content related inserts, i.e. commercial blocks inserted by the broadcasters.
As early spin-out of this work we implemented a commercial skip application. We did this due to the strong customer and market request we witnessed during our research work.

Being aware of the subjectivity of our semantic content segmentation task we decided to research the art of film production, trying to extract common objective film grammar production rules, which we could apply to cluster and further segment our audiovisual content. The analysis of the state-of-the-art of chapter three unveiled some clustering techniques and our aim was to understand the underlying production rules enabling us to extract objective common rules. The latter we intended to apply to build an appropriate robust clustering solution. Our study showed that in narrative content clusters of interleaved narrative sequences, so-called *Parallel Shots*, are commonly applied. The latter form semantic sub-entities and they can be divided into two classes, i.e. *Cross-Cuttings* or *Shot-Reverse-Shots*. By definition these *Parallel Shots* do not contain any scene boundaries. We researched several techniques for *Parallel Shot Detection*, which allowed us to pre-cluster a substantial amount of shots into such narrative semantic sub-entities. We described this work in more detail in the fifth section of chapter four.

After clustering a substantial part of the narrative content into sub-entities, which shared audiovisual commonalities following certain film grammar rules, we researched methods to identify semantic meaningful discontinuities, i.e. *Scene Boundaries*, in the remaining parts of the content. We described this work in the fifth section of chapter four. During this research we identified several shortcomings of colour discontinuity methods, which we retrieved during the state-of-the-art analysis. In subsequent steps we eliminated some of these shortcomings by applying our film grammar knowledge. The robustness achieved with the resulting colour-based boundary detector was reasonable. Nevertheless, we researched independent boundary detection methods, such as audio-based class transition detection and shot-length based boundary detection. The combination of our colour-based and shot-length-based boundary detection methods resulted in a robust Scene Boundary Detector fulfilling finally our requirements. Hence, in this work we achieved the goal to group interleaved narrative sequences and to identify strong audiovisual discontinuities in the remaining parts of the narrative content. A subsequent step towards real semantic characterization of audiovisual content, not covered in this work, using our segmentation method enabling users to search intuitively at a semantic level would be to attach cognitive descriptions, i.e. semantic tags, to individual scenes and sub-elements.

In the last chapter, i.e. chapter five, we concluded our work, summarized our conclusions and gave some perspectives.

1.2 Introduction (en Français)

Nos activités quotidiennes se décomposent en tâches professionnelles et en tâches privées. Nous attribuerons les termes techniques suivants pour qualifier ces deux tâches: *lean-forward*, tâches professionnelles et *lean-backward*, désigne l'activité dédiée aux loisirs. Les producteurs d'équipements domestiques essayent d'offrir des solutions à l'écoute du grand public. En effet, ils s'orientent plus ou moins vers le secteur *lean-backward*. Avec les progrès rapides liés aux nouvelles technologies en termes de performances de calcul, de capacité de stockage, de largeur de bande de transmission les prix des technologies ont largement baissé permettant ainsi de s'ouvrir à un public plus large et que ces produits ont rapidement conquis le marché grand public, par exemple en ce qui concerne les enregistreurs DVD et HDD. De plus en plus la plupart de ces équipements peuvent être interconnectés, créant ainsi un réseau domestique relié à l'Internet, offrant par la même occasion un accès important aux contenus audiovisuels.

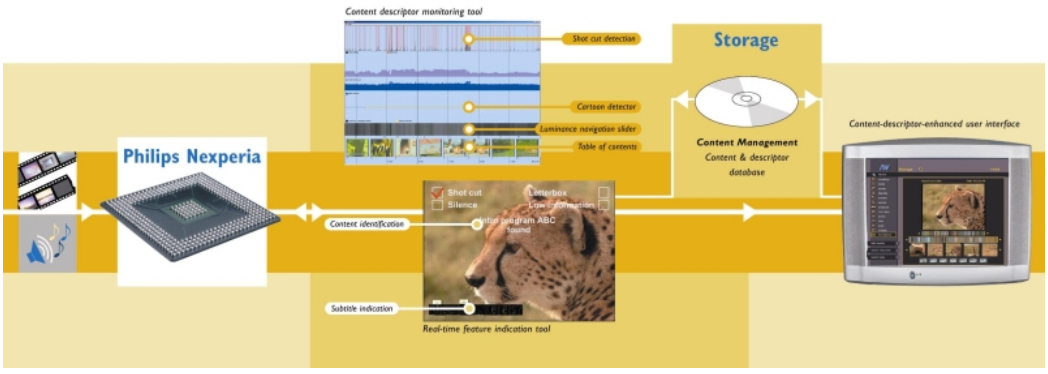


Figure 3. Analyse du contenu pour équipement domestique chez Philips Research[1].

Cependant, cette avancée n'offre pas que des avantages, mais, au contraire peut génère de nouveaux problèmes. En effet, l'accès facilité à une plus grande quantité de données audiovisuelles implique la nécessité de pouvoir organiser ces dernières Un domaine d'activité avantageux et prospectif s'ouvrait alors à la communauté des chercheurs de l'industrie et des sciences. En effet, ils doivent se pencher sur des solutions de gestion des grandes quantités de données. Les grandes compagnies ont très rapidement localisé le problème, et elles ont rapidement trouvé des solutions, comme par exemple avec le projet de recherche *Cassandra* développé par *Philips Research*. Le diagramme abstrait du projet *Cassandra* est présenté en Figure 3. Le but du projet consiste en l'analyse des vidéos enregistrées, par exemple par un enregistreur HDD, permettant ainsi la génération de métadonnées liées au contenu. Les descripteurs du contenu, aussi appelées métadonnées sont crées par un logiciel analysant le flux audio ou vidéo qui est peut aussi être traité par un analyseur matériel (hardware). Les descripteurs sont enregistrés en temps-réel en même que le contenu lui-même voire [1] et [2]. Les descripteurs ainsi obtenus permettent, entre autres, à l'utilisateur de naviguer a travers le contenu en modus *lean-backward* et de rechercher les passages désirés.

Le consommateur veut avoir la possibilité de naviguer à travers le contenu intuitivement et sans barrières linéaires. Ils ont l'habitude de naviguer à travers les DVD commerciaux selon leur volonté avec l'aide des index des chapitres existants. Les index des chapitres sont mis à la disposition au consommateur par le propriétaire de l'information, par exemple la société du film. Le contenu est segmenté en unités sémantiques cohérentes, les chapitres. Les utilisateurs attendent naturellement une solution adéquate pour classifier le contenu qu'ils ont enregistré. Cette ouverture du marché évidente nous a persuadés de nous engager dans la recherche de solutions de segmentation automatiques pour les médiums audiovisuels, appliqués aux équipements domestiques, mais aussi sous formes de services Internet afin de traiter des archives audiovisuelles.

L'objectif de ce travail est de trouver une solution de segmentation automatique et sémantique avec laquelle il sera possible de transformer un contenu neutre télédiffusé, ou vidéo privées, en matériel audiovisuel enrichi d'index sémantiques autrement dit en chapitres. Cette solution devrait être de telle sorte que l'utilisateur doit pouvoir enregistrer toutes les informations présentes dans son équipement domestique. D'abord, nous avons analysé en détail les développements techniques ainsi que le marché grand public pour avoir la possibilité d'offrir les meilleures solutions technologiques aux consommateurs. Dans le chapitre suivant nous présentons les

études des méthodes et du marché qui nous ont aidé à choisir la technologie la mieux adaptée.

Considérant la qualité sémantique de notre sujet de recherche nous avons très vite réalisé qu'il fallait incorporer plusieurs modules indépendants d'analyse du contenu dans un système commun et interactif. Chaque unité du module indépendante à sa propre tâche dédiée, transparence et complexité, il est donc nécessaire de les traiter comme unité indépendante dans un système commun. Cette nécessité de respecter l'architecture du système et la solution basée sur des modalités orientés nous ont amené à choisir pour notre travail une solution de *Service Oriented Architecture* SOA. Dans la deuxième partie du chapitre 2, nous donnons les détails sur notre approche basée sur cette solution: une approche menant a un prototype d'un système analysant le contenu. Chaque composante, module indépendant, a été incorporé dans un conteneur appelé *Service Unit* SU. Nous avons rapidement réalisé que nous allons avoir des problèmes de robustesse et de maintenance. Cherchant à maitriser ces problèmes nous avons développé des solutions appropriées – le *Health Manager* et le *Connection Manager*. Une solution définitive de la méthode correcte d'analyse sémantique est décrite à la fin du second chapitre, il s'agit d'une solution contenant plusieurs *Service Units*.

Dans le but d'éviter les redondances nous avons examiné en détail un ensemble de techniques que nous avons préalablement sélectionné dans la littérature pour notre application de segmentation afin de les rangées sous forme de famille. Dans le cas de groupes ne contenant qu'un seul élément, nous les avons regroupé en tenant comptes de toutes les informations et nous avons évalué les différentes techniques selon leur qualification pour notre application : robustesse et niveau de développement. Dans le troisième chapitre, nous avons étudié l'état de l'art pour chaque technologie, puis nous faisons un résumé des résultats de l'analyse. Ainsi, nous avons pu formuler un ensemble de propositions de solutions pour la segmentation d'un contenu multimédia audio et vidéo. Les résultats de cette analyse sont décisifs pour les activités futures, surtout pour décider quelles technologies devraient être examinées plus en détail pour aboutir à une solution satisfaisante pour la segmentation sémantique.

Dans notre analyse, nous avons découvert que les émissions de télévision contiennent une grande quantité de passages secondaires, comme par exemple les réclames, qui peuvent influencer négativement sur les solutions de segmentation. Nous avons trouvé une variété de solutions modernes de détecteurs de réclames, mais malheureusement

aucune n'est satisfaisante selon nos exigences. Notre analyse prenant départ au bout de la pyramide, – voir Figure 4, la première étape consiste en l'examen des descripteurs de bas et moyen niveau pour le flux vidéo compressé. Les descripteurs sont rangés en commençant par le *Monochrome Frame Detector*, le *Progressive-Interlaced Detector*, ensuite le *Letterbox Detector*, allant jusqu'au *Shot Boundary Detector*. Tous ces détecteurs sont décrits en détail dans la première section du chapitre 4.

Concernant le *Shot Boundary Detector* nous avons constaté que les corpus utilisés dans les différentes applications industrielles et dans les applications scientifiques n'étaient pas satisfaisant puisque aucun d'entre eux ne contenaient pas plusieurs genres de vidéos, ou bien ne provenaient i pas de plusieurs stations d'émission et/ou différents pays.

Afin d'obtenir un ensemble de test audiovisuel le plus représentatif de la population actuelle, nous avons tout d'abord analysé l'habitude des consommateurs par tranche d'âge, ainsi, avec les informations obtenues nous sommes en mesure de proposer une base de données de test qui est satisfaisante pour notre *Shot Boundry Detector*.

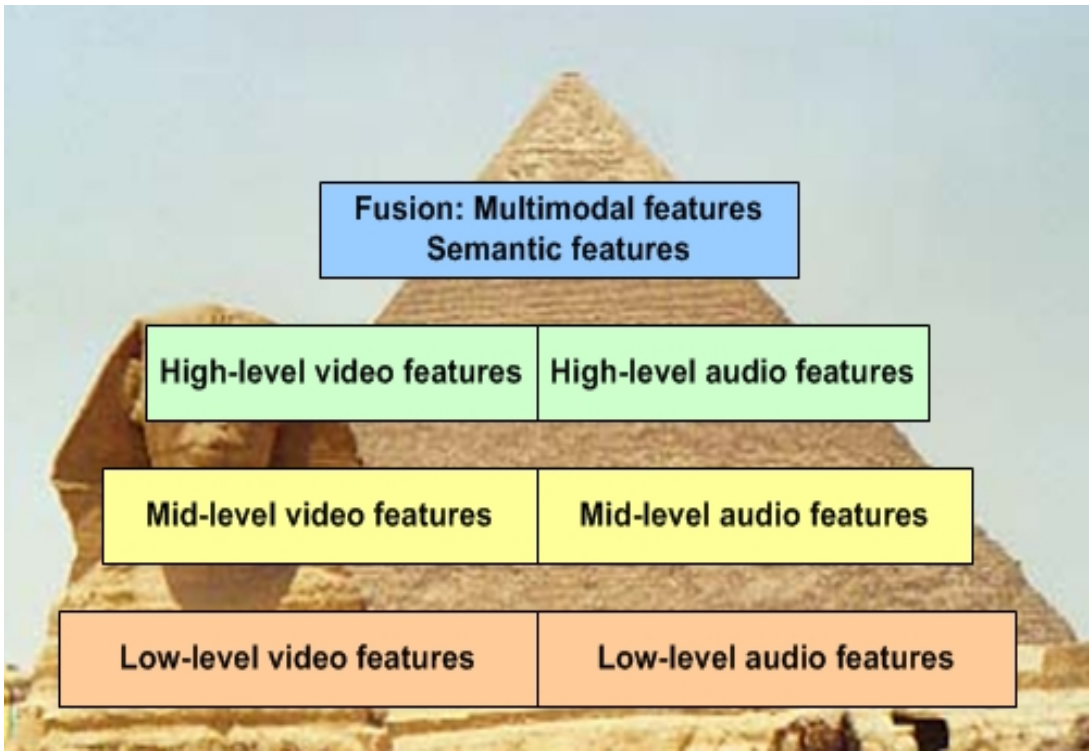


Figure 4. Classement des attributs selon l'analyse du contenu – hiérarchie sémantique[1].

Ce détecteur est particulièrement important, puisque les chapitres d'un film et les *Shots* dont ces derniers dépendent sont une partie élémentaire d'un film et sont la base de toutes les étapes suivantes de l'analyse.
Les solutions présentées ailleurs ne donnent pas une précision suffisante, nous avons donc été obligés de proposer des améliorations. Nous avons examiné trois solutions : le *Marcoblock Correlation Cut Detector*, le *Field Difference Cut Detector*, et le *Colour Segmentation Cut Detector*. Nous avons analysé ces trois détecteurs à l'aide de notre base de test vidéo. Ensuite, nous avons comparé les résultats avec un détecteur appliqué en industrie, un détecteur qui a été utilisé lors de la campagne d'évaluation *TRECVid*. Selon notre étude, le *Field Difference Cut Detector* donne de meilleurs résultats. Nous avons donc sélectionné ce détecteur pour la suite de notre travail et nous avons essayé de faire des corrections additionnelles pour améliorer sa robustesse, en y ajoutant par exemple le 'Feature Points' et l'analyse rétrospective. Ensuite, nous avons développé un moderne *Gradual Transition Detector*, avec des corrections de notre part en vue de d'améliorer sa robustesse – voir la première section du chapitre 4.

Nos sources audiovisuelles sont multimodales, ce qui nous amène à examiner des détecteurs de bas et moyen niveau ad-hoc, par exemple le détecteur de silence dans le domaine compressé. Le détecteur commun appliqué normalement avait des performances acceptables pour notre travail. Nous donnons les résultats de notre détecteur de silence nous permettant ainsi de localiser les publicités dans nos contenus dans le domaine compressé, dans la seconde section du chapitre 4.
Ce détecteur de silence pour les publicités est combiné à un ensemble de descripteurs vidéo de bas et moyen niveau constituant ainsi notre détecteur spécial, nous l'appelons le détecteur de publicités. La combinaison de notre détecteur de silence de publicité avec notre *Monochrome Frame Detector* donne de meilleurs résultats et dépasse largement la robustesse des détecteurs communs. Dans la quatrième section du chapitre 4 un aperçu des étapes de recherche est donné. Nous avons finalement intégré ce détecteur de publicité comme Service Unit dans notre système global. Cette unité a permis d'éliminer automatiquement toutes les séquences qui ne sont pas propre au contenu lui-même, comme par exemple les publicités intercalées au cours d'un. L'élimination des réclames est une réussite immédiate de notre travail, puisque cette application est vivement recherchée par le marché du grand public.

Puisque le sujet principal de notre recherche – le film – apporte une segmentation subjective en scènes sémantique nous avons étudié les règles communes de la

production d'un film, afin de pouvoir plus correctement appliquer les solutions de segmentation. Evidement quelques solutions sont connues, mais notre but est d'examiner les règles objectives et d'acquérir de la connaissance des interdépendances, ce qui devrait améliorer les solutions concernant leur robustesse. Notre recherche montre que dans les contenus narratifs il y a beaucoup de séquences associées les unes avec les autres. On appelle ces séquences communes les *Parallel Shots*. Elles forment des unités de base et des sous-unités, normalement scindées en deux groupes : les *Cross-Cuttings* et les *Shot-Revers-Shots*. Par définition les groupes ne comprennent aucun changement de scène. Nous avons - profitant de cette caractéristique – examiné plusieurs solutions de *Parallel Shot Detection*, ce qui nous a permis de ranger en sous-catégories la majorité de l'information, donc en séquences narratives sémantiques. Nous donnons les détails de cette recherche dans la cinquième section du chapitre 4.

Ayant classifié la majorité des informations en sous-groupes il faut maintenant classifier et analyser le reste de l'information. Pour ce faire nous devons normalement caractériser les changements de scène qu'ils sont susceptibles de contenir. Nous décrivons les recherches en détail dans la cinquième section du chapitre 4. Les méthodes permettant de caractériser les changements de scène basée sur l'étude des couleurs que nous avons trouvées dans la littérature présentent quelques défauts. Nous avons appliqué des corrections provenant de notre étude sur les règles de production d'un film. La robustesse atteinte avec notre détecteur est acceptable. Néanmoins, nous avons apporté des corrections en ajoutant des détecteurs indépendants pour le flux audio et des détecteurs de durée du *Shot*, dans le but d'améliorer la robustesse. La combinaison de notre détecteur couleur et notre détecteur de durée du *Shot* donne un détecteur de changement de scène très robuste.

Dans le dernier chapitre, nous faisons un résumé de notre travail et donnons quelques perspectives pour les recherches à venir.

CHAPTER 2

2 PROBLEM OF AV SEGMENTATION AND ARCHITECTURE APPLIED

2.1 Aim of 'Semantic AV Content Item Segmentation'

The ultimate aim of the work done in this PhD is 'Semantic AV Content Item Segmentation' of commercial content acquired by consumers e.g., through recording analogue or digital broadcast content. This means to automatically segment unstructured content into semantic entities, meaningful to the final user.

2.1.1 Relevant technology and consumer trends

Semantic content segmentation is an example for semantic content-awareness creation, which gained importance due to several consumer and technology trends, observed during the last decade. These trends include:

- the exponential growth of processing power. Moravec's prediction [4] is that individual 1000,- € consumer devices will reach processing-wise, not intellectual-wise, human capabilities by mid of this century, as published in [3] / [4] and shown in Figure 5. Seen realistically this seems to be optimistic, but as IBM's Deep Blue won against Kasparov in chess, other intellectual challenging domains will follow soon.
- the increase of available storage on individual consumer devices outperforming human memory during this century and, hence, potentially augmenting it, as stated in [3] / [4] and shown in Figure 5.
- the strong growth of bandwidth, which finally connects transparently all these individual powerful processing and memory units together into one distributed system architecture, such as a grid or a smart In-home Network.

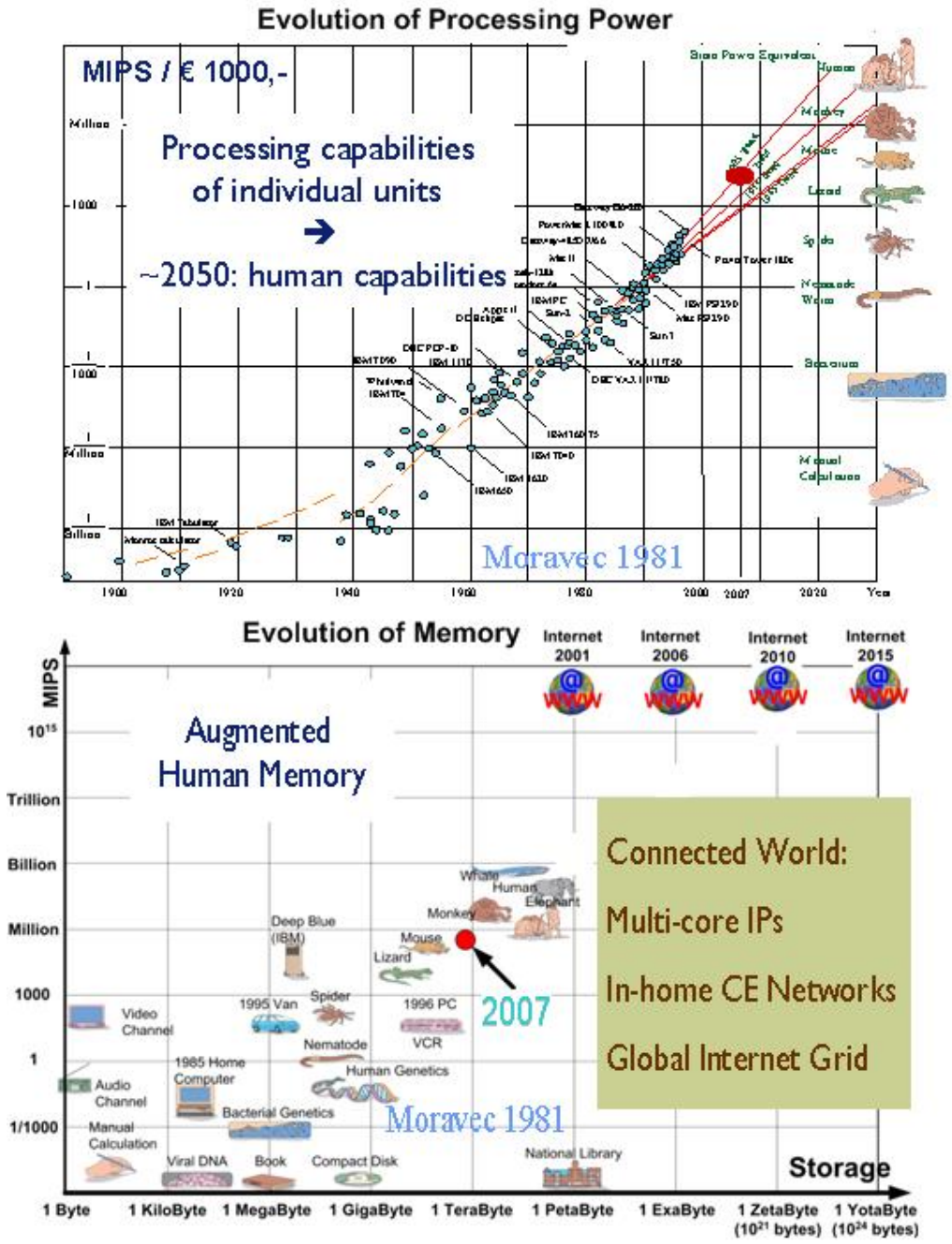


Figure 5. Technology trends: evolution of processing, memory and connectivity[1].

As may be expected, the availability of new technology solutions influences individual consumers' content consumption behavior resulting in several new consumer trends. Some of these consumer trends are:

- the smooth shift from push-model based passive content consumption to pull-based active content selection enabled e.g. by Electronic Program Guides EPG in combination with hard-disk-based Personal Video Recorders PVR.

Figure 6. Technology trend stimulated consumer trends[1].

- that today's consumers replace more and more commercially produced content by consumer self-created content shared e.g. within user communities (Web2.0) or through proprietary Internet services / portals, which maintain huge AV archives.
- that technology-aware consumers desire to consume content appropriately to the context, such as the consumers current mood and location.
- the desire to enhance the experience through replacing e.g. passive rendering devices by active devices, e.g. Philips' AmbiLight TV as shown in Figure 6, or letting the content itself become smart, i.e. pro-active self-aware content items.
- the wish to augment the experience by replacing single-sense stimulating solutions by a plethora of well-conducted sense stimuli. The latter are expected to create experiences perceived as pleasant and harmonious by using e.g. smart environments, i.e. Ambient Intelligence.

These technology and consumer trends have great impact on the content distribution business models, as sketched in Figure 7. The traditional uniform broadcast push model, i.e. viewer-agnostic broadcast channels (Figure 7, left) well suited when frequency-spectra were limited, loses market share to hybrid push-pull-based solutions (Figure 7, centre). Representatives of the latter are *Internet Protocol TV* IPTV based *Video-on-Demand* VOD or *Personal Video Recorders* PVR in combination with *Electronic Program Guides* EPG enabling individual users to 'cherry pick' content of desire. But finally, fully personalized pull-based solutions (Figure 7, right) are going to conquer the market. Examples of the latter are proprietary ('Web2.0' based) Video Internet Services maintaining own AV archives. AV service interface standardization will finally lead to personalized 'Web3.0' portals enabling (for the user) transparent access to diverse AV archives, as shown in Figure 7 (right).

For Consumer Electronics CE this means, when analyzing these trends and following the vision of Ambient Intelligence AmI, that CE solutions should be sensitive, adaptive and responsive to consumer and context, but also anticipatory to consumer's desires. But - in order to achieve this vision - smart, personalized and interconnected systems are required, which are ubiquitous and transparent hiding the underlying system complexity. The underlying system needs to be therefore anticipative, personalized, adaptive, self-organizing, distributed and content- / context-aware. Thus, in the remainder of this chapter, we will investigate first an appropriate architecture for a multimedia content analysis and segmentation system.

2.1.2 Semantic AV content segmentation – content-awareness creation

The before mentioned technology and consumer trends and omni-present business models exploiting *AudioVisual* AV content, e.g. through AV search engines and services, justify the effort to think about new concepts of content-awareness creation.

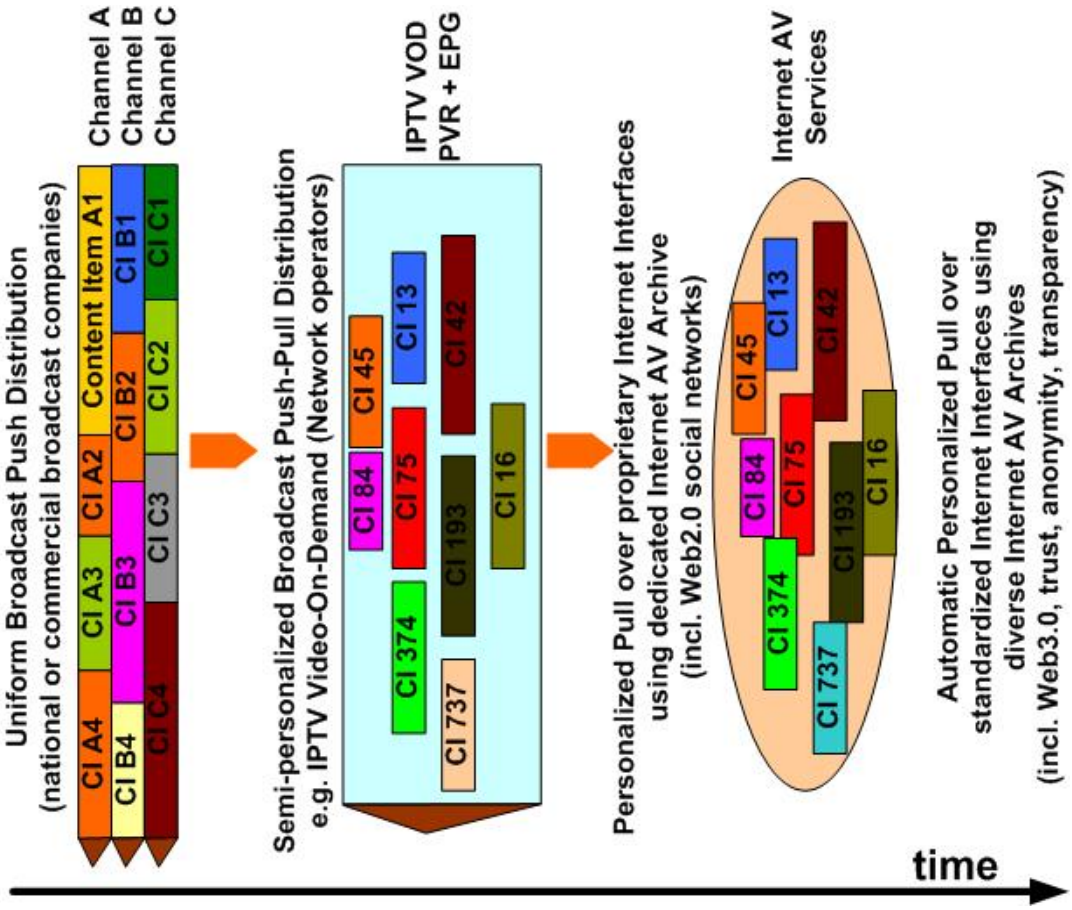


Figure 7. Content distribution trends [5] resulting from consumer and technology trends.

Personalization, intuitive access and experience enhancement require content awareness, especially when aiming for interaction at a semantic human communication like level.

The consumer-oriented application solution, researched in this PhD thesis, is therefore aiming to segment acquired AV content item, e.g. streaming video or a video file, into semantically coherent entities, so called scenes. The contents chosen for this PhD project are recordings captured by consumers by means of their *Consumer Electronics* CE home storage devices such as hard-disc-enhanced *Digital Versatile Disk* DVD-recorders, hard-disc-enhanced *Set Top Boxes* STB, hard-disc-enhanced Media Centre *Personal Computers*, i.e. PCs with hard-disc and AV broadcast recording capability.

Unfortunately, today's AV broadcast of Internet AV content contains little or no metadata i.e., data describing the AV content, which would help to segment it into semantic coherent chapters. This shortcoming applies not only for proprietary solutions, e.g. used by company 'Gemstar' [6], but also for AV services using mostly standardized solutions, e.g. *British Broadcast Corporation* BBC [7]. The reasons for missing metadata are various, but mainly it is related to the difficulties broadcasters and service providers are facing to develop the necessary profitable business models. Multiple standards for metadata are available, such as TV-Anytime [8], *MPEG-7* (Moving Picture Expert Group) [9] and *DVB-SI* (Digital Video Broadcast Service Information) [10], enabling to transmit, e.g., auxiliary information, such as *Electronic Program Guide* EPG [11] metadata, to the regular broadcast signal. For example, MPEG-7 forms a normative framework for multimedia content descriptors, user preferences and usage history. But missing business models are the reason that those data also required for segmentation are provided only incidentally and then they are often inaccurate or inconsistent or in a proprietary format. This led to the conclusion, that the intermediate solution to provide users with 'Semantic AV Content Item Segmentation' is an automatic content segmentation running at the receiver, i.e. consumer, side in CE devices, as sketched in Figure 8.

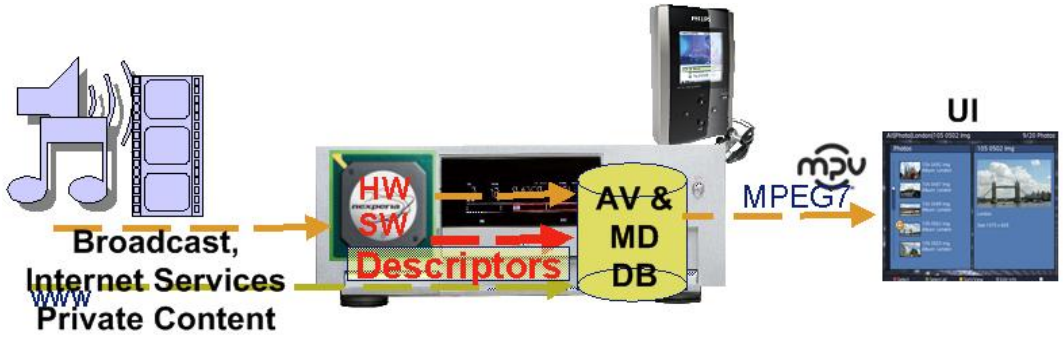


Figure 8. Consumer Electronics devices with content-awareness creation.

This is even more attractive for CE device vendors, because this enables them to promote their devices with differentiating, appealing and, consequentially, turnover increasing applications. The specification of the final application solution, researched and developed in this PhD thesis, has been thoroughly discussed and elaborated with a representative group of potential consumers, CE device marketers and CE device product managers. The final conclusion was that the application should consist of the several building blocks as listed hereafter.

(a) Firstly, low-level audio and video features have to be extracted from the received AV signal, as shown in Figure 9, and elaborated in sections 4.1.1 and 4.2.1.

(b) Secondly, various audio and video mid-level features are required, as described in sections 4.1.2 and 4.2. As example, for semantic AV segmentation it is to chop up the AV content item into its individual video shots, which themselves are semantic meaningful given the underlying production rules as described in section 4.5.1. The required *Shot Boundary Detectors* SBD, i.e. video editing points, are further presented in section 4.1.2.

(c) Thirdly, before starting with the semantic scene boundary detection non-content related inserts, i.e. advertisements further referenced as *Commercial Blocks* CBs, have to be detected and excluded from the subsequent steps. The latter is required because inserts deviate from the semantic flow of the content item itself and, hence, would mislead the algorithms applied for *Scene Boundary Detection* ScBD. The *Commercial Block Detector* CBD is described in section 4.4 and is also applied to enable consumers to fast browse through commercial blocks or to automatically skip them. The latter is a very demanded feature of HDD recorders.

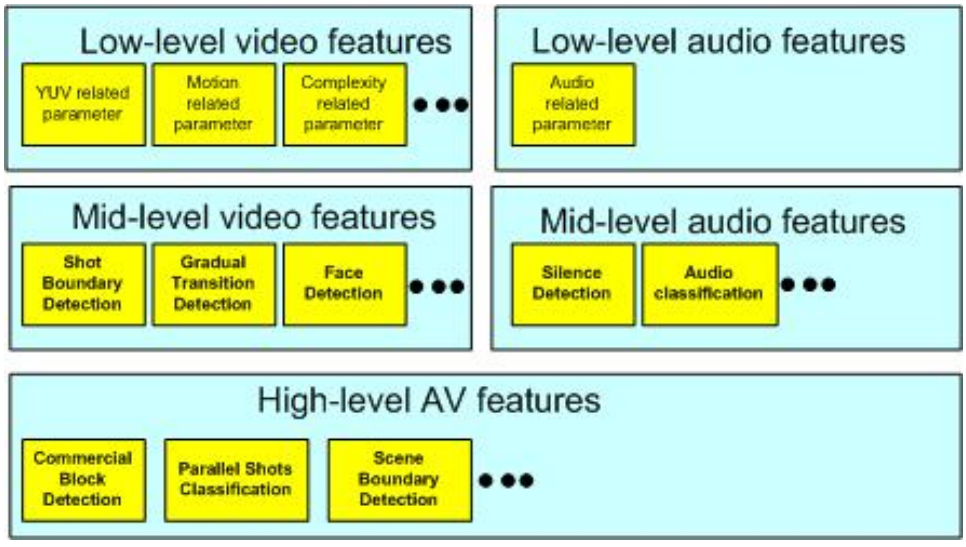


Figure 9. Building blocks applied in the context of this work.

(d) Fourthly, remaining content item shots are clustered into semantic meaningful coherent entities, i.e. film-grammar-based parallel shots as described in section 4.5. An appropriate *Parallel Shot Detector* PSD is researched and developed, which is able to detect and categorize *Shot-Reverse-Shots* SRS and *CrossCuttings* CC, are described in sections 4.5.2 and 4.5.3, respectively. In this way between 50% and 80% of all shots can be clustered together, depending on the genre of the content item, as described in detail in section 4.5.4.

(e) Finally, the discontinuities at all shot boundaries, which are not embedded within a parallel shot sequence, are analyzed, quantified and used as parameter for *Scene Boundary Detection* ScBD. The developed methods of this work are described in section 4.6. The achieved results for automatic chaptering of the AV content into semantic coherent chapters are summarized in section 4.7.

The results of this work have been further applied to elaborate the author's application idea of *Content Item Boundary Detection* CIBD, described in detail in the author's patent applications [12]/[13]. The CIBD clusters coherent chapters, i.e. scenes, together and identifies discontinuities, which are indicative for e.g. the start and end of a movie. Hence applied in consumer devices, this enables consumers to obtain a clean recording of e.g. recorded broadcast content. Furthermore, with the results of this work meaningful key-frame-based summaries, i.e. *Table of Contents* TOC, have been generated, and the shot- and scene-based key-frames have been applied in a *User Interface* UI to implement the author's concept of intuitive editing, as presented in the author's patent application in [14].

2.1.3 Definition of audiovisual scenes

The task of scene boundary detection can be seen as a reverse engineering of the director's or producer's intended story unit concept, which is of high semantic nature as one can imagine and, hence, also subjective. Many authors proposed, because of the task's relevance, definitions for scenes and scene boundaries. Beaver, for example, defines in [15] a scene as "*usually composed of a small number of inter-correlated shots, that are unified by location or a dramatic incident*". Bordwel states in [16] that "*feature films are usually divided into three acts, each of which consists of about a half-dozen scenes*". He also writes that scenes of narrative content contain often one or several interleaved narrative events and can be bordered by introduction and conclusion shots. Hence, with this we know that scenes can contain one of multiple interleaved narrative events, further called parallel shots, but parallel shots by definition never cross a scene boundary because they form a story entity.

In general, a semantic scene conveys a special message or meaning to the audience to understand the flow of the story. Hence, we define for the moment a scene with following set of rules:

- Scenes consist of one or more shots conveying one single, consistent underlying semantic or narrative element;
- Scenes may incorporate one or more interleaved narrative events, i.e. cross-cuttings, or dialogues, i.e. shot reverse shots. Scene boundaries may not appear inside a parallel shot sequence;

2.2 System Architecture – the prototyping framework

2.2.1 Multimedia content analysis prototyping framework

The technology and consumer trends of section 2.1 showed that a terabyte of storage capacity on individual *Consumer Electronics* CE devices, and several terabytes of storage with massive processing capabilities and connectivity bandwidth within In-Home networks, no longer belong to the realm of fiction. Ubiquitous and pervasive content creation CE devices, such a mobile phones and cameras, boost the production of private content. The latter is stored together with commercially produced content in a scattered fashion across available enormous memory resources distributed across networked In-Home devices. The massive acquisition stir consumers into the dilemma of multimedia retrieval and management, a problem, which can be resolved by augmenting the material with content- and to some extend context-awareness. Fortunately, distributed, but connected processing and memory faculties of future CE In-Home networks provide sufficient computational resources to perform the required content-awareness creating multimedia content analysis and to memorize the generated content-awareness-creating content descriptors. It is, therefore, natural to consider future In-Home CE networks efficiently and transparently sharing their functionalities, content and resources (memory, processing). The latter is of importance, because, as the consumer trends in section 2.1 unveiled, users desire to interact with devices in an intuitive way, e.g. on a semantic level. The here for required semantic metadata cannot be extracted by only applying isolated mono-disciplinary content analysis algorithms, but instead this requests for a syndication of results from multiple modalities, i.e. content analysis results from independent content-awareness creating algorithms each e.g. exploiting one sensorial signal in isolation, as we state in our publication [17].

Such a federation of smart content analysis engines leads, naturally, to increased software (SW) complexity, which demands new software architectures and development frameworks efficiently exploiting the capabilities and information available across autonomous and heterogeneous peers within the network. Several teams are already working on such advanced distributed processing architectures. *Defence Advanced Research Projects Agency's* (DARPA) 'Hypersmart Computer Systems' project develops systems that can maintain itself, assess its performance and adapt itself context dependent, as Halal describes in [17]. In addition, IBM's 'Automatic Computing' project develops computer networks that are able to solve network problems dynamically and autonomously to accomplish undefined goals similar to a human organism [17]. Huhns goes even further in [19] and foresees that a network of

computing entities will become soon conscious and sentient. Huhns differentiates here for in [19] between 'outwardly perceiving non-mental entities', i.e. knowledge about external system parameters e.g. input-output data, and 'inwardly perceiving own mental entities'. The latter includes, based on Huhns definition in [19], system self-consciousness and self-awareness, i.e. knowledge of ongoing internal processes, of internal states of system, e.g. buffers, connections and system usage. Moravec, in additions, foresees that computing systems will grow soon beyond human capabilities to manage them, as described in his work in [20]. Therefore, he predicts that those systems soon need to have several additional features. The latter are (a) self-optimization, i.e. being able to automatically manage available resources, (b) self-configuration, i.e. dynamically arrange itself based on the given requirements and circumstances, (c) self-healing, i.e. being able to determine system problems and recover accordingly, further referenced as auto-recovery, and (d) self-protection, i.e. being able to defend itself against e.g. unauthorized access.

The complexity of the task of our work forces us to consider as well developing a distributed modular content analysis framework. It should admit self-organization, self-awareness, dynamic resource management for efficient workload distribution of modular processing tasks and, on the long run, a transparent cooperation of connected heterogeneous CE devices e.g. for real-time semantic content-awareness creation. But, especially the product development, assessment and evaluation cycle demands for such a framework, because many independent expert engines need to be syndicated in an efficient, quick and hassle-free way. Such a product-concept assessment contains four phases, i.e. (a) the imagination-, (b) the invention-, (c) the implementation- and (d) the inspection-phase, as published by us in [21]/[22] and sketched in Figure 10 for the application of *Advanced Content Navigation*.

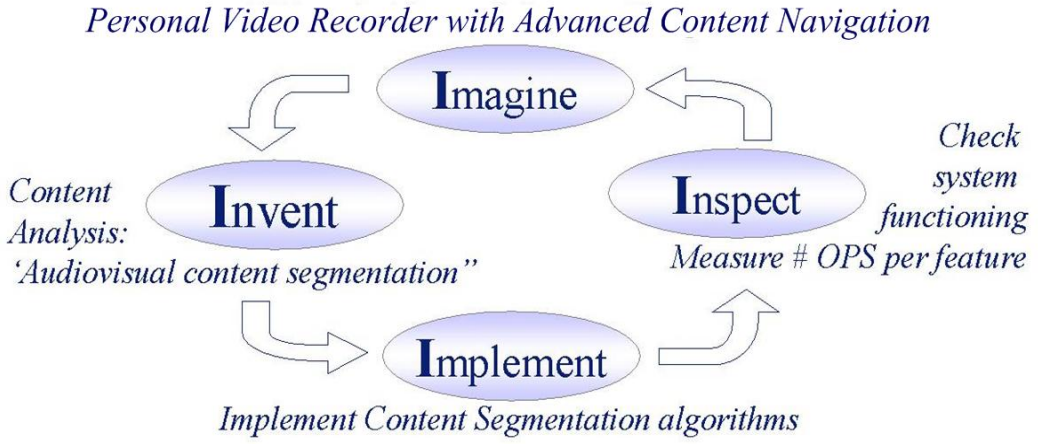


Figure 10. Product concept evaluation based on multimedia content analysis example.

Firstly, one envisions a new product or application, for example an *Advanced Content Navigation* feature for PVRs, which cleans the content from non-content related inserts, such as commercials, and adds semantic meaningful chapter markers throughout the recorded content item. In the next stage one invents the enabling technologies required to realize the envisioned application such as a *Shot Boundary Detector*, a *Commercial Block Detector* and *Scene Boundary Detector*. In the implementation stage, one looks for available technology solutions and prototypes critical parts to learn more about their technical requirements and limitations. Finally in the inspection phase, one checks the system behaviour, measures important characteristics and checks with the final user the attractiveness and intuitiveness of the product concept. If the concept fails the circle has to be restarted. But in almost all case, because of the fact the applied technical features, e.g. content analysis algorithms, are often in their infancy they are often subject of frequent changes. Hence, the prototyping framework should offer during the development time a transparent implementation of each feature as-is, i.e. non-optimized and hence processing demanding, into the framework. This requires a very processing powerful and performance-scalable prototyping framework allowing transparent integration of e.g. new content analysis modules and a seamless upgradeability of existing ones.

For our prototyping framework, which we aim to use to evaluate, test and verify our results in real-time, we decided to use an inhereted simple PC network solution, where each PC simulates a processing and memory node in a grid and data stream across the grid. For example, for a simple video streaming application realized e.g. on one PC one requires three software components, i.e. an encoder, database and decoder. These three components have to be controlled by a control instance preferably using a standardized interface to guarantee interoperability and extensibility, as sketched in Figure 11.

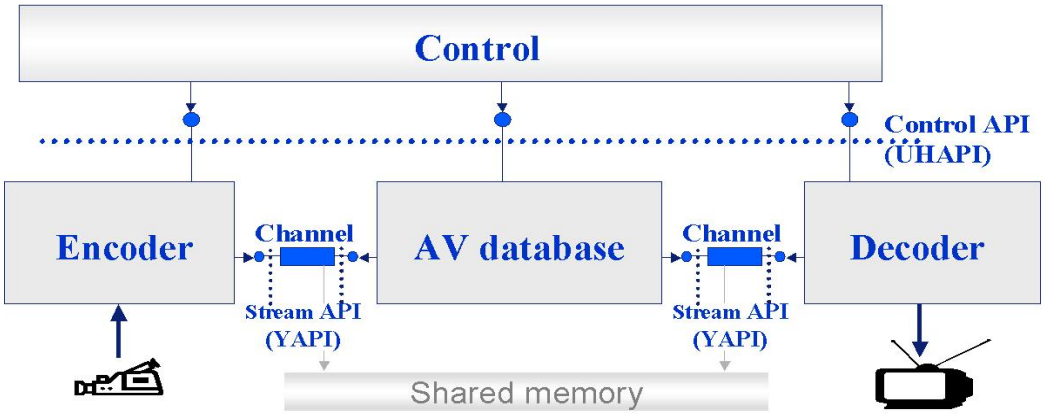


Figure 11. Streaming application with control interfaces – local processing.

For our framework we chosed together with the co-author of [21] as control *Application Programming Interface* API the especially for CE In-Home networks deployed *Universal Home API* UHAPI, as specified in [23] and published in [21]. For the independent streaming API, which is required to pass streaming audiovisual data from component to component, sketched in Figure 11 as horizontal data path, we choose the especially deployed streaming interface YAPI [25]. For streaming the audiovisual data between the components over the network we apply TCP/IP sockets. But, although we describe here our real-time streaming prototyping framework the concept and the interfaces are as well applicable for an offline implementation of the application running on a consumer device.

As stated in Figure 10, we do not intend to optimize algorithmic components during the *Implementation* and *Inspect* stages, at least as long the application was not evaluated, hence, the individual components are still quite processing demanding. Therefore, each single processing unit, i.e. here a PC, is often not enough to assess the combined multi-component functionality and, therefore, the components run at different units, i.e. PCs, in the network. To be able to control and set-up components remotely each component is extended with a networking functionality, as sketched in Figure 12 and published by us in [24]. For each logical component a proxy is introduced at the client control site and a stub are introduced at the real logical component, e.g. dedicated hardware. As communication control and notification protocol we choose *Universal Plug and Play* UPnP [26], because it established itself as standard for PC and CE In-Home networks and other for our framework useful functionalities.

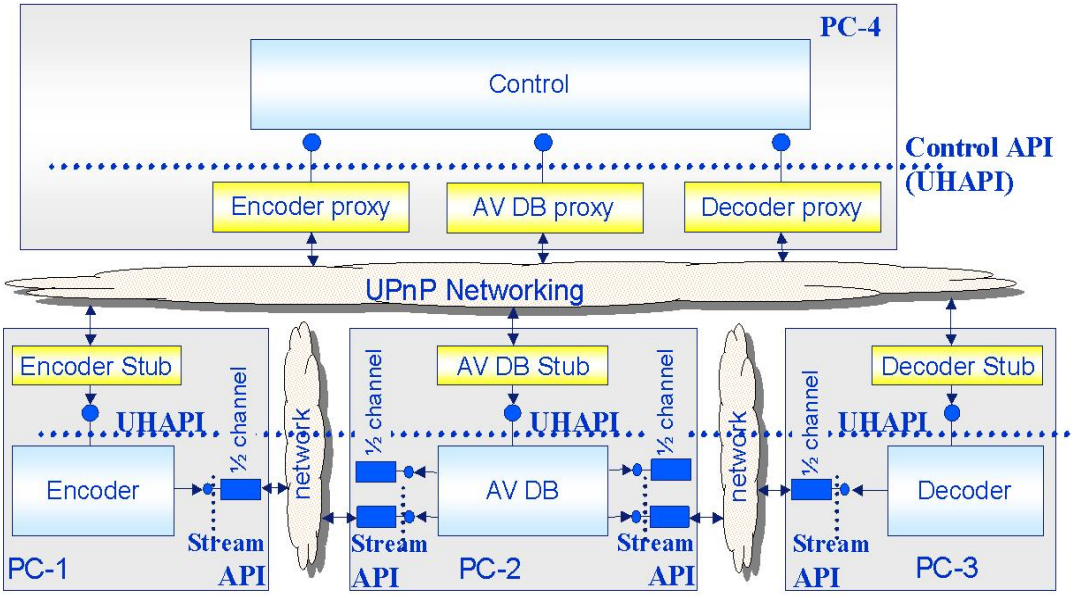


Figure 12. Streaming application with control interfaces – distributed processing.

It enables an UPnP control point e.g. to automatically discover new UPnP devices, i.e. physical components, in a network, to execute remote procedure calling and remote control of UPnP devices and their embedded UPnP services. Automatic discovery, i.e. dynamic service discovery, is started as soon as either an UPnP control point or an UPnP device is added to the network. In the first case the control point actively searches for UPnP devices and in the second the UPnP device broadcasts its presents, as further described in [24]. Furthermore, UPnP devices and services dynamically propagate their internal state to the UPnP control point and, hence, the system is self-aware what concerns the state of its components. Setting-up specific applications, here called *use cases*, e.g. *Advanced Content Navigation* requires a dedicated component here for, a *connection manager*. The latter contains an UPnP control point, establishes for every discovered UPnP device a device proxy and connects UPnP devices accordingly to the predefined use case, i.e. application, as sketched in Figure 13 and described in [24]. Through the framework's modular nature using UPnP devices running on remote processing nodes, i.e. PCs, adding new functionality such as new content analysis components only requires to add an additional processing node to the framework, as sketched in Figure 14. Each content analysis component, i.e. content analysis algorithm, is now encapsulated by an UPnP device layer, which communicates with the control point and other UPnP devices through the standardized interfaces and exposes its capabilities as service to the network.

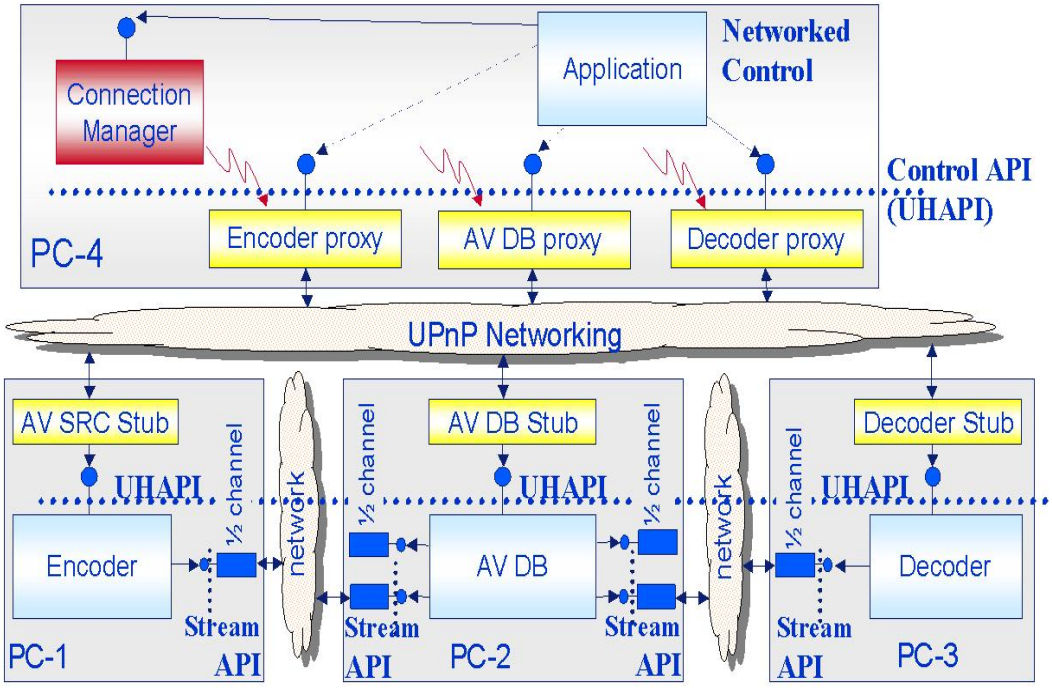


Figure 13. Application with Connection Manager for distributed streaming system.

Each content analysis component, further referenced as *Service Unit* SU, is seen by the framework as black-box with an defined input and output. Hence, algorithm developers can here with transparently upgrade, i.e. replace, their solutions and our Service-Unit-approach allows to clearly split system-architecture-related work from algorithm-related work. Furthermore, the usage of an standardized control API controlling logical components allows the application middleware to be unaware of the real implementation, i.e. real components such as hardware or software building blocks, and, hence, implementation-independent and portable to other platforms. Raw content data are stored within the framework in a real-time-file-system-based database. The descriptor data, i.e. metadata, are stored in a central SQL database. But for the future, we envision transparent data management across connected heterogeneous peers applying an Ambient DB data management layer logically interlinking the underlying databases and *DataBase Management Systems* DBMS, as we summarize in [27].

Finally, our distributed service-oriented analysis framework hosts a multitude of disciplinary–independent analysis algorithms developed and integrated by independent specialists with fluctuating programming capabilities, resulting in a high software failure probability. Hence, because of our frameworks complex and *Service-Oriented Architecture* SOA nature and its high probability of software failure we include as well a health monitor component in our framework, which monitors the health status of individual UPnP devices in a hybrid way, i.e. central-watchdog- and distributed heartbeat-message-approach, as we present as well in [28].

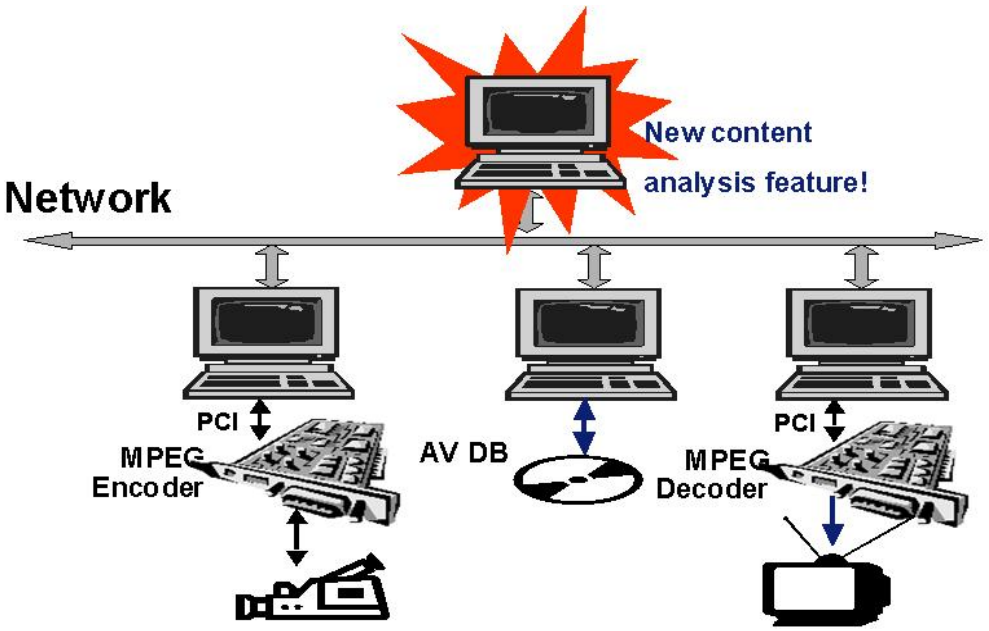


Figure 14. Scalability of prototyping framework.

The self-healing and auto-recovery mechanisms of the health monitor masks the errors and, hence, make the error recovery process transparent to the application guaranteeing the required *Quality Of Service* QoS, as we also mention in [29]. In addition, we also include a self-optimization component, which efficiently exploits the networks resources applying dynamic load balancing, also stated in [30].

Finally, we include as well a Use Case Manager *Graphical User Interface* GUI, as shown in Figure 15, which visualizes a selected group of *Service Units* SU required for a specific application, i.e. use case. The use case manager GUI enables the selection of required service units, in Figure 15 in blue, and the appropriate definition of data connections between these service units for a specific use case. The raw data flow connections, in Figure 15 in pink, and metadata flows, in Figure 15 in green, are established and the auto-configuration component, i.e. load balancing, distributes the service units according to their resource requirements across the available network nodes. During execution time a dedicated *Health Monitor and Fault Recovery* UPnP control point monitors the behaviour of all Service Units, i.e. UPnP devices and services, and reacts accordingly in the case of misbehaviour, i.e. informing dependent service units of the unavailability of crashed service units, but also the recovery of crashed service units at the same or a different node and, subsequently, re-establishing of connections after re-healing.

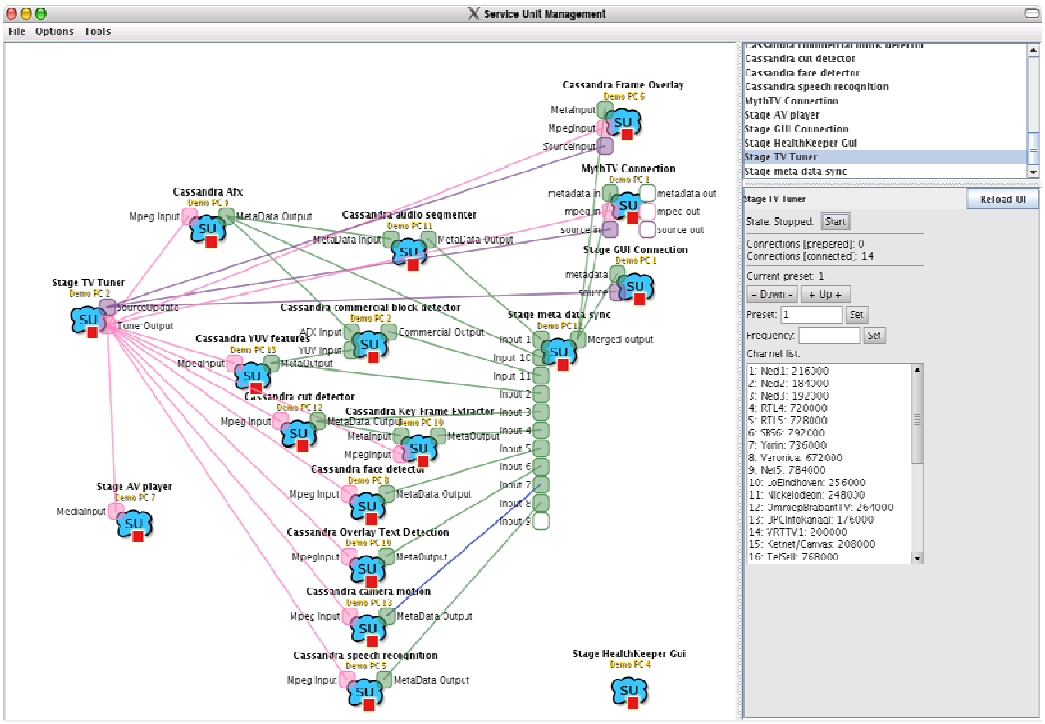


Figure 15. Use case manager graphical user interface showing service unit network.

In Figure 16 we sketch a possible set-up for the desired use case application *Advanced Content Navigation*, which we will elaborate in more detail throughout this work.

Our framework offers now all of the attributes we needed for fast prototyping exploiting the resources of a network efficiently, i.e. upgradeability flexibility, extensibility, self-configuration, self-awareness, self-adaptation and self-healing.

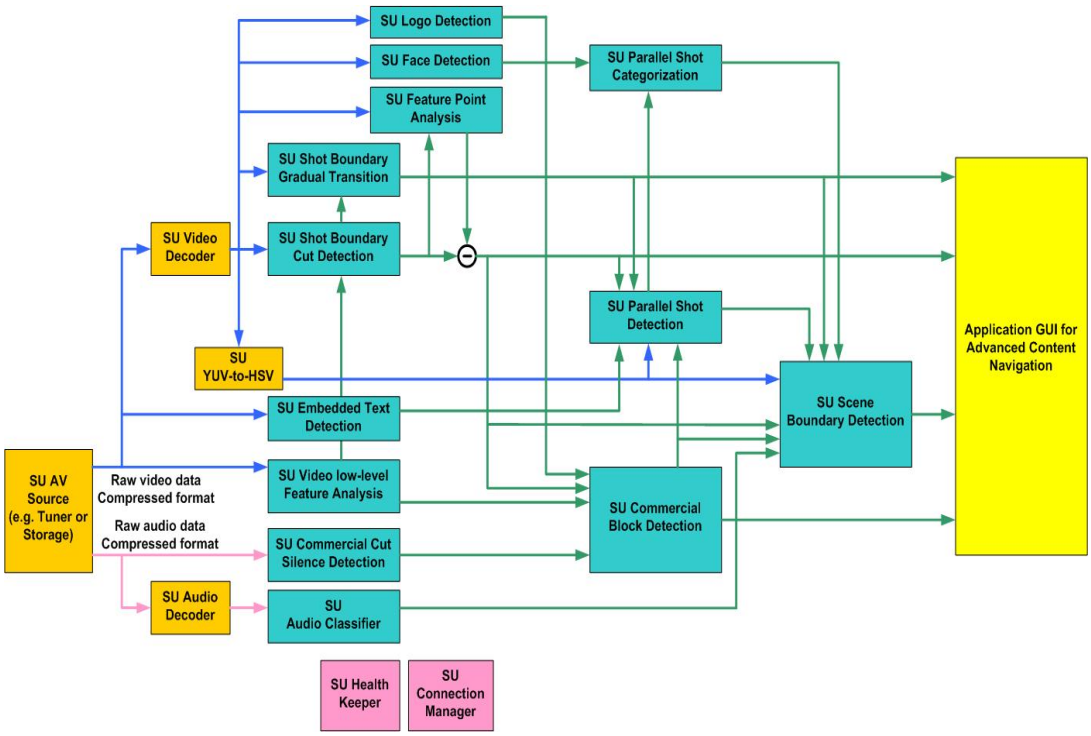


Figure 16. Envisioned set-up of Use Case *Advanced Content Management*.

2.3 Service units in distributed content analysis prototyping framework

In section 2.2.1 we presented the product concept evaluation cycle, as sketched in Figure 10. In the first phase we invent the application product. For this work we first selected one application out of a range of possible ones. The potential application are ‘Zap Circle’ and ‘Intelligent Channel Zapping’, where multiple channels are analyzed simultaneously and channels are provided to the consumer in a clustered way, e.g. by genre, while zapping through, as published in our patents [12]/[31]/[32]. Another one is ‘Clean Recording Movie Enhance’, where non-content related inserts such as commercials are removed from the content items and subsequently the clean recording is stored onto DVD, as claimed in our patents [33]/[34]. Finally, we decided to elaborate an extension of application ‘Clean Recording’, i.e. ‘Advanced Content Navigation’. The latter not only removes non-content related inserts, but also segments the content into semantic meaningful entities, so called scenes or chapters. Because of the applications semantic level multiple modality independent content analysis features need to be fused together to reach semantic content-awareness. The complexity of the required fusion framework and the heterogeneity of expertise required syndicating multiple content analysis modalities are the main reason that only few fusion examples are available so far. For our work we used our in section 2.2 described content analysis prototyping framework enabling independent expert teams to integrate their audio-, speech, image- or video content analysis (expert) algorithms into Service Units in a time efficient, transparent and effortless way. The resulting service-oriented application-generic but domain-specific for the moment content analysis engine, as described in detail in our publication in [35], hosts now a multitude of disciplinary-independent analysis algorithms allowing the fusion reaching semantic levels. The latter allows reaching human-communication-like content-awareness, allowing human-like search queries or interaction with the system.

2.3.1 Audio and video service units for *Advanced Content Navigation*

During the invention phase we identified several disciplinary-independent content analysis service units required for our intended application ‘Advanced Content Navigation’. Firstly, audio- and video feature extraction, i.e. low-level features, is required for the higher-level analysis engines. These extracted features are provided to audio- and video mid-level content analysis blocks such as a video shot boundary

detector and an audio silence detector, as sketched in Figure 17. Subsequently, the output results of these mid-level units are fused to e.g. identify scene boundaries. In our simple case these scene boundaries are detected by the temporal correlation of audio silences and video shot boundaries, as we describe in more detail in [36]. Hence, with this schema we have reached a first concept, which concludes our first invention phase.

To allow an objective development and benchmarking of the individual service unit algorithms led we aim to split the problems of the individual service units from each other. Hence, each service unit will be evaluated and benchmarked against its own ground truth. Nevertheless, the latter is used for subsequent dependent service units as input. In this way the development of individual service unit algorithms has no influence on the results of subsequent service units and, therefore, objective benchmarks are possible. In addition, in the end of this work a brief evaluation was given on the impact of the real robustness on the individual Service Units on the final segmentation results.

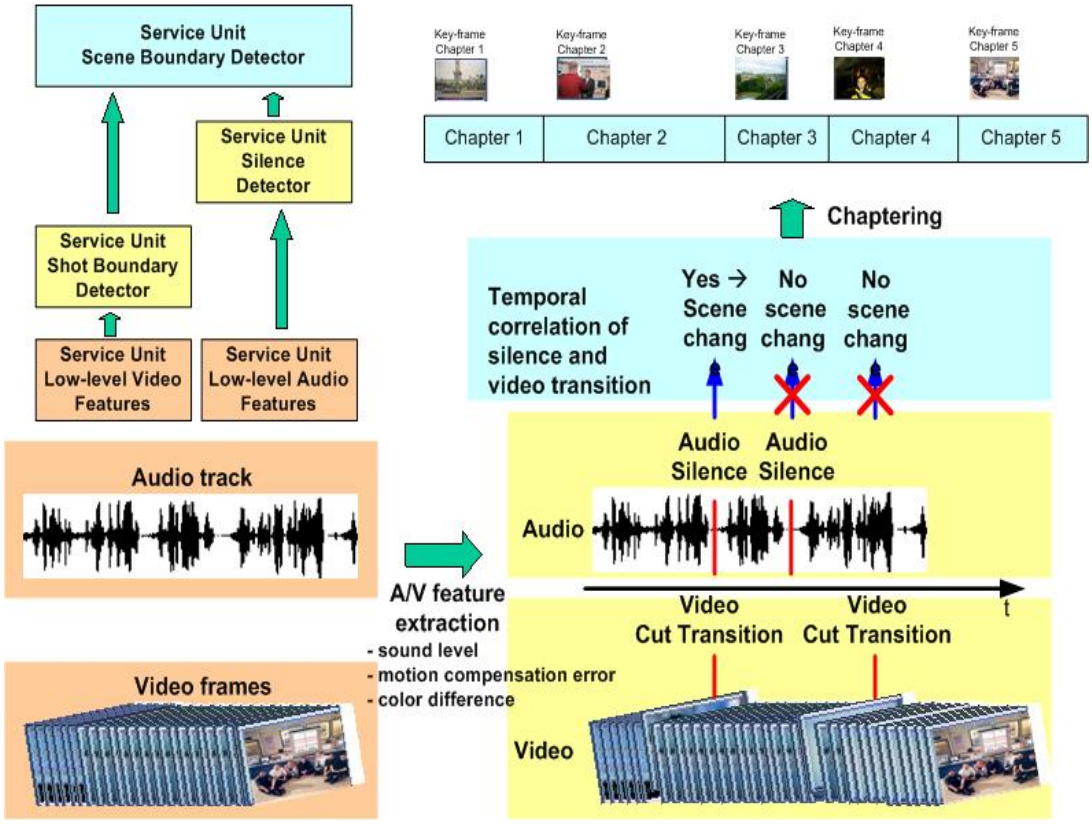


Figure 17. Application 'Advanced Content Navigation' with required service units[1].

2.4 Conclusions

The analysis of consumer and technology trends of this chapter unveiled that the massive technology capabilities scattered across our processing- and memory powerful inter-connected and broadband connected consumer In-Home devices demanded to consider new concepts. Exploiting the distributed resources in an efficient way and to be able to cope with complexity of the software and hardware architecture stimulated applying *Service-Oriented Architectures* SOA concepts and to exploit grid-computing-like technologies. The complexity of multimodality content analysis engines, i.e. fusing the results of disciplinary-orthogonal content analysis expert teams, motivated to elaborate a content analysis prototyping framework. In the latter the individual mono-modal content analysis components were embedded in *Service Units*, which were realized by UPnP devices and services. An UPnP control point, here a *connection manager*, initiated the individual Service Units required for a specific *use case*, i.e. application, connected the Service Units according to the use case's specification and controled the behaviour of the units hereafter. Flexibility, upgradeability and portability were guaranteed by using standardized interfaces within the framework, i.e. UHAPI / UPnP for the vertical control interfaces and YAPI / TCP/IP for the orthogonal horizontal streaming interfaces. Stability and QoS were secured by dynamic self-configuration and self-healing components such as the *Health Monitor*. Hence, our prototyping framework provided a good platform to implement and test selected applications in an efficient and transparent way. Following the product concept evaluation circle we *imagined* and *invented* a specific application, i.e. 'Advanced Content Navigation', requiring various modality-independent content analysis modules. Our framework served as prototyping platform for this application.

Hence, in this PhD work we will investigate research and develop existing and new concepts and methods for our service units, which will allow us to accomplish the task of semantic content segmentation. First of all, we will present in the next chapter the state-of-the-art of potential concepts and methods, which we consider as useful for our application task.

CHAPTER 3

3 STATE-OF-THE-ART AV SEGMENTATION METHODS

In the previous chapter we introduced the concept of *Service Units* SU and a related distributed architecture for the task of semantic segmentation of audiovisual content and in particular for the application 'Advanced Content Navigation'. We specified that each SU is in charge of one task of content analysis or segmentation, which can be either low-level, mid-level or high-level. Now the goal is to fill all these service units with a specific algorithm to accomplish the application task. Hence, before we can propose a solution and design a method for each SU, we will present, in this chapter, the state-of-the-art in AV segmentation methods. First of all, the shot boundary detection task is of primary importance. Mainly because shots are the elementary units of video content, i.e. individual recordings, which have been concatenated together during the post-production editing cycle. At a higher semantic level, clusters of correlated shots form meaningful entities, so called scenes or chapters. Many attempts have been published to identify those meaningful transition instances. Therefore in the second part of this chapter, we will give as well an overview of audio related segmentation work and, thereafter, an overview of relevant video segmentation works published by various teams active in this specific content analysis domain.

3.1 Video mid-level features for AV segmentation

Digital audiovisual content can be considered as a combination of two signals. One is a temporal signal *a(t)* representing the audio stream. The second one is a 2D signal, i.e. video stream. In order to perform AV content segmentation various low-level audio and/or video based features can be exploited, as for example Louis explains in [37]. In our work we will concentrate on mid-level and high-level features, which consider low-level features as basic descriptors, which can be extracted from compressed or raw AV signal. Contrary to most low-level features, e.g. colour histograms, mid-level features already contain semantically meaningful audiovisual information. These visual based mid-level features span from

- shot boundaries,
- specific object identification including text and faces,
- camera motion to key frames.

Nevertheless, our video mid-level feature set is not exhaustive, but the most prominent ones required for our selected application are covered here.

3.1.1 Shot Boundary Detection

The production of content is elaborated as a process following generally predefined production rules. Before a content item is produced a (shooting) script is written describing the story's flow. Subsequently a storyboard is drawn to visualize the individual elements of the story. Simultaneously during the shooting multiple cameras, as sketched in Figure 18 (left), capture the scenery and the target objects or people each called a continuous *take*, which is repeated on average four times resulting in four takes per camera position.

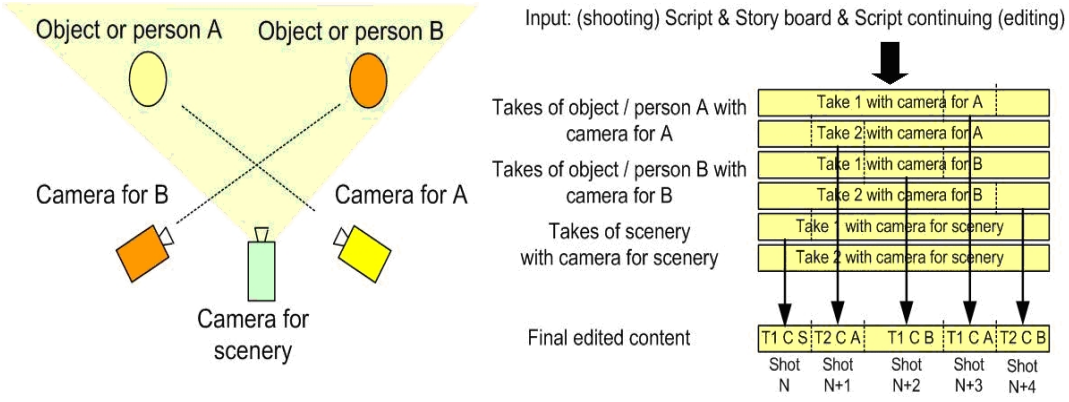


Figure 18. Shot-based editing of camera takes and various camera positions during takes.

Subsequently, based on a continuation script, the editor chooses the best sub-sequences from the multiple takes and concatenates them - further referred to as *shots* - as shown in Figure 18 (right), each comprising a sequence of consecutive frames. The ratio between takes and final edited content is on average 12:1, according to [16]. *Shot Boundary* SB analysis is an essential element since video shots can be seen as fundamental to a multitude of mid-level and high-level applications based on content analysis. During the editing process the editor can either concatenate two shots, which results in an abrupt instantaneous frame-to-frame *cut* transition, as shown in Figure 19 (middle), or he can use an artistic continuous transition spanning several frames creating a smooth *gradual* transition, which includes wipes, fade-ins, fade-outs or dissolves, as shown in Figure 19 (right). Both types are clustered into the group of shot boundaries.

Automatically identifying, i.e. retrieving, theses editing instances, i.e. shot boundaries, can be seen largely as a reverse engineering of this editing process. Shots are consistent entities and their individual data contents, but also the intercorrelation between them provide valuable insights about the contents' message and story line. A reliable shot boundary detector is, therefore, of utmost importance for our work. The detector used to retrieve temporal video segmentation of this type is called a *shot boundary detector* SBD, and it consists of a *cut detector* CD and a *gradual transition detector* GTD, as shown in Figure 19 (left).

Shots, which described by Brunelli in [38] as the basic unit of video structure, can be further segmented by means of camera motion activities such as zooming, tilting and panning, which is not dealt with here in this work.

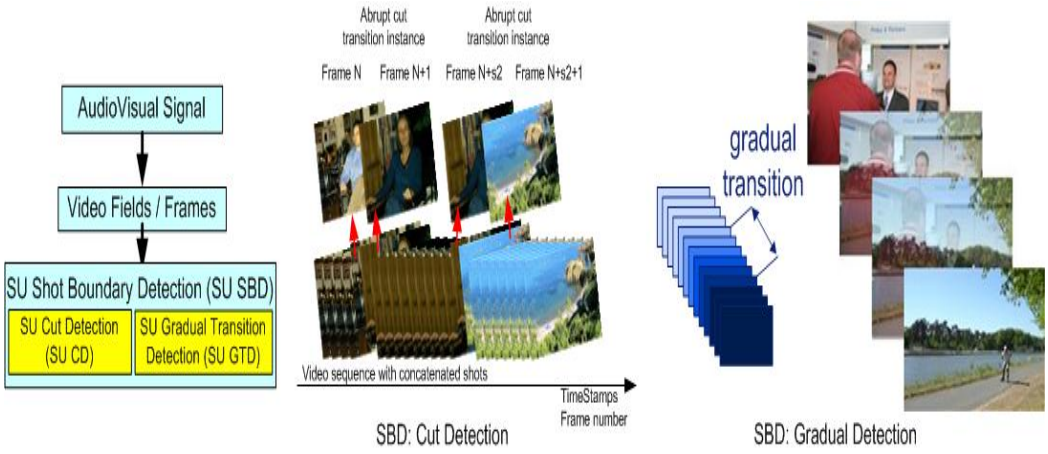


Figure 19. Shot boundary examples – cut and gradual transitions[1].

Nonetheless, shot boundaries themselves are essential for various video technologies, such as efficient video coding in MPEG-2 or H.264, where the first frame of a shot could be encoded as an I-frame. Moreover, shot boundaries are required for enhanced video algorithms such as 2D-to-3D content conversion when the processes of segmentation, calibration and depth estimation require reinitiating. In addition, shot boundary detection is used for high-level applications such as audiovisual chaptering, where multiple shots are clustered into semantically meaningful scenes, as described in [36], and which is the aim of this PhD.

Since the last decade, extensive research has been conducted in search of an efficient SBD algorithm due to its significance in the domain of *video content analysis* VCA. Despite the large number of proposed cut detectors reported in e.g. [39], [40], [41] and TRECVid [42], a superior algorithm, capable of detecting all possible transitions, highly accurate and entirely independent of the video sequencing properties, has not yet been discovered as summarized in [41]. Therefore, we will research and evaluate promising cut detectors and gradual transitions detectors separately from each other in this work.

Cut Detection

In general, the process of *cut detection* is common to all of them and can be divided into three stages as presented in [43], (a) the extraction of an appropriate video feature, (b) a metric–based frame-to-frame consistency measure (inter-frame correlation coefficient) followed (c) by a metric inconsistency evaluation instance, which indexes instances as cuts once a certain threshold is exceeded.

Various solutions for (a) and (b) were published dealing with robust and reliable metric-based frame-to-frame consistency measure for both the compressed as well the uncompressed video domain.

For the uncompressed domain, e.g., Luo used in [44] and Kikukawa in [45] pixel-based consistency values by summing up the absolute intensity changes of corresponding pixels, but this method is sensitive to camera and object motion. Successively, Zhang improved in [46] this method by adding a pre-processing stage. Shahrara further improved the system in [47] by adding a block-based motion estimator succeeded by a block-by-block comparer.

Another common method is based on histogram consistency, also called the histogram intersection method, which is less sensitive to weak camera and object motion, but more sensitive to illumination conditions, as Yeo describes in [48]. For this, the *red-green-blue* RGB colour space is divided into *B* discrete colours, called *bins*, and the number of pixels members of each discrete colour bin is counted. Hereafter, the normalized sum of the absolute difference of corresponding bins, as presented in [49],

which is the normalized intersection $S \epsilon \{0,..1\}$ of the resulting histograms HST_i and HST_j of two images $F_{(n=i)}$ and $F_{(n=j)}$, respectively, is calculated by

$$S\left(HST_{\ i}, HST_{\ j}\right) = \frac{\sum_{k=1}^{B} \min\left(HST_{\ i,k}, HST_{\ j,k}\right)}{\sum_{k=1}^{B} HST_{\ j,k}} \qquad (3\text{-}1),$$

where $HST_{i,k}$ represents the value of HST_i in the k-th bin. S gives an indication of the amount of pixels of $F_{(n=i)}$ and $F_{(n=j)}$ with similar colour. For cut detection the dissimilarity value

$$D\left(HST_{\ i}, HST_{\ j}\right) = 1 - S\left(HST_{\ i}, HST_{\ j}\right) \qquad (3\text{-}2),$$

has been applied. In general, various colour-histogram-based models can be used for this approach as Gargi summarizes in [50] by comparing YIQ, L*a*b* and Munsell colour spaces. An enhanced version of this method consists in including the spatial colour distribution, by e.g. dividing the frames into equal sized regions or blocks as Nagasaka describes in [51] using X^2 test on regional histograms. Nevertheless, the reliability of histogram-based methods is limited by camera motion and the fact that different frames can share similar histograms even though the content is completely different.

Besides pixel-intensity- and histogram-based cut detectors, various segmentation-based cut detectors have also been introduced e.g. by Zabih in [52] and Yusoff in [53]. Based on the theory that at cut instances the new edges appear relatively far from the old edges of the previous shot, the number of non-matching edge pixels is counted. In addition, motion compensation is applied to make the algorithm motion insensitive, increasing the computational expenses of those methods. Unfortunately, edge-based methods have problems with fast moving objects. They also heavily dependent on the accuracy of the applied edge detection algorithm.

In parallel, various compressed-domain cut detection solutions have been developed for processing-constraint environments, as summarized in [40] by Koprinska. One example of this is a DC-image- and macroblock-based method using the number of intra-, forward- and backward-coded MBs to decide on cut instances as described by Meng in [54]. Using only the DC information of I-frames Patel presented another example for compressed domain analysis in [55]. Three histograms – a global-, a row- and a column-based one – of two successive I-frames were compared to one another using a Chi-square X^2 test to identify discontinuities in the video stream. In [56] another compressed domain shot boundary detector was introduced, further referenced as *rough indexing cut detection* RI CD, which is based on I-, P-frame and global camera

motion features. As the detector was benchmarked on the TRECvid 2004 corpus, we consider applying this detector as objective benchmark for our own cut detectors in this work. The method is based on two assumptions: presence of (a) motion changes, which does not always hold in real content but is realistic in MPEG encoded motion, and (b) spatial content changes at cut instances. The method consists of two cooperative processes running on an MPEG stream: change detection in P-frames and I-frames. In P-frames, it is supposed that macroblock motion vectors $(dx_i, dy_i)^T$ follow a single affine motion model for the frame:

$$\begin{aligned} dx_i &= a_1 + a_2 x_i + a_3 y_i \\ dy_i &= a_4 + a_5 x_i + a_6 y_i \end{aligned} \quad (3\text{-}3),$$

with (x_i,y_i) representing the coordinates of individual macroblock centres. Hereafter, the normalized absolute differences of estimated motion parameters for consecutive P-frames $\Delta^* a_m(n)$ and an absolute difference of the number of intra-coded macroblocks $\Delta Q(n)$ form a multiplicative mixture $D(n)$ used to detect a cut transition:

$$D(n) = (|\Delta Q(n)| + 1)^{\beta} \left(1 + \sum_{n=1}^{6} \Delta^* a_m(n)\right)^{1-\beta} \quad (3\text{-}4),$$

with $\beta \in \{0..1\}$, and set to 0.8 by default. Supposing a Gaussian distribution $N(\mu,\sigma)$ of $D(n)$ inside each shot, a shot–adaptive detection threshold $\lambda=\mu(D,W)+T\sigma(D,W)$ is trained during the W first P-frames. A shot boundary is indexed at instance n where $D(n)>\lambda$ and $D(n)/D(n-1)>\alpha$ with a consistency check of the sign of ΔQ. For the parallel I-frame path, the change detection is based on spatial content matching. DC representations of consecutive I-frames are warped using motion-estimation-based compensation. After warping, a weighted mean squared error $WMSE(k)$ with

$$WMSE_{n+k} = \frac{1}{|v|} \sum_{p \in v} w(x'_p, y'_p)^2 * \left(DC_{n+k}(x_p, y_p) - DC'_n(x'_p, y'_p)\right)^2 \quad (3\text{-}5),$$

is applied as a similarity measure, which is weighted by the inverse of the energy of the local image gradient $w(x'_p,y'_p)$ in order to reduce the contribution of errors on image contours. Then, instances fulfilling $WMSE(k)>T_h*\mu(WMSE,W)$ trigger to insert a shot boundary index prior to the related I-frame $I(t_{k+1})$.

After the calculation of a metric-based frame-to-frame consistency value, either simple fixed-threshold-based or more advanced variable-threshold-based methods can be applied to make the final cut detection decision. In general, most of the abrupt transition detectors use some kind of an adaptive threshold mechanism, meaning that the threshold value is computed locally for each frame, when considering the nature of past and future frames. The detector is therefore able to distinguish low consistency values inside a shot from those at the shot boundaries. In [53], in this regard, Yusoff has given an overview and performance evaluation of common threshold methods.

Gradual Transition Detection

The problem of gradual transition detection and classification is much more complex. During gradual transitions the contribution of one signal – pictures of shot sh_N – decreases whilst that of another signal – pictures of shot sh_{N+1} – increases as shown in Figure 19 (right). Where the pictures of sh_N are solid in colour, the process is called fade-in and when the pictures of sh_{N+1} are solid it is known as fade-out. In the case of non-solid colours the process is called dissolve (Figure 19 right), resulting in the video signal $S_n(x,y)$. Other progressive effects are possible such as *store* or *wipe* or local dissolves, where a change is observed inside a region of an image known as *compositing effect* [57]. As fade-ins, fade-outs and dissolves are the most frequent transition effects, we will analyse here the detection of a linear dissolve. The model of intensity / colour signal in such a transition can be presented as follows [58],

$$S_n(x,y)=\begin{cases} f_n(x,y) & 0 \le n < L_1 \\ \left[1-\left(\frac{n-L_1}{W}\right)\right] f_n(x,y)+\left(\frac{n-L_1}{W}\right) g_n(x,y) & L_1 \le n \le (L_1+W) \\ g_n(x,y) & (L_1+W) < n \le L_2 \end{cases} \quad \text{(3-6)},$$

where $f_n(x,y)$and $g_n(x,y)$ are the pictures of sh_N and sh_{N+1}, respectively, and L_1, W and L_2 the length sequences of sh_N alone, the dissolve and the total length, respectively, as presented by Fernando in [58]. Under the assumption that the video sequences are ergodic processes the mean μ and variance σ expose a linear and quadratic behaviour, respectively, with

$$\mu_{s,n} = E[S_n(x,y)] = \begin{cases} \mu_f & 0 \le n < L_1 \\ \left[\mu_f - \frac{L_1}{W}(\mu_g - \mu_f)\right] - \frac{n}{W}(\mu_f - \mu_g) & L_1 \le n \le (L_1+W) \\ \mu_g & (L_1+W) < n \le L_2 \end{cases} \quad \text{(3-7)},$$

$$\sigma^2_{s,n} = E\left[S^2_n\right] - E[S_n]^2 =$$

$$\begin{cases} \sigma_f^2 & 0 \le n < L_1 \\ \xi n^2 - \left(\frac{2\sigma_f^2}{W} + 2L_1\xi\right) n + \left(\sigma_f^2 + L_1^2\xi + \frac{2L_1\sigma_f^2}{W}\right) & L_1 \le n \le (L_1+W) \\ \sigma_g^2 & (L_1+W) < n \le L_2 \end{cases} \quad \text{(3-8)},$$

with $\xi=(\sigma_f^2+\sigma_g^2)/W^2$, but in reality the process is not always ergodic due to motion. Fernando presents further in [58] a combination of the two parameter mean μ and variance σ^2 to identify gradual transitions, represented through sequences during which the ratio of the second derivative of the variance curve to the first derivative of the mean curve is a constant. Fade-ins and fade–outs are detected accordingly leaving the pictures of one sequence solid in colour.

Due to processing restrictions a simpler but also very efficient spatiotemporal block based gradual transition method of Naci, presented in [59], was applied. Another reason for this choice was that this method was benchmarked in TrecVid [60] and scored very well. After block based motion estimation, similar to the MAD method (see section 4.1.2), Naci applied a spatiotemporal block based analysis both in time direction and in the estimated motion direction on the intensity value blocks $I_{i,j,k}(m,n,f)$, with *i,j,k* corresponding the indices of the spatiotemporal block (*k* indexes the time) and *m,n,f* corresponding the pixel position within each block. Dissolves and fades are characterized by monotonously changing luminance values during a gradual transition. Naci detected the luminance flow monotonousness of a block *(i,j,k)* with

$$F_1(i,j,k) = \max\left(\left|\frac{\nabla_k^d I_{i,j,k}}{\nabla_k^a I_{i,j,k}}\right|, \left|\frac{\nabla_v^d I_{i,j,k}}{\nabla_v^a I_{i,j,k}}\right|\right) \quad (3\text{-}9),$$

by applying the absolute cumulative luminance change

$$\Delta_v^a I_{i,j,k} = \frac{1}{C_x * C_y} * \sum_{m=0}^{C_x-1}\sum_{n=0}^{C_y-1}\sum_{f=0}^{C_t-1}\left|\Delta_v I_{i,j,k}(m,n,f)\right| \quad (3\text{-}10),$$

and average luminance change

$$\Delta_v^d I_{i,j,k} = \frac{1}{C_x * C_y} * \sum_{m=0}^{C_x-1}\sum_{n=0}^{C_y-1}\sum_{f=0}^{C_t-1}\left(\Delta_v I_{i,j,k}(m,n,f)\right) \quad (3\text{-}11),$$

and hence, representing monotonous changing sequences by $F_1 \approx 1$. In addition, the smoothness (gradualness) of the dissolves / fades was specified by

$$F_2(i,j,k) = 1 - \min\left(\left|\frac{\nabla_k^{\max} I_{i,j,k}}{\nabla_k^a I_{i,j,k}}\right|, \left|\frac{\nabla_v^{\max} I_{i,j,k}}{\nabla_v^a I_{i,j,k}}\right|\right) \quad (3\text{-}12),$$

using in addition the maximum luminance change

$$\Delta_v^{\max} I_{i,j,k} = \frac{1}{C_x * C_y} * \sum_{m=0}^{C_x-1}\sum_{n=0}^{C_y-1}\left|\Delta_v I_{i,j,k}\left(m,n,f_{i,j,k}^{\max}(m,n)\right)\right| \quad (3\text{-}13),$$

where

$$f_{i,j,k}^{\max}(m,n) = \underset{f}{\arg\max}\left(\left|\Delta_v I_{i,j,k}(m,n,f)\right|\right) \quad (3\text{-}14),$$

expressed the time wise expansion of the gradual, e.g. with $F_2 \approx 1$ for a very smooth transition. Naci used the multiplication of $F_1(i,j,k)$ and $F_2(i,j,k)$ as the confidence value of gradual transition in the corresponding block. The average of all the block based confidence values with the same time index *k*=K is a measure for the probability of gradual transition in the video in the Kth time interval. Due to a very limited set of complex computer generated wipes in Naci's AV corpus (in total only 2 computer generated wipe-like gradual transitions), wipes have been ignored in his work.

Benchmark parameter for cut and gradual transition detection

The objective evaluation of various temporal video segmentation methods demands objective benchmark parameters using (a) a sufficiently heterogeneous reference benchmark corpus, (b) a manually (and therefore to some extent subjective) annotated algorithm-independent reference ground truth and (c) objective "quality measure" criteria.

An audiovisual benchmark corpus with manually annotated ground truth required for this work will be presented in section 0. Ruiloba summarizes the "quality measure" criteria in [61] comparing various solutions. The basic parameters for such a performance evaluation are $N_{Correct}$, N_{Missed} and N_{False}, which represent the number of correctly detected instances, missed instances (missed detections, also called false negatives) and falsely detected instances (false detection, also referenced as false positives or over-segmentation), respectively, as sketched in Figure 20. Herewith the

$$Accurancy = \frac{N_{Correct} - N_{False}}{N_{Correct} + N_{Missed}} \quad \text{or} \quad ErrorRate = \frac{N_{Missed} + N_{False}}{N_{Correct} + N_{Missed} + N_{False}} \qquad (3\text{-}15),$$

as proposed by Corridoni in [62] can be calculated, but none of the above take the complexity of the video sequence nor its size into consideration.

Hence, the most frequently used benchmark criteria, e.g. used by the benchmark competition TRECVid [42], is recall *Re* and precision *Pr* of Nagasaka described in [51], they will also be used in this work. Recall *Re* represents the percentage of correctly detected examples, here cut instances, in relation to all existing cut instances are further defined by

$$\mathrm{Re} = \frac{N_{correct}}{N_{Correct} + N_{Missed}} * 100 \quad , \mathrm{Re} \in [0\ \%,\ 100\ \%] \qquad (3\text{-}16).$$

On the contrary, precision *Pr* is the percentage of all correctly detected cut instances in relation to all detected cut instances, as calculated with

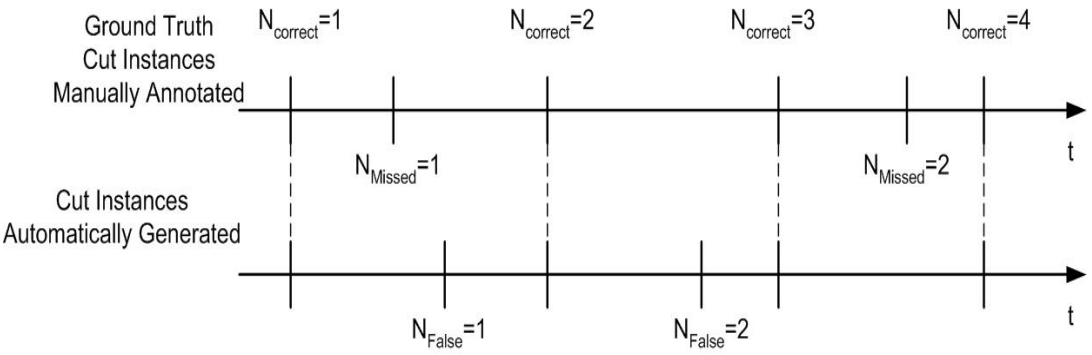


Figure 20. Description of $N_{correct}$, N_{False} and N_{Missed}.

$$\mathrm{Pr} = \frac{N_{Correct}}{N_{Correct} + N_{False}} * 100 \quad , \mathrm{Pr} \in [0\ \%,\ 100\ \%] \tag{3-17}.$$

Also other evaluation metrices were used for video indexing as well, such as those based on the intersection of shots, i.e. ARGOS metric of Joly [63] and the F-measure,

$$F = \frac{2}{\frac{1}{\mathrm{Re}} + \frac{1}{\mathrm{Pr}}} \tag{3-18}.$$

Nevertheless, we limit ourselves to the classical metrices recall and precision, as they remain the most widely applied ones, e.g. in TRECVid. We apply them to specify the performance of three own cut detectors (section 0).

3.1.2 Motion estimation and camera motion

Other MPEG-7-defined mid-level video parameters are motion trajectories and camera motion. The latter represents background motion, i.e. pure camera motion, which implies that foreground motion, i.e. motion of foreground objects, is excluded. Solutions described in literature can be categorized in three camera motion analysis methods, (a) a feature-based [64], (b) intensity-based [65] and (c) a method for estimating camera motion from initially estimated motion vector fields, e.g. applying MPEG-2 macroblock vector fields [66]. The feature-based method (a) identifies and tracks a set of features through a video sequence, here a video shot, whose movements are fitted into a motion model. The intensity-based method (b) uses derivatives of image intensities of selected image points, which is followed by a gradient-descent-based error-minimizing step to estimate motion vectors. The general case of motion estimation, which covers foreground and background motion, is an essential parameter for video compression e.g. as used for MPEG-2 video codecs, see annex 1. Background motion, i.e. camera motion, is required to create e.g. two/three-dimensional models of a scene, also called mosaicing as intended to be used in MPEG-4. On contrary, foreground motion, i.e. the motion of objects, is needed to incorporate virtual objects into a scene, e.g. as used in MPEG-4 object layers.

Transformation and motion methods

The available solutions described in literature are initially based on a general net motion (estimation) method, which includes foreground and background motion, and in a subsequent step background- is separate from foreground motion. The simplest transformation is probably the rotation, which may be represented as $\boldsymbol{x'}=H_R\boldsymbol{x}$, where $\boldsymbol{x}=(x,y)$ and $\boldsymbol{x'}=(x',y')$ represent the original and the transformed 2D coordinates, respectively, of an image point with rotation matrix

$$H_R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \quad (3\text{-}19)$$

wherein θ represents the rotation angle. Removing restrictions on the matrix elements leads to the general case of a linear transformation, and by applying *homogeneous coordinates* of projective geometry, as described in [67], individual points may be represented by triplets of $\boldsymbol{x}=(x,y,1)$. This increase of the transformation matrix to size *3x3* enables the incorporation of translation, which leads to the affine transform $\boldsymbol{x}'=H_A\boldsymbol{x}$, wherein $\boldsymbol{x}$ and $\boldsymbol{x}'$ are represented in their homogeneous form and the transformation matrix H_A, which, through further generalization, results in the projective transform matrix H with

$$H_A = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow H = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \quad (3\text{-}20)$$

Here $a_{11}..a_{22}$ are affine deformation parameters and t_x, t_y are translation parameters. Finally, the projective transformation, also known as *homography* or *homographic* transform, can be represented with $\boldsymbol{x}'=H\boldsymbol{x}$ with the homogeneous representations of $\boldsymbol{x}$ and $\boldsymbol{x}'$ or with the non-homogeneous coordinates

$$x' = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + 1}, \; y' = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + 1} \quad (3\text{-}21)$$

Hence, one, as in the case of H_R, to eight parameters, as in the case of H, can be recovered from the underlying motion method. Most estimation methods use middle between those two extremes and in [67] a hierarchical overview of those transformations is given.

Feature-based camera motion analysis methods

The feature-based camera motion estimation methods detect and track a set of selected image points through a video sequence. Farin provides in [64] a comparative evaluation of four representative detectors suited for this method, which are Shi-Thomasi, SUSAN, Moravec and Harris, wherein the latter is identified as the best performing one. The detected and tracked points provide a set of correspondences, which are required to estimate the motion parameters using a selected motion method. In the case of an eight-parameter projective method (each correspondence provides two equations) at least four points have to be tracked to allow a unique solution for H, but, because usually more points are tracked, least-squares methods are used to solve the over-determined equations.

Intensity-based camera motion analysis methods

The intensity-based method, also often referenced as *direct* or *optical flow* method, is based on directional derivatives of image intensities, e.g. luminance Y-values, at individual selected image points to retrieve the required motion parameters by mean of a gradient-descent-based error-minimizing step. Applying, for example, a three-parameter motion method, i.e. two images *I(n)* and *I(n+m)* differ only by a translation- (panning, *a* and *b*) and a scaling (zoom, *s*) factor, the relation between the images can be expressed by

$$I(x, y, n) = I(a + sx, b + sy, n + m)) \tag{3-22}$$

with x and y representing the pixel position in the image. The problem is, in this example, reduced to retrieve the best corresponding values for *a*, *b* and *s*, whereby most papers assume that initial estimates of the parameters are available and finding the incremental changes is done by minimizing an error criterion, as for example for

$$I(x, y, n) = I(a + \Delta a + (s + \Delta s)x, b + \Delta b + (s + \Delta s)y, n + m)) \tag{3-23}$$

wherein the right side is expanded using a Taylor series,

$$I(x,y,n) - I(a + sx, b + sy, n + m) = \left(\frac{\delta I(n+m))}{\delta a}\right)\Delta a + \left(\frac{\delta I(n+m))}{\delta b}\right)\Delta b + \left(\frac{\delta I(n+m))}{\delta s}\right)\Delta s \tag{3-24}$$

The equation has to hold for all image points of *F(n),* and the resulting set of equations can be written in matrix form using the Jacobian matrix ***M*Δu=e*** containing the partial derivatives of *I(n+m)* and a vector ***Δu*** containing the parameter updates, which can be subsequently solved by least-square methods. Hager revealed in [65], that the Jacobian matrix could be expressed in terms of derivatives of *I(n)* instead of *I(n+m)* reducing the complexity of the equation drastically.

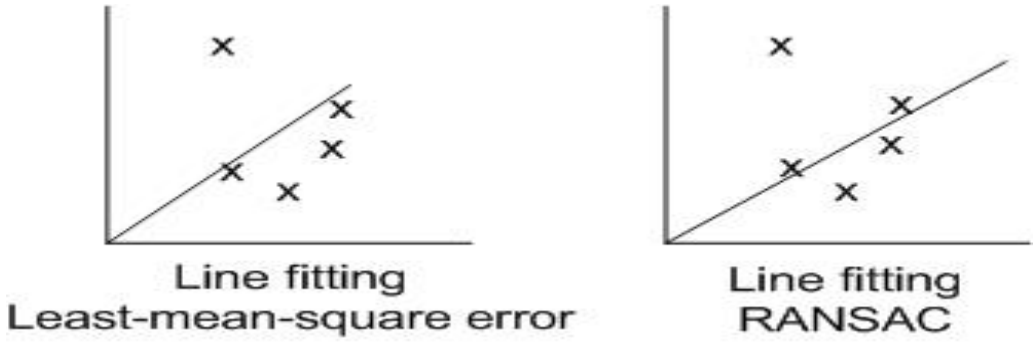


Figure 21. Line fitting methods for camera motion analysis.

Separation of foreground and background motion

The problem with camera motion analysis is, that foreground object motion has to be separated from the background motion, i.e. camera motion, a non-trivial problem. For feature-based camera motion analysis methods dominant motion can be separated from other motion by means of Random Sample Concensus (RANSAC) algorithm, as described by Hartley in [67]. The presence of highly deviant outlier, e.g. fast moving foreground objects, can affect the least-square method significantly, as shown in Figure 21, whereas RANSAC eliminates the outliers by approximating the result fitting the largest number of samples (in this case samples represent motion vectors and foreground motion would result in outliers).

A random set of samples is used to instantiate the method – here the homography – and the fit is defined by number of samples, which conform to the homography. Subsequently, the homography with the best fit will be selected.

For intensity-based camera motion analysis methods other techniques such as coarse-to-fine processing or M-estimators are used to recover the dominant motion as described by Iran in [68].

The problem of separation of foreground and background motions in a compressed stream was also addressed in [66] and further developed in [69]. Here the initial macro-block vectors of MPEG2 flow (see Annex 1) represent observed data. They are supposed to follow a global 6-parameter affine model similarly to H_R of equation (3-19). The estimation of a global model by a robust estimator, i.e. Tuckey function, allows removing outliers corresponding to foreground objects. The results the authors obtained for the camera motion characterisation task in the Trec Video evaluation campaign, as published in [70], show that the compressed stream motion descriptors can be used for estimation of a global model subject to application of a robust statistical estimator in order to filter the estimation noise.

Comparison of feature- and intensity-based camera motion analysis methods

Feature-based methods track features well in slow-moving scenes, but fail during rapid motion sequences and have difficulties with occlusions. On contrary, intensity-based methods are sensitive to illumination changes and noise. Optimally, the two methods are combined, as for example, the parameters computed for feature-based approach may be used to initialize the gradient-descent procedures used for least-square minimization of the intensity-based approach.

3.2 Audio features for segmentation

When considering content segmentation it is usual to try to extract meaningful information primarily from the video signal, but combinations with audio analysis are the natural extension to it. The awareness and willingness to apply the orthogonal modality of audio for content segmentation grows and, hence, audio cues are either applied to supplement or to complement visual cues in segmenting the audiovisual material. Even more, often coherence or dissimilarity of audio cues help to aggregate or separate visual shots, which would be difficult or even impossible when applying visual cues only. Film directors, for example, often connect visual dissimilar shots through coherent audio in the background to express the semantically connection. Furthermore, a specific audio signal, i.e. speech, contains high-level audio cues, i.e. spoken text extracted by *Automatic Speech Recognition* (ASR), which can be used for semantically segmentation by applying mature text analysis solutions. But also audio classification, often applied as pre-processing step for audio segmentation, provides valuable insights about segmentation boundaries. Especially, classifying temporal audio sequences into one of the main audio classes, i.e. *speech*, *music*, *silences*, *background* and *crowd noise*, helps to identify audio scene boundaries, because they often correlate with such a class transition. In this section we study audio topics relevant or related to audio-based segmentation, i.e. audio silences, audio classification and audio segmentation.

3.2.1 Audio silences

The discontinuity-like nature of a scene boundary mostly is achieved and augmented by using a temporally correlating audio silence at the boundary between two scenes. Several studies, therefore, concentrate on this topic. Speech / silence discrimination, for example, as studied extensively to improve ASRs, determine boundaries of words and, hence, sentences using *signal energy* / *zero-crossing* thresholds schemes, as described by Rabiner in [71] and Biatov in [72]. In the latter speech / non-speech discrimination is applied as pre-processing step to silence detection, where low-level features are extracted from zero-crossing intervals within overlapping 20ms frames, forming the input for a multivariate Gaussian classifier. A segmentation algorithm is then applied to smooth the results of the frame level classification reporting a silence detection rate of 93.4%.

Another method, as described by Pfeiffer in [73], is based on a classification process based on perceptual loudness measure, which is extracted directly from MPEG-1 layer 2 audio parameters. It aims to detect relative silences between dominant foreground

audio, enabling the detection of silences even during noisy background instances, which is difficult when applying conventional energy thresholding. The loudness feature is extracted from consecutive 10ms frames, which are classified using an adaptive threshold and a sliding window concept. The adaptive part sets the threshold to a certain percentage of the maximal loudness in the window at a given time. Subsequently silent frames are clustered into longer silence intervals with minimum duration (mind) and maximum tolerated interruption (maxi). We restrict our overview to these examples.

3.2.2 Audio classification

The segmentation by silence can be preceded, as already mentioned, by audio classification as pre-processing step. Several methods are available to categorize audio instances into one of the five general audio classes, i.e. *speech*, *music*, *silences*, *background* and *crowd noise*, and then further into e.g. music related sub-classes. Most of the known classification methods use as pre-processing an audio feature extraction unit and, here after, an audio classification unit. McKinney provides in [74] an extensive overview of distinct audio features for feature extraction with regard to their feature strength for classification. The resulting selected features applied in [74] within the feature extraction unit are: (a) low-level signal properties, (b) *mel-frequency spectral coefficients* (MFCC), (c) psychoacoustic features including roughness, loudness and sharpness, and (d) an auditory model representation of temporal envelope fluctuations. The classification unit used these features to classify the audio signal into five general groups, i.e. *speech*, *classical music*, *popular music, background noise* and *crowd noise*. The four individual feature extraction stages are evaluated using the same classification stage, i.e. Gaussian-based quadratic discriminate analysis. The classification performance is measured in terms of probability, i.e. standard error. The detection rates for the general classes reach about 93.2%. The papers' conclusion is that temporal modulation in combination with audio perception features performs most effective.

Kim publishes another comparison in [75], where he compares the strength of mel-frequency spectral coefficients and MPEG-7 *Audio Spectrum Projection* (ASP) for different classification and segmentation tasks. The conclusion is that MFCC outperforms the ASP features in respect to performance and computational complexity.

Pfeiffer presents in [76] another classification framework applying low-level and psycho-acoustical features, such as volume, frequency, pitch, psycho-acoustical onset / offset, frequency transition maps, fundamental frequency and beat. 10ms audio frames are used as input for the feature extraction unit. Loudness is used to identify silences, as described by Li in [77], and loudness with pitch is applied to differentiate between

speech and music. Finally, characteristic overtone and rhythmic patterns are used to distinguish between speech and environmental noise.

In [78] Li describes a framework for real-time TV broadcast content segmentation and classification applying the audio signal. The incorporated features include signal energy, average zero-crossing rate and fundamental frequency spectral peak track. For the segmentation and classification a heuristic rule based procedure is applied based on morphological and statistical analysis. The classification results in the general groups, i.e. *speech*, *music*, *song, speech with music, environmental noise* and *silence*. The precision rates reported span from 84% for songs to 94.5% for music and the achieved recall rates are 89.6% for speech with music to 100% for silences.

Audio classifiers are of high relevance for various consumer applications and, therefore, many methods have been developed recently. Furthermore, classifiers reach reasonable recall and precision results. Hence, we consider applying audio classification as pre-processing for audio-based segmentation augmenting our video-based segmentation.

3.2.3 Audio segmentation

We briefly summarized some audio classification methods, which often serve as pre-processing step for audio-based content segmentation, as presented for example by Nitanda in [79]. The latter does not rely on the usual prior mid-level categorization and, hence, can be useful for e.g. semantic segmentation of more complex audio scenes representing multiple general classes at the same time. This is an approach, which is comparable with efforts to elaborate *Audio Scene Analysis* (ASA), which has the aim to describe the way, how the human auditory system perceives complex sound environments, comprising multiple sound sources varying independently from each other. The results are currently incorporated into MPEG-4, which provides tools for semantic and symbolic description of audio.

Another method, described by Sundaram in [80], defines audio scenes as semantically consistent segments of audio characterized by a few dominant sources of sound. The method uses several features to characterize dominant sounds, i.e. cepstral flux, multi-channel cochlear decomposition and cepstral vectors. Here after, it determines the dominant source analyzing the sequences of feature vectors with regards to periodicity, envelope and randomness. Then a causal listener model, which mimics human's perception at multiple time scales, applies several parameters to audio scene changes. The parameters used are memory (length of required buffer) and attention span (subspace of the buffer). Audio scenes are detected based on the correlation between the data in the attention span and the past data in the memory. The audio scene

detection rates achieved on a specific type of content, i.e. science fiction movie, result in a reported accuracy of 97% and false alarm probability of 10%.

Alternatively, Foote describes in [81] an audio segmentation method applying the audio similarity between past and future sliding windows. The similarity is expressed as feature vector distance, which corresponds to the power spectrum of the signal in the individual windows, derived after tampering the analysis window with a Hamming window and transforming it with an FFT. The logarithm of the magnitude of the FFT results in an estimate for the power spectrum of the signal in the individual sliding windows. Performing local thresholding the lowest similarity instances are identified and indexed as audio segment boundaries.

In [82] Cettolo compares three different *Bayesian Information Criterion* (BIC) based audio segmentation methods. Here a parameter is computed, which is based on the value of the covariance of the audio signal inside a sliding window of variable size. One method uses the sum of the input vectors and the sum of the squares of the input vectors for the BIC computation. The second exploits the encoding of the input signal with cumulative statistics for the efficient estimation of the covariance matrices. The third, finally, encodes the input stream with the cumulative pair of sums of the first method. Using performance and computational complexity as criteria, the conclusion is that the third method outperformed the other two.

The analysis of the state-of-the-art work unveiled that many methods for classification and segmentation, e.g. [71]-[76]/[79]/[81]/[82], have been elaborated for pure audio content, i.e. music and audio broadcast. Nevertheless, the results published for audio-based segmentation of visual content, e.g. [77]/[78]/[80], are promising. Hence, we consider fusing audio classification, as proposed by McKinney [74], and audio segmentation with visual cues augmenting our audiovisual segmentation.

3.3 Video high-level features for segmentation

The focus of this work is semantic audiovisual content segmentation enabling meaningful structuring of the audiovisual content into chapters comparable with the chaptering of commercial DVDs. Similar to the audio domain, visual classification can be applied as pre-processing step for semantic audiovisual segmentation. In this section we summarize several relevant methods for high-level video classification and segmentation.

The abstract semantic nature of semantic chaptering results in a certain subjectivity of the task. On the contrary, the definition of a *shot boundary* (SB), for example, is quite objective and does not vary when done manually by various individuals. Unfortunately, this is not the case for audiovisual *scene boundaries* (ScB). Indeed the procedure to choose these boundaries requests from the human annotator a priori knowledge at an abstract level, e.g. which shots are semantically related and should therefore be clustered together into a chapter, i.e. segment. Hence, some possibly objective rules have to be defined first to aggregate consecutive shots into semantic meaningful units, that is scenes or chapters, and the latter into entire content item entities, as shown in Figure 22. These objective rules are essential when trying to compare the results of various, otherwise incomparable because of their subjectivity, semantic audiovisual segmentation methods.

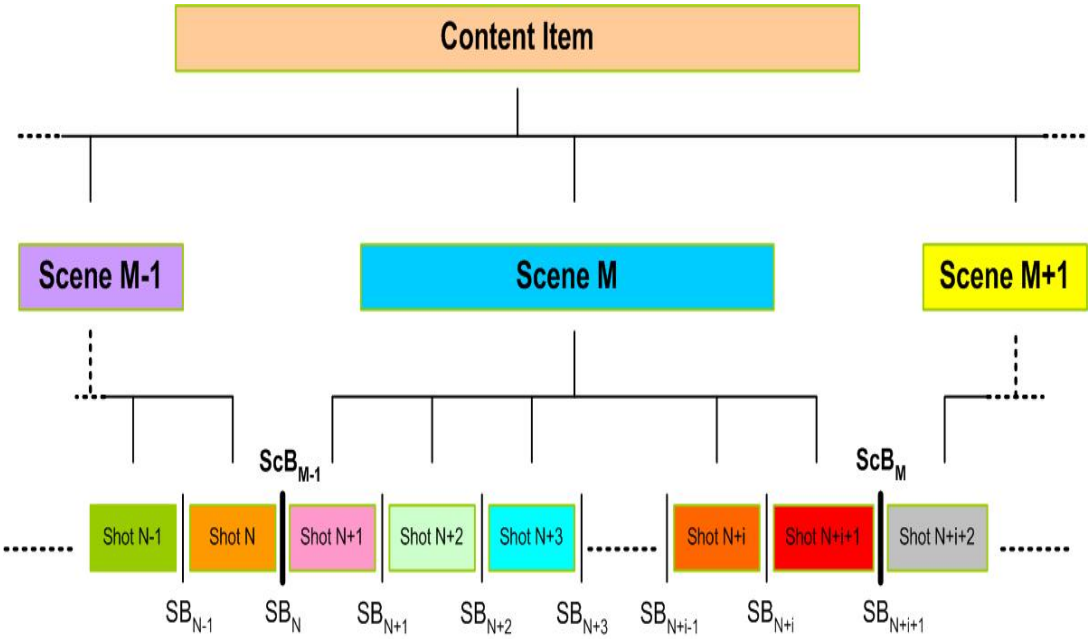


Figure 22. Scene Boundary Segmentation – schematically.

From Bordwell we know [16] that scenes represent a natural progression of a content item and they form a part of a story, comparable to chapters of a book. Various research teams have been busy to elaborate a definition of scenes, which were also referenced as *video paragraphs*, *story segments* or *logical story units* (LSU), for example by Hanjalic in [83] and Vendrig in [84]. Boggs summarized some of the widely applied scene definitions in [85], i.e.

- Definition 1: A scene is usually composed of a small number of interrelated shots that are unified by location of a dramatic incident, as published by Beaver in [15].
- Definition 2: A scene is a collection of in time continuous shots that have a single, consistent, underlying semantic.
- Definition 3: A scene is a continuous segment of data having coherent audio and video characteristics.

Some authors consider the notion of a *Hyper-scene* [86]. The latter means a union of content similar contiguous sets of shots. Hyper-scenes are of interest for a non-linear browsing in a content item. Nevertheless, in the framework of our research, we will focus on a more classical definition, supposing signal continuity. Hence, definition 2 comes closest to our definition we gave in section 2.1.3, which we applied for manual annotation of the audiovisual corpus, i.e.

- a scene consists of one or more shots conveying one single, consistent underlying semantic or narrative element; and
- a scene may incorporate one or more interleaved narrative events, i.e. cross-cuttings, or dialogues, i.e. shot reverse shots. Scene boundaries may not appear inside a parallel shot sequence;

But seen today's technical state, the identification of location consistency and/or consistent semantics exceeds the capabilities of current analysis and computer vision systems (definition 1 and 2). Hence, also our chosen definition for manual annotation is similar to definition 2 we expect that the technical realization will be located between definition 2 and 3. This implies that a scene change probably will correlate with a measurable significant change of the audio and/or video signal. An alternative for the latter is to use audio-video mid-level features instead, which we will consider in this work.

3.3.1 Video genre classification

The specific nature of our chosen audiovisual content, i.e. TV broadcast with embedded commercial blocks disrupting the coherence of the content of interest, forces us to consider separating the task of video classification and content segmentation. We do

not want to burden the segmentation task with the problem of non-content related inserts, which can be seen as 'content noise'. Hence, to accomplish our main task of AV content segmentation we have to develop a specific genre classifier, i.e. commercial block detector, which we apply for content filtering. The latter has to index non-content related inserts to exclude them automatically from subsequent processing steps. Because of their commercial value commercial block detectors experience a lot of interest, which results in many methods published. Hence, we will give a brief overview of them in this section.

Commercial block detection

Commercial blocks, i.e. a block of individual advertisements, have various distinctive detectable attributes, mainly because commercials have to convey an attractive appealing message in a very short period of time. Commercials, therefore, consist of short individual clips, each containing short shots, corresponding to a high shot boundary (cut) frequency, as exploited by Blum in [87], and rich activity. Furthermore, black frames separate these individual commercial clips in most broadcast streams. Blum presents in an early patent [87] a method based on the combination of black frame detection and activity classification, wherein the latter is realized through the change of luminance levels between two different clusters of frames. Unfortunately, black frames appear quite often at e.g. dissolves and luminance activity at motion-rich activity content sequences, both leading to false detections. Iggulden extended, in his patent [88], the previous concept by including the time-wise distance between consecutive black-frames increasing herewith the robustness of the method. At the same time, Lienhart published in [89] a method in combining black frame detection, with shot boundary (cut) detection and action detection. The latter consists of the combination of macroblock-based motion vector length and *edge change ratio* ECR with

$$ECR(N) = \max\left(\frac{Edg_N^{in}}{NP_N}, \frac{Edg_{N-1}^{out}}{NP_{N-1}}\right) \quad \text{(3-25)},$$

wherein N defines frame instance, NP_N number of pixels of the frame, Edg_N^{in} and Edg_N^{out} the amount of entering and exiting edge pixels. The robustness of commercial block detection can be improved recognizing time-wise repetition of commercial clips. Signatures of known or previously detected commercial clips are matched with new commercial clips by means of e.g. *colour coherence vectors* (CCV) as described by Lienhart in [89]. Hereafter, the scientific community started to combine audio and video cues, i.e. the co-occurrence of e.g. simple audio silences and black frames as

presented by Marlow in [90]. But also extensions with legal-based rules, i.e. non-presents of logos, are presented as e.g. by Albiol in [91].
Iggulden's and Lienhart's methods attracted some interest in the consumer market, because of their simplicity, but failed due to their robustness. We consider to extent their methods with audio cues in this work.

Nevertheless, next to genre classification many attempts were published for semantic in content classification, such as highlights for summaries or moods of the content, as e.g. published by Hanjalic in [92]. In the latter Hanjalic described content description at the affective level. The latter was visualized within a 2D emotion space characterized by arousal, i.e. the intensity of emotion load changes along the video e.g. highlights, and valence, i.e. the expected changes of the moods e.g. negative segments. His analysis using soccer videos were summarized in his papers [92] and [93].

3.3.2 Visual and Audio-Visual based segmentation

So far we have summarized in this chapter available methods for segmenting the content into its elementary units, i.e. shots by means of shot boundary detectors, the identification of non-content related inserts, i.e. commercial blocks, and audio-based classification and segmentation methods. Our main aim of this work is audiovisual content segmentation. Hence, in this section we investigate in total eight representative methods for video- and audiovisual segmentation, i.e. scene boundary detection, in more detail and evaluate them based on two criteria, i.e. their

- performance, i.e. their reported robustness for the aim they are developed for, and
- computational complexity, i.e. the processing power required for the method.

Unfortunately, the various methods were benchmarked against various corpora and, therefore, we categorize and rate the individual methods only coarsely by using a small scale of grades, i.e. *excellent*, *good*, *average* and *poor*.
In general the eight methods can be clustered into two groups, i.e. those applying only visual features and those who combine audio and video cues, with one exception, the method published by Wang in [94], which augments visual cues with cues derived from cinematographic rules to detect scene transitions. The majority of the methods apply shot boundaries as input and, hence, their robustness is very much dependent on the robustness of the foregoing shot boundary detector. Unfortunately, all works applied own shot boundary detectors, instead of shot boundary ground truth and, hence, the scene boundary detection results of all methods are 'polluted' by the detection rates of the shot boundary detectors.

The first method, a framework published by Kang in [95], is based on three hierarchical steps, i.e. *initial segmentation*, *refinement* and *adjustment*. The entire method is based on a continuous coherence-computing model. Shot boundaries are detected by means of colour histograms and regions', i.e. optical, flow followed by a camera motion based key frame selection. The latter are stored in a first-in-fist-out (FIFO) buffer with dynamic memory size, representing each shot as a symbol. The symbols of the entire content item are shifted through the buffer and coherent shots within a specified attention span are clustered together. A feature called shot recall specifies the coherence, where the one minus dissimilarity value between two shots A and B, i.e. *(1-dissim(A,B))*, is multiplied shot length of both shots, i.e. shot length $Shot_A$ and $Shot_B$, normalized by the memory buffer size, resulting in

$$\mathrm{Re}\ call\ (A,B) = (1 - dissim\ (A,B)) * Shot_A * Shot_B * (1 - \Delta n / N_m) * (1 - \Delta t / T_m) \quad \text{(3-26)}.$$

The normalization requires the parameter N_m (number of total shots in the memory buffer), Δn (the number of intermediate shots between shot A and shot B), T_m (memory buffer size) and Δt (time difference between shot A and shot B). The resulting shot recall is, here after, normalized by the maximum shot recall value $Co_{max}(S_a)$, which results in the coherence value $Co(S_a)$ with

$$Co(S_a) = \sum (\mathrm{Re}\ call\ (A,B)) / Co_{\max}(S_a) \quad \text{(3-27)}.$$

Another method, presented by Rasheed in [96], identifies first shot boundaries exclusively by means of a colour histogram. Each shot is then represented by a set of selected key frames, using the middle frame of a shot as initial frame and key frames, which surpass a dissimilarity level, are added to the initial one. In addition to the key frames also shot motion and shot length are extracted, as sketched in [96]. Rasheed's scene detection method is based on a two-pass solution. In the first pass a colour similarity measure, i.e. *Backward Shot Coherence* (BSC), is applied to quantify the shot matching between successive shots. Dips of the BSC graph are, here after, indexed as *Potential Scene Boundaries* (PSB).

High over-segmentation, e.g. caused by non-repetitiveness in for example action-loaded sequences, inspired Rasheed to include a second pass applying a *Scene Dynamic* (SD) measure to improve the robustness of the method. The SD is a function of shot length and shot motion used to merge scenes of the first pass and, here with, to reduce the over-segmentation. For each scene *i* the correlated *SDi* is computed by normalizing the shot motion SMC_j of the j-th shot in the scene by the shot length L_j of the corresponding shot, resulting in

$$SD_i = \frac{\sum_{j \in Scene_i} SMC_j}{\sum_{j \in Scene_i} L_j} \qquad (3\text{-}28).$$

The large values of SMC_j and smaller values of L_j in dynamic scenes cause SD to be large and, hence, increase the tendency to merge scenes.

At the same time Rasheed published in [97] a scene boundary detector, which combines an indoor- and outdoor-scene oriented clustering method applying visual contents- and motion contents similarity, respectively. During indoor shots multiple cameras capture similar background with similar foreground, which can be identified with visual similarity, i.e. colour similarity. Outdoor sequences like action- or travel-scenes, on the other hand, exhibit correlating motion. First he calculates the colour similarity *ColSim(x,y)* between two frames, i.e. *x* and *y*, by applying the minimum HSV histogram distance of the two frame histograms, i.e. H_x and H_y, with

$$ColSim(x, y) = \sum_{k \in bins} \min(H_x(k), H_y(k)) \qquad (3\text{-}29).$$

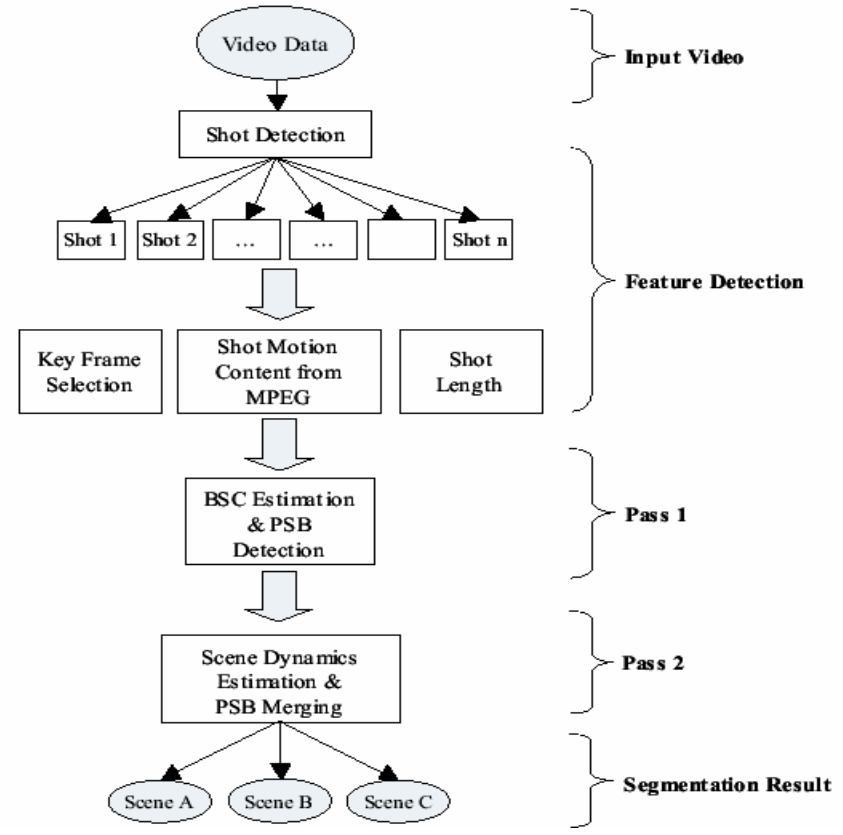


Figure 23. Flow chart of Scene Boundary Detection method of [96].

Here with he computes the motion contents similarity Mot_z by normalizing the overall dissimilarity with the length of shot $z=b-a$. Here b is the end frame index and a the start frame index, which results in

$$Mot_z = \frac{1}{b-a}\sum_{f=a}^{b-1}(1 - ColSim(f, f+1)) \quad (3\text{-}30).$$

The resulting inter-shot similarity *ShotSim(i,j)* between two shots, i.e. $shot_i$ and $shot_j$, is the sum of visual similarity *VisSim(i,j)* and motion content similarity *MotSim(i,j)* with

$$ShotSim(i, j) = VisSim(i, j) + MotSim(i, j) \quad (3\text{-}31).$$

VisSim(i,j) is the maximum colour similarity between all key frames of the two shots with

$$VisSim(i, j) = \max_{p \in K_i, q \in K_j}(ColSim(p, q)) \quad (3\text{-}32),$$

and *MotSim(i,j)* is derived by computing the motion content similarity between two shots:

$$MotSim(i, j) = \frac{2 * \min(Mot_i, Mot_j)}{Mot_i + Mot_j} \quad (3\text{-}33),$$

Here after, shot similarity *ShotSim(i,j)* is multiplied with a temporal-based exponentially decreasing weight function *w(i,j)* resulting in the shot similarity measure *W(i,j),*

$$W(i, j) = w(i, j) \cdot ShotSim(i, j) \quad (3\text{-}34),$$

which reflects the likelihood of two shots belonging together to the same scene.

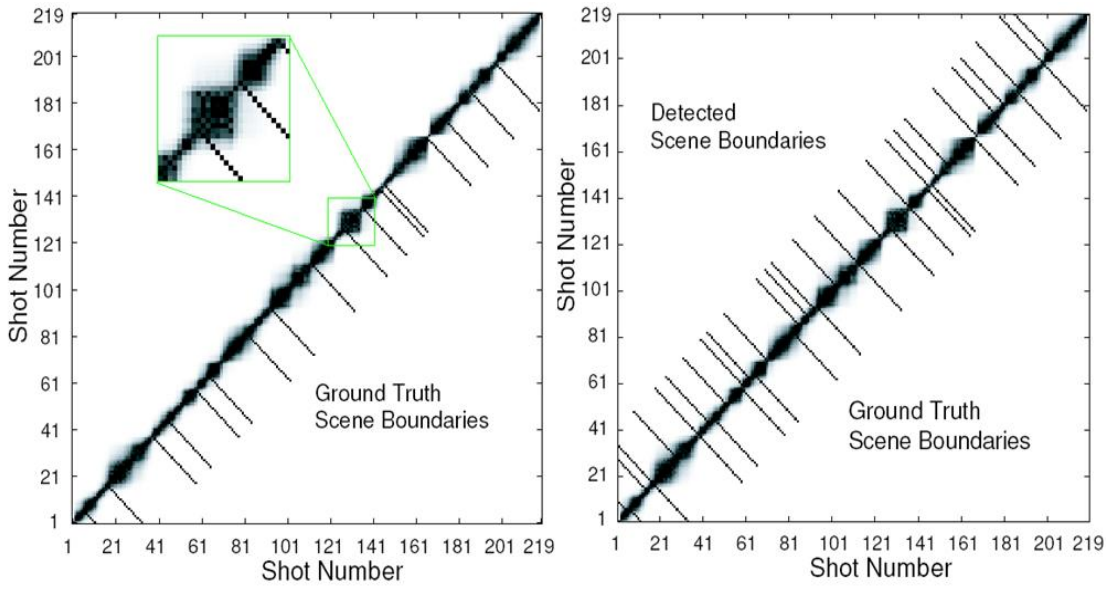


Figure 24. Rasheed's shot similarity graph as described in [97].

The weight function, intended to decrease the likelihood of temporal distant shots, is calculated by using the temporal distance of the middle frames of two shots, i.e. m_i and m_j, the decreasing rate *d* and the standard deviation of the shot duration in the entire video σ resulting in

$$w(i,j) = e^{-\frac{1}{d}\left|\frac{m_i - m_j}{\sigma}\right|^2} \qquad \text{(3-35).}$$

The results of the shot similarity measure *W(i,j)* for a 36 minutes movie sequence are shown in Figure 24 together with scene boundary ground truth. This method is based only on low-level features such as histograms and it therefore not convincing. Specifically, the authors consider a very limited definition of scenes based on colour similarity / dissimilarity. We will see that indeed this method exhibits only low robustness.

But visual cues can be enhanced by audio cues, which Huang presents in [98], where Huang augments colour- and motion content based discontinuity scene detection, similar to the previous methods described, with audio break detection. For the latter he first divides the audio signal into non-overlapping one second clips. For each clip a number of feature values are extracted, i.e. *non-silence ratio*, *volume standard deviation*, *volume dynamic range*, *4 Hz modulation energy*, *pitch period deviation*, *smooth pitch ration*, *non-pitch ratio*, *frequency centroid*, *frequency bandwidth* and *energy ration in 3 sub-bands*. The feature values form together the feature vector $\vec{f}(0)$ of the current clip, $\vec{f}(i)$ for the i-th clip. With the standard deviations σ^2_{A-} *and* $\sigma^2_{A\_}$ of the features of the predeceasing N-clips and a small constant *c*, to avoid division by zero, the Euclidian distance ||.|| between two clips is computed resulting in the audio dissimilarity D_A with

$$D_A = \frac{\left\| \frac{1}{N}\sum_{i=-N}^{-1} \vec{f}(i) - \frac{1}{N}\sum_{i=0}^{N-1} \vec{f}(i) \right\|^2}{\sqrt{(\sigma^2_{A-} + c)(\sigma^2_{A+} + c)}} \qquad \text{(3-36).}$$

The local maxima of D_A are then indexed as audio breaks if they exceed a defined threshold. The visual-dissimilarity-based break detection, i.e. with colour histograms and motion content, is derived in a similar way as described in [97]. Here after, for each audio break the close neighbourhood is checked for present visual breaks and if case of temporal correlation the instance is indexed as scene boundary.

A method, not only applying audio segmentation but also audio classification, is presented by Zhu in [99], where he categorizes the audio signal into four audio groups, i.e. silence, speech, music environmental sound, before segmentation is applied. For the case of speaker also speaker change detection is applied. Firstly, as shown in the flow graph in Figure 25, a differentiation is done between silences, which have the attribute of low *short-time average energy* and *short-time average zero-crossing rate*, and non-silences. The remaining non-silences instances are check for environmental sounds, i.e. applause, whistle and other noise, which have the attribute of containing high frequencies, in the contrary to music and speech, which remain in the low frequency range.

Hence, he applies the *frequency centroid* to identify environmental sound instances. In the remaining instances Zhu discriminates between speech and music by using *Low Energy Ratio* (LER), which represents the ratio between the number of frames with an energy level below a certain threshold and the number of frames in the entire audio clip.

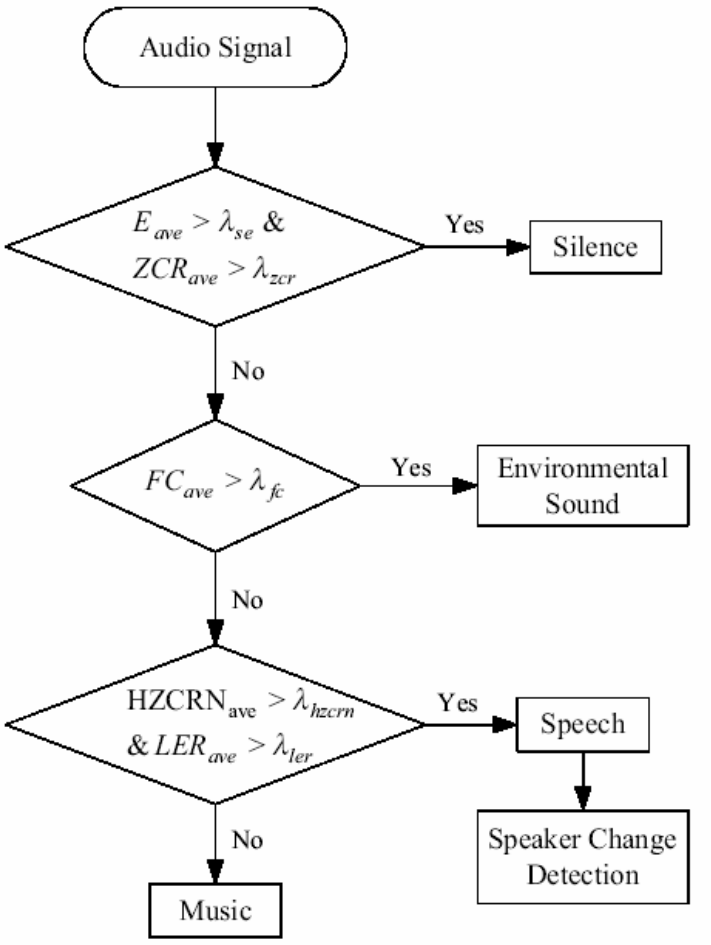


Figure 25: Zhu's audio classification flow chart block diagram from [99].

Zhu calculates the LER for the i-th clip, i.e. the sequence from the initial frame t_1 and to the terminal frame t_2, by thesholding and normalizing the energy function, where f_t^L=1 if the energy of the audio frame is below the threshold and otherwise f_t^L=0, which results in

$$LER_i = \frac{\sum_{t=t_1}^{t_2} f_t^L}{t_2 - t_1} \qquad (3\text{-}37).$$

LER reaches higher values in speech than in music. Furthermore, the speech signal exhibits frequent evenly distributed peaks and, hence, he applies, in addition to the LER, the *high zero-crossing rate number* (HZCRN) as speech-music discriminator. Audio class transition instances are indexed as audio breaks and if they temporally correlate with visual breaks, i.e. cut or gradual transitions, then they are indexed as scene boundaries.

Another audio classification augmented scene boundary detection method is described by Rho in [100], where he first applies audio classification prior audio segmentation to detect audio breaks, similar to Zhu's method of [99].
Here after, temporal correlating instances of the latter with visual breaks. i.e. shot boundaries, are indexed as scene boundaries. The method applies *short time average energy function* to derive loudness, which is used to discriminate between voice and noise.

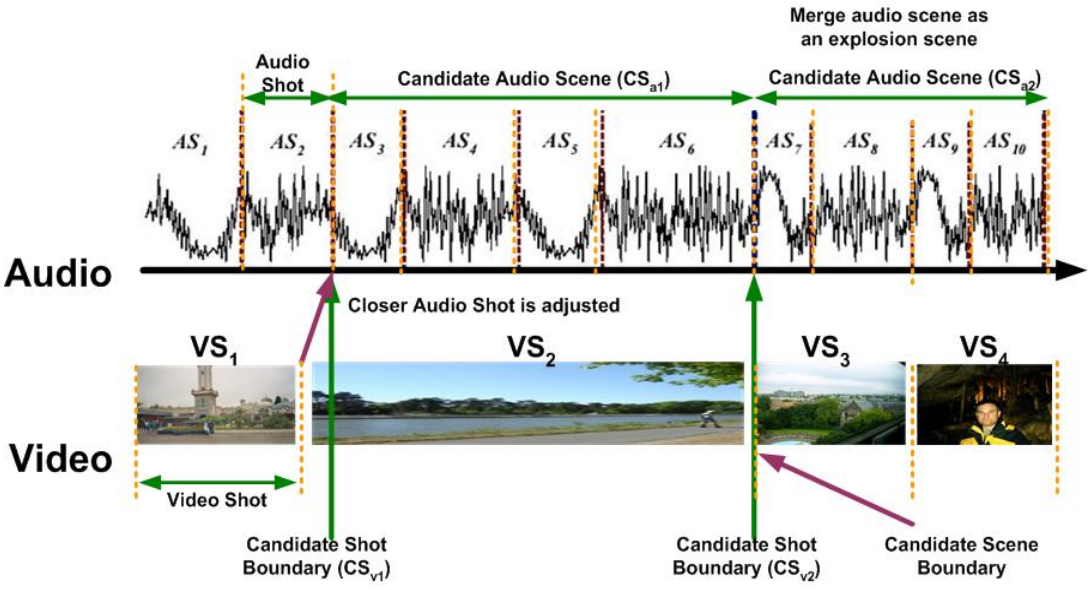


Figure 26. Rho's scene boundary detection method described in [100][1].

The distinction between voiced and un-voiced speech signal *x(n)* is done by using the *average zero crossing rate* (ZCR) with the sign of *x(n)* normalized across the clip length *N* with

$$ZCR = \frac{\sum_{n=1}^{N} \left| \operatorname{sgn} x(n) - \operatorname{sgn} x(n-1) \right|}{2N} \qquad \text{(3-38).}$$

To distinguish between the high frequency containing music and speech the *energy distribution* is applied in combination with the bandwidth, because music exhibits is normally scattered across a broader frequency range than speech. The bandwidth is simply measured from the lowest to the highest frequency of the non-zero spectrum components. The last discriminating audio feature applied is harmonicity, because music contains usually harmonic sounds, whereas speech is a mixture of harmonic and non-harmonic components and environmental sound only contains non-harmonics. The audio class transition instances are indexed as audio breaks between Candidate audio scenes CS_{ai} and in the case of temporal correlation with Candidate shot boundaries CS_{vi} they are indexed as scene boundaries, as shown in Figure 26. Hence, in this method an in deep attempt is made to classify audio into usual classes and to use these data as an indication for scene boundaries.

In [101] Chen describes as well an method syndicating audio and video to identify scene boundaries, but in this case the shot boundary method is enhance with object tracking and for the audio break detection three parallel audio detectors are applied. This method claims the best performance results. For the visual part segments are identified within the frames and tracked by means of a Euclidian distance between the centroids of the segments in adjacent frames, as shown in Figure 27.

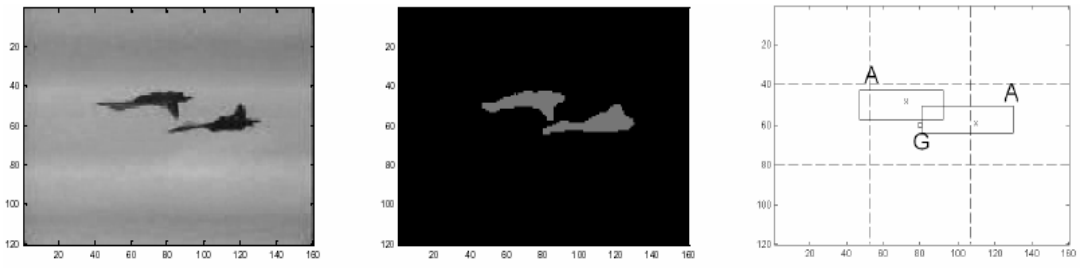


Figure 27: (a) Reference frame; (b) segmentation mask map of reference (a); (c) bounding boxes and centroids for the objects/segments in (b), as shown in [101].

Object tracking discontinuities are augmenting the robustness of the shot boundary based visual break detector. For the audio breaks nine different audio features are clustered into three different audio groups, i.e. *volume group*, *power group*, *spectrum group*. For each group the Euclidian distance is measured between two successive clips and if at least one of the three measures exceeds its corresponding threshold the instance is indexed as audio break. Here after, the usual temporal correlation between audio and visual break is applied to index scene boundaries.

The last method, presented by Wang in [94] is the only one incorporating cinematographic production rules, i.e. film grammar, into the scene boundary detection approach. In his method he exploits the knowledge of the 180^0 rule, i.e. that the cameras' viewpoints should be always at one side of a virtual line. Actor, for example, standing in the right half in the scene will maintain his right position as well in close-ups, as shown in Figure 28 (left). But also montage rules, i.e. concentration and enlargement rules are used. The *concentration rule* defines that a sequence start with a long distance shot and progressively changes by zooming into a close-up shot by decreasing the camera's focal distance. The reverse of it is the *enlargement rule*, which specifies the approach how to switch from a close-up of the object to a total view by progressively zooming out, as shown in Figure 28 (top right).

Figure 28: Cinematographic rules from [94][1].

This sequence is indicative for an upcoming scene boundary. Finally, Wang includes as well the parallel rule, which specifies how to compose scenes involving multiple themes, i.e. shots of two or more themes are shown in an alternate fashion within the same linear sequence. An example for the latter is a dialogue between two persons, as sketched in Figure 28 (bottom right), or parallel activities shown in an alternating way. Shots belonging to the same theme tend to have strong visual similarities and the same focal distance, which Wang exploits in his method.

Performance

The reported scene boundary detection performance results, i.e. recall and precision, of all methods are summarized in Table 1. Unfortunately, each method was benchmarked against its own corpus and various scene boundary ground truth rules were used for the manual annotation. Furthermore, the results of preceding analysis steps fuzzify the performance results, e.g. instead of applying the shot boundary ground truth the output of the real shot boundary detector was incorporated. Nevertheless, the overview shows that audio augmented methods, i.e. [98] to [101], outperform the visual-only methods, i.e. [95] to [97]. The method exploiting cinematographic rules [94], i.e. film grammar, performed in between the two groups, but offers the broadest potential.

Complexity

Also the computation complexity is not of high relevance for our work, we have coarsely labelled the methods based on their technical requirements to extract the applied features. We assume compressed content as input, i.e. MPEG-2, and exploit as much as possible features available in the compressed domain. The three video-based methods (i.e. [95] to [97]) apply colour histogram and motion information. These data are easily extractable from MPEG-2. But in addition all three methods require a memory buffer. The audio augmented methods, i.e. [98] to [101], all apply easy extractable cheap audio features in combination with cheap video features, except the method of [101], which requires complex object tracking. The cinematographic method is the most complex one, because it requires object analysis. The coarse classification of the eight methods is summarized in Table 2.

Table 1: Performance evaluation of the methods presented in this section.

Method	Tested on	Recall	Precision	Rating
[95]	Movie (20 min.)	82.0%	85.0%	good
	Drama A (20 min.)	88.0%	85.0%	good
	Drama B (20 min.)	87.0%	87.0%	good
[96]	Terminator 2 (55 min.)	86.1%	81.6%	good
	Golden Eye (60 min.)	88.0%	62.9%	average
	Gone In 60 sec. (58 min.)	84.6%	76.7%	good
	Top Gun (50 min.)	88.5%	76.7%	good
	A Beautiful Mind (36 min.)	88.2%	71.4%	average
	Seinfeld (21 min.)	86.4%	70.0%	average
[97]	A Beautiful Mind (36 min.)	83.3%	53.6%	poor
	Terminator 2 (55 min.)	88.9%	45.7%	poor
[98]	Basketball (155 sec.)	100.0%	100.0%	excellent
	Football (223 sec.)	100.0%	100.0%	excellent
	News + Commercials (218 sec.)	100.0%	85.7%	excellent
	News (158 sec.)	100.0%	0.0%	worse
[99]	News 1	93.4%	97.5%	excellent
	News 2	94.5%	98.2%	excellent
	Commercials	90.8%	92.6%	excellent
	Story	88.3%	91.7%	excellent
[100]	TV Commercials + News	84.0%	90.0%	good
	Movies	86.0%	97.0%	excellent
[101]	V1	93.0%	93.0%	excellent
	V2	92.0%	92.0%	excellent
	V3	78.0%	100.0%	good
	V4	91.0%	91.0%	excellent
	V5	90.0%	86.0%	good
[94]	Movie + Documentary	86.2%	82.7%	good

Table 2: Overvie of computational complexity evaluation.

Method	Visual Feature Calculations	Audio Feature Calculations	Temporal Processing (Buffering)	Special Processing	Rating
[95]	2	0	YES	0	good
[96]	2	0	YES	0	good
[97]	2	0	YES	0	good
[98]*	2	1	NO	0	excellent
[99]	2	4	NO	0	excellent
[100]	3	5	NO	0	excellent
[101]	1	9	NO	1	average
[94]	X	X	YES	X	worse

Furthermore, many attempts were published approaching the semantic gap from two sides, (a) bottom-up, i.e. combing audio and video features to extract higher level semantics such scene data, and (b) top-down, i.e. selecting semantic events and extracting feature behaviour. One of the two side approach concepts was published by Leonardi in [102], applying a finite-state machine using MPEG-2 motion data for the top-down approach and a *Hidden Markov Model* HMM for the bottom-up approach. The conclusions of Leonarid's study were that knowledge of the content, i.e. top-down, was necessary achieving semantic characterization reliably.

Conclusions

We have seen that many methods were elaborated to identify scene boundaries, but because of the task's subjective nature and the herewith-related different requirements for different content can be seen as a reason here for. Many methods apply visual features only, but are outperformed by those augmented with audio-based features, as may expected. Furthermore, the audio augmented methods, replacing partly costly video features by cheap audio features also complexity wise outperform the video only methods. Nevertheless, we think that the highest potential has the approach based on cinematographic rules, i.e. film grammar, because it approaches the task by exploiting semantic understanding of the content, as proposed by Wang in [94].

3.4 Conclusions

In this chapter we have summarized work dealing with a selected group of topics relevant for our work. The basis for audiovisual content analysis is almost always to segment the content into its atomic units, i.e. shots in our case. Many compressed- and uncompressed-domain cut detectors are available, e.g. [39] to [53], as presented in section 3.1.1, but achieved performances are still below our requirements. Hence, we consider researching various new cut detection methods and benchmark them against one selected method, i.e. Rough indexing cut detection [56], which participated in TRCVid [42] and, hence, provides us with a more objective reference point. For gradual transition detection fewer methods are available and for our work we consider using Naci's method [59]. The analysis of audio-based methods to segment audiovisual content, summarized in section 3.2, unveiled that promising results can be achieved, as presented in [77],[78] and [80]. Even more, many methods use an audio classifier as pre-processing step to enhance the results, which we consider as well by applying McKinney's method [74] for classification. The specific nature of our content, i.e. TV broadcast, requires considering to develop a genre specific video genre detector, i.e. commercial block detector, to eliminate non-content related inserts. The methods of Iggulden [88] and Lienhart [89], described in section 3.3.1, are attractive due to their simplicity but under perform what concerns robustness. We consider elaborating their methods in the context of this work. In the final section of this chapter, i.e. section 3.3.2, the analysis of video- and audio-based segmentation methods showed that audio-augmented method outperformed the video-only-based ones performance and complexity wise. Nevertheless, we believe that the cinematographic rule based method presented by Wang [94], offers the highest potential because it exploits semantic understanding of the content. Hence, we aim to follow this concept.

CHAPTER 4

4 Audiovisual content analysis methods for semantic segmentation

In this chapter we present our contribution to the research and development of advanced methods for audiovisual content analysis.

Firstly, we present in section 4.1 a selected group of video low-level (section 4.1.1) and mid-level features (4.1.2), which we aim to apply for general and application-oriented audiovisual content analysis tasks. In section 4.1.2 we present, namely, our contribution in the domain of audiovisual shot segmentation, i.e. shot boundary detection.

Here after, we describe in section 4.2 task oriented audio low- and mid-level features, which we consider applying amongst others for a specific genre classifier.

In section 4.4 we present our contribution developing a dedicated genre specific content filter to identify non-content related inserts, i.e. commercial blocks. This commercial block detector is applied as pre-processing step to audiovisual content segmentation.

Once filtered, the content can be analyzed from a production point of view. In section 4.5 we propose a method to check the content consistency exploiting knowledge of film grammar production rules.

Finally, in section 4.6, we develop a method for the detection of scene boundaries in audiovisual content embedded in a general content analysis framework.

4.1 Video low-level and mid-level feature

In this section we describe several of low-level and mid-level video analysis based features, which constitute the basis for our high-level segmentation and classification tasks. The partially constraint target environment forced us to research not only base-band, but also in some cases compressed content analysis solutions. In section 4.1.1 we describe in-depth a set of compressed- and base-band-domain low-level video features. In 4.1.2 we present several of our video shot boundary detectors and benchmark them against each other. Furthermore, we describe the methods we propose for a selected group of other mid-level features such as text and face detection.

4.1.1 Video low-level features

In this section we describe various compressed- and base-band-domain low-level video analysis solutions we developed to serve high-level application algorithms as input parameters. The specific MPEG-2 terms used in this section and specific compression parameter settings were described and summarized, respectively, in Annex 1.

Macroblock matching parameter in MPEG-2

The advantage of applying compressed video content, e.g. MPEG-2 material, for video feature extraction is based on the fact that during video compression, i.e. encoding, spatio-temporal information is extracted from video frames to enable efficient encoding. The same information can be re-applied as low-level features for video analysis and indexing. Let us consider a motion estimation algorithm used for MPEG-2 compression. As described in our MPEG-2 annex, i.e. Annex 1, frames are subdivided into *macroblocks* (MB). A *motion estimator* (ME) is applied to the macroblcok of the current frame searching for best corresponding macroblck in a reference frame, as described in detail in Annex 1. The ME seeks to minimize the *mean absolute difference* (MAD) criterion. The MAD measure is computed by

$$MAD(x, y, dx, dy) = \frac{1}{256} \sum_{i=0}^{15} \sum_{j=0}^{15} \left| I_N(x+i, y+j) - I_M((x+dx)+i, (y+dy)+j) \right| \quad (4\text{-}1),$$

where *MAD(x,y,dx,dy)* represents the MAD between a 16*16 array of pixels (pels) of intensities $I_n(x+i,y+j)$, at MB position *(x,y)* in the source frame *N*, and a corresponding 16*16 array of pixels of intensity $I_M(x+dx+i,y+dy+j)$, in reference frame *M*, with *dx* and *dy* representing the shift along the *x* and *y* coordinates, also called motion vector. The optimal, i.e. minimal, value of MAD allows quantifying the MB similarity between two consecutive frames. Here after, we calculate the normalized difference value

$MAD_{Norm}(N)$ per frame, here for frame *N*. For this we first compute the sum of all MADs across all slices excluding letterbox[2] and subtitle slices. Herein

- *MBPS* represents the number of *macroblocks per slice,*
- *SPF* is the number of *slices per frame,*
- *LB* is the number of slices containing the *letterbox*,
- *STS* is number of slices containing *subtitles*.

The sum is, here after, normalized by the total number of MBs, i.e. *(SPF-STS-2*LB)*MBPS*, multiplied by the maximal possible MAD value, i.e. 256, resulting in

$$MAD_{Norm}(N) = \frac{\sum_{i=LB+1}^{SPF-STS-LB-1} \sum_{j=1}^{MBPS} (MAD[N,i,j])}{\max(MAD) * (SPF - STS - 2 * LB) * MBPS} \qquad (4\text{-}2).$$

Frame complexity analysis in MPEG-2

For efficient variable MPEG-2 compression encoders normally compute internally another valuable measure, i.e. normalized complexity COM_{Norm}, to which we have access by using our own MPEG-2 encoder for this work. In fact COM_{Norm} is related to the product of motion and texture. To compute COM_{Norm}, which is valid for I-frames only, we apply two compressor internal parameters, i.e.

- *BG* the number of *bits generated* or required to represent the information within a segment, where a slice here is divided into three segment, and
- *QS* the *quantizer scale*, which is the quantization scale set per segment.

Next, we sum this product, i.e. BG*QS, across

- all specified segments, i.e between *fs / ls* representing the first and last segment inside a slice, and
- all specified slices, i.e. between *fi / li*, i.e. the first and last slice taken into consideration.

Finally we normalize the complexity by the area considered multiplied by the maximal value possible with BG*QS, which results in

$$COM_{Norm}(N, area(fi, li, fs, ls)) = \frac{\sum_{i=fi}^{li} \sum_{s=fs}^{lls} (BG[N,i,s] * QS[N,i,s])}{\max(BG * QS) * (li - fi) * (ls - fs)} \qquad (4\text{-}3),$$

Unfortunately, *BG* and *QS* are video compressor internal parameters and, hence encoder dependent. Nevertheless, for the context of our work this is a valuable parameter, which we intend exploiting.

[2] Letterbox represents in the 16:9 modus black bars on the boundaries (top, bottom) of a video frame.

Progressive - Interlace classification in MPEG-2

Furthermore, an MPEG-2 encoder is able to differentiate between progressive content, which is used e.g. to capture movies, and interlaced content, which is applied e.g. to record soaps and series. Hence, this mode parameter is also applicable to detect boundaries in video broadcast content. Furthermore, this enables the encoder to further increase the compression, because both formats have a specific repetitive pattern, as described in Annex 2. Hence, MPEG-2 allows switching between DCT field mode and DCT frame mode on a macroblock basis. The DCT field mode appears prominently with interlaced content, whereas the DCT frame mode with progressive material, as described in Annex 2. The DCT field / frame mode decision per macroblock is based on two Hadamard transforms, one calculating the correlation between pixels of consecutive rows (*'1st Hadamard* → frame mode) and the other between every second row *('2nd Hadamard* → field mode). If *'1st Hadamard' > '2nd Hadamard'* is true, then the macroblock is labelled with DCT field mode, otherwise DCT frame mode, but the condition of sufficient horizontal motion has to be fulfilled, due to the specific nature of the decision coefficients. In static sequences, for example where the motion condition is not fulfilled, the decision is always DCT frame mode independent of the material's original format. Based on experiments the following rule is defined empirically. If more then 90% of all macroblocks of a slice *i* in frame *N* have an absolute horizontal motion |*x-motion*|*>8* halpels, i.e. half pixels, then

$$\begin{aligned} |x-vector| > 8.halfpels &\Rightarrow SUM_{Prog/Inter}(N,i) = \sum_{j=1}^{MBPS}(MB_{DCTMode}[j]) \\ |x-vector| \le 8.halfpels &\Rightarrow SUM_{Prog/Inter}(N,i) = 0 \end{aligned} \qquad (4\text{-}4),$$

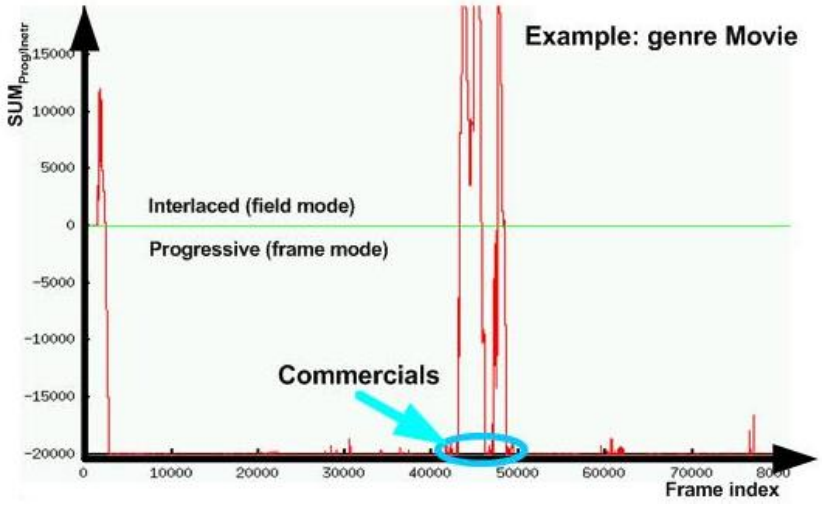


Figure 29. Interlaced-progressive classification for a movie with a commercial block.

with $MB_{DCTMode}=1$ for a DCT field macroblock and $MB_{DCTMode}=-1$ for a DCT frame macroblock. Subsequently, the slice values are added to a running sum $SUM_{Prog/Inter}(N)$ with

$$SUM_{Prog/Inter}(N) = SUM_{Prog/Inter}(N-1) + \sum_{i=1}^{SPF} SUM_{Prog/Inter}(N,i) \quad (4\text{-}5),$$

which is clipped at the empirically derived values ±20.000, as shown in Figure 29 for a progressive mode feature film with several interlaced/progressive mode commercial clips.

Black and monochrome frame detection in MPEG-2

Black and monochrome frames are features, which we categorize as low/mid-level features. Both frames are often delimiters in specific content sequences such as commercial blocks, as claimed by Blum [87], Iggulden [88] and Lienhardt [89]. Initial performance restrictions motivated us to research as well MPEG-2 4:2:0 sub-sampling format based solutions. Hence, additional parameters we look at are the normalized average luminance and chrominance values $Y_{DC\_AV\_Norm}(N)$, $U_{DC\_AV\_Norm}(N)$ and $V_{DC\_AV\_Norm}(N)$. They are individually calculated on each available intra frame (I-frame) with

$$Y_{DC\_AV\_Norm}(N) = \frac{1}{\max(Y_{DC}) * (SPF - 2 * LB) * MBPS * 4} \sum_{i=LB+1}^{SPF-LB-1} \sum_{j=1}^{MBPS} \sum_{b=0}^{3} Y_{DC}[N,i,j,b] \quad (4\text{-}6),$$

$$U_{DC\_AV\_Norm}(N) = \frac{1}{\max(U_{DC}) * (SPF - 2 * LB) * MBPS} \sum_{i=LB+1}^{SPF-LB-1} \sum_{j=1}^{MBPS} U_{DC}[N,i,j] \quad (4\text{-}7),$$

$$V_{DC\_AV\_Norm}(N) = \frac{1}{\max(V_{DC}) * (SPF - 2 * LB) * MBPS} \sum_{i=LB+1}^{SPF-LB-1} \sum_{j=1}^{MBPS} V_{DC}[N,i,j] \quad (4\text{-}8),$$

wherein Y_{DC}, U_{DC}, V_{DC} represent the normal and max(Y_{DC}), max(U_{DC}), max(V_{DC}) the maximal values of the DC luminance- and chrominance values, respectively, per block. *b* defines the number of blocks per macroblock, *MBFS* the macroblocks per slice, *SPF* the number of slices per frame and *LB* the number of letterbox slices. $Y_{DC}[N,10,5,3]$, for example, represented the 3rd Y_{DC} block, of the 5th macroblock in the 10th slice of frame *N*. It should be stated that the method is light in computation as we use only DC coefficients, i.e. scaled values of block mean values, as explained in Annex 1.

Next to the average values also the normalized 'variances' of the Y, U and V components, $Y_{DC\_VAR\_Norm}(N)$, $U_{DC\_VAR\_Norm}(N)$ and $V_{DC\_VAR\_Norm}(N)$, respectively, are calculated to improve the robustness of the monochrome frame detector with

$$Y_{DC\_VAR\_Norm}(N)=\frac{\sum_{i=LB+1}^{SPF-LB-1}\sum_{j=1}^{MBPS}\left(\left|\overline{Y_{DC}[N,i,j,0]}-\overline{Y_{DC}[N,i,j+1,3]}\right|+\left|\overline{Y_{DC}[N,i,j+1,1]}-\overline{Y_{DC}[N,i,j,3]}\right|\right)}{\max(Y_{DC\_VAR})*(SPF-2*LB)*MBPS*2} \quad (4\text{-}9),$$

$$U_{DC\_VAR\_Norm}(N)=\frac{\sum_{i=LB+1}^{SPF-LB-1}\sum_{j=1}^{\left\lfloor\frac{MBPS}{2}\right\rfloor}\left(\left|\overline{U_{DC}[N,i,2j-1]}-\overline{U_{DC}[N,i,2j]}\right|\right)}{\max(U_{DC\_VAR})*(SPF-2*LB)*\left\lfloor\frac{MBPS}{2}\right\rfloor} \quad (4\text{-}10),$$

$$V_{DC\_VAR\_Norm}(N)=\frac{\sum_{i=LB+1}^{SPF-LB-1}\sum_{j=1}^{\left\lfloor\frac{MBPS}{2}\right\rfloor}\left(\left|\overline{V_{DC}[N,i,2j-1]}-\overline{V_{DC}[N,i,2j]}\right|\right)}{\max(V_{DC\_VAR})*(SPF-2*LB)*\left\lfloor\frac{MBPS}{2}\right\rfloor} \quad (4\text{-}11),$$

as schematically explained in Figure 30. Statistical analysis on manually annotated ground truth showed, that with $Y_{DC\_VAR\_Norm}(N)$, equation (4-9), optimal black and monochrome frame detection results are achievable. For simplicity reasons $Y_{DC\_VAR\_Norm}(N)$ is normalized to one, i.e. range [0 .. 1]. Hereafter we defined empirically some thresholds for black frames and monochrome frames. For $Y_{DC\_VAR\_Norm}(N)<0{,}009$ frames are labeled as black frames and frames with $0{,}009 \le Y_{DC\_VAR\_Norm}(N) \le 0{,}015$ are indexed as monochrome frames.

Letterbox detection for format classification in MPEG-2

The smooth transition and, therefore, parallel existence of both 4:3 and 16:9 TV contents and screens, i.e. two TV formats, necessitates to render e.g. 16:9 content on 4:3 displays with letterboxes.

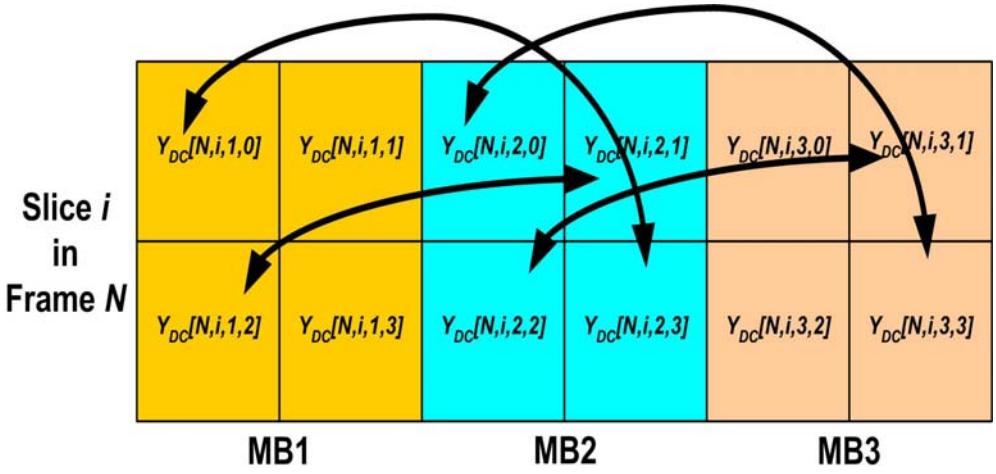


Figure 30. $Y_{DC\_VAR\_Norm}(N)$ *luminance* differential value.

Letterboxes are black bars spanning across the top and bottom slices of a frame as shown in Figure 31. Hence, when rendering 16:9 content on a 4:3 display the missing slices are replaced by black bars. The detection between 16:9 and 4:3 format, is not only valuable to distinct between normal video content, e.g. soaps, and feature films, which are often recorded in a 16:9 cinema format. But it is also useful for commercial block detection, e.g. when a 16:9 feature film with letterboxes is interrupted by individual 4;3 advertsiment clips. For the detector we compare the normalized DC average luminance value,

$$Y_{DC\_AV\_LB\_Norm}(N) = \frac{1}{\max(Y_{DC}) * LBs * MBPS * 4} \sum_{i=LB1}^{LB2} \sum_{j=1}^{MBPS} \sum_{b=0}^{3} Y_{DC}[N,i,j,b] \qquad (4\text{-}12),$$

with the normalized DC variance luminance value,

$$Y_{DC\_VAR\_\_LB\_Norm}(N) = \frac{\sum_{i=LB1}^{LB2} \sum_{j=1}^{MBPS} \left(\left| \overline{Y_{DC}[N,i,j,0]} - \overline{Y_{DC}[N,i,j+1,3]} \right| + \left| \overline{Y_{DC}[N,i,j+1,1]} - \overline{Y_{DC}[N,i,j,3]} \right| \right)}{\max(Y_{DC\_VAR}) * LBs * MBPS * 2} \qquad (4\text{-}13).$$

The applied index numbers for letterbox slices, i.e. *LB1* to *LB2,* are summarized in Annex 1 in Table Table 57 for various formats. For our experiments we use PAL D1 content, i.e. for the upper letterbox calculation *LB1=1*, *LB2=4* and *LBs=4* and for the lower *LB1=33*, *LB2=36* and *LBs=4*. As in the case of black / mono-chrome frame detection the equivalent normalized luminance, here $Y_{DC\_VAR\_LB\_Norm}(N)$ of equation (4-13), outperformes the predecessing one (4-12) what concerns detection robustness. Statistical analysis based on manually annotated ground truth resulats in a threshold $Y_{DC\_VAR\_LB\_Norm}(N) \leq 0{,}012$ for letterbox detection. Hence, if either the upper letterbox part of the lower letterbox part falls short the detection threshold than the frame is indexed as one containing a letterbox.

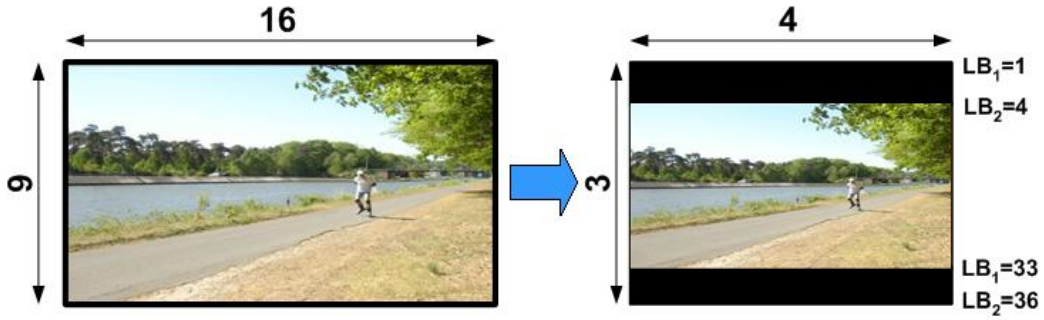


Figure 31. Conversion from 16:9 to 4:3 format with letterboxes[1].

4.1.2 Video mid-level features

In this section we introduce various video mid-level features we research and develop in the context of our PhD research to fulfil our task of semantic chaptering. These mid-level features apply as input several of our previously described low-level parameters.

Shot Boundary Detection – Cut Detection

In our work, temporal segmentation of video content, i.e. *shot boundary detection* (SBD), is an essential element of various spatio-temporal video-processing technologies, as we described in chapter 3 'State-of-the-art'. SBDs can be classified into *cut detector* CD and *gradual transition detector* GTD. The importance of SBDs and strict requirements (real-time analysis at 25 fps with above 96% precision and recall) based on platform, processing and performance constraints force us to develop and to benchmark various own cut detectors. In particular, the cut detector that is finally chosen has to be reliable since temporal segmentation forms the basis for a multitude of other mid-level and high-level analytical features. The need enabling appropriate trade-offs to be made between reliability and the required processing power, necessitates to research four cut detector algorithms benchmarked against a ground truth of a generic, culturally-diverse multi-genre AV corpus. The latter is presented in the next section 0. In the following sections, i.e. section 0 to section 0 we will introduce three cut detectors we propose in this work and one one cut detector from academic research. In section 0, the four cut detectors are benchmarked against each other and the best performing cut detector is then selected, as also published by us in [103]. Finally, a post-processing step is applied, as presented in section 0, and the potential future work is summarized in the conclusion of this section in 0.

AV Corpus for Objective Development of Content Analysis Algorithms

In order to be able to objectively benchmark the quality of various content analysis algorithms, used e.g. for applications in CE recording devices, a representative AV corpus, with manual or semi-automatically generated ground truth, has to be defined. In the CE context 'representative' means, that the AV corpus covers the average content recorded by a representative group of consumers. Moreover, the term *ground truth* is used for the correct and objective indexation data, i.e. manual annotations based on objective rules, which is required to objectively benchmark automatically generated results of content analysis algorithms. The on news focused, and therefore narrow scope genre-specific nature of the AV corpus of the benchmark initiative called TrecVid

[104] (2003), and the absents of an viable alternative forces us to set-up a more genre-wise AV benchmark corpus, which we present next.

Selection of AV content genres according to population pyramids

First of all, sufficiently objective rules have to be specified to select an appropriate AV corpus to benchmark our content analysis methods, which we propose in this work. Hence, based on these objectives, which we describe below, we decided to record 20 hours of content. Hereafter, we studied the consumer needs to find an optimal repartition of these 20 hours of AV broadcast content for such a representative AV corpus.

Hence, firstly we identified the distribution of a global (world-wide) consumer group by means of population pyramids, based on data acquired from the Internet [105]. To keep it manageable, only the populations of China, the United States of America, and the European Union (15 states) are considered.

The following graphs (see Figure 32) represent demographic distribution across various selected geographical regions[3] and time variations (see basic data at Table 59, Table 60 and Table 61 in Annex 3). In Figure 32 we show first the three demographic distributions of China for the years 2002, 2025 and 2050, and subsequently the same distributions for USA and European Union. In each graph the left side represents the male-, and the right side the female population distribution ('population in millions') across the various age classes, which are represented on the vertical axis. The age classes are clustered into 16 equal groups (from 0 to 80) and one group containing the population above 80 years.

In the next step, the following population age classes[4] are chosen as focus age groups: 15 – 20, 20 – 25, 25 – 35, 35 – 45, 45 – 55 and 55 – 65. As Table 3 shows, the selected age groups represent the majority of the entire population.

[3] Selection of three geographic areas is necessary due to time constraints. A broader geographic coverage will be the aim for future research.

[4] An extension of this work will be to include other age classes to achieve a better demographic coverage.

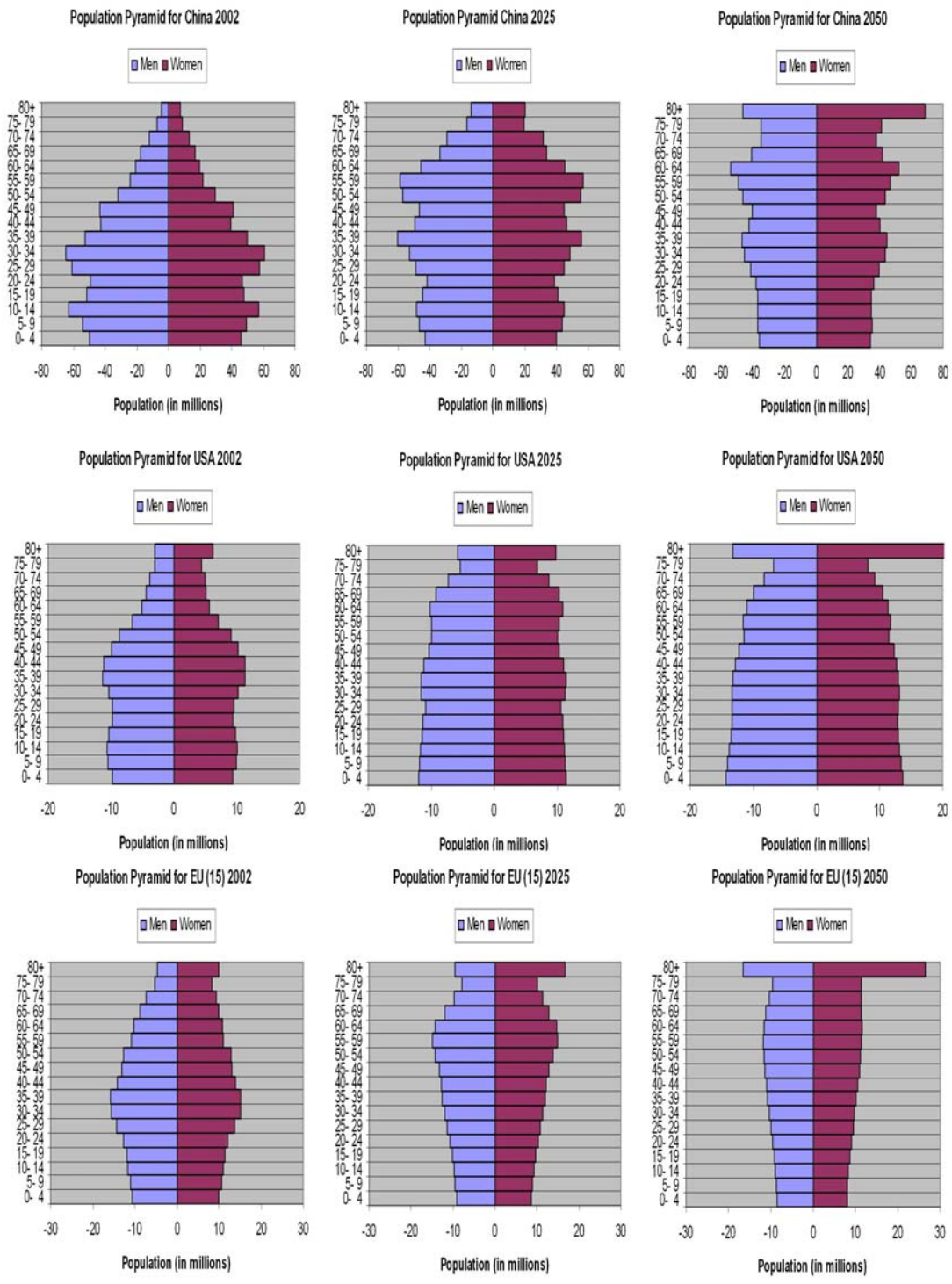


Figure 32: Population pyramids China, USA and EU.

Statistical Calculation

Once various age classes are selected, the distribution of each class has to be estimated and comparisons made.

Table 3 summarizes the size (i.e. number of people) of each age class, where the purple zones represent the selected age classes and the cell called 'Focus group Total' – also in purple - contains the total number of persons selected, i.e. the total target group for our research results and final applications. Furthermore, the percentage of each age class is calculated in comparison with the rest of the selected group – light blue column. In this way it is possible to estimate the proportion of each demographic class inside the focus group. Moreover, the yellow column represents the dispatch of the total corpus time – 20 hours - according to the percentage calculated before. We suppose that the target group audience f a given age group prefers to watch programs of a specific genre according to the proportion we show in Table 4. This table illustrates the time-wise dispatch of the 20 hours of AV content, attributed to each age class – for the scope of this work only based on the 2002 population - across the different kinds of genres of TV programs, which are:

- Series: all kind of series program (e.g. Friends),
- Shows: all kind of variety programs and popular programs,
- Movies: all kind of feature films (action, romantic, dramatic, etc…),
- Info + Pol: deals with news items and political discussions,
- Mag: all kind of TV magazines and TV reports (scientific, educational)
- Cartoons: all kinds of cartoons,
- Sports: any kind of sports (soccer, basketball, etc) and Music: video clips.

	2002					2025				2050			
Age classes	Europe [mill]	USA [mill]	China [mill]	%	Time	Europe [mill]	USA [mill]	China [mill]	%	Europe [mill]	USA [mill]	China [mill]	%
0 -> 5	18	18.5	98	-	-	16	22	88		16	27	78	
5-> 10	19	18.5	101	-	-	16.5	21	91		16.5	26.5	78	
10-> 15	21	20.5	121	-	-	17	21	88		17	26.5	79	
15 -> 20	22	20	99	11.2	2h15min	17.5	21	90	9	17.5	27	82	9.5
20 -> 25	22.5	18.5	99	11.1	2h13min	18	43	97	11.1	18	26	82.5	9.5
25 -> 35	46	38	247	26.3	5h17min	42	44	194	19.6	34.5	52	185	20.5
35 -> 45	45	46	183	21.8	4h17min	44	42	216	21.2	35	50	173	19.4
45 -> 55	43	39	143	18	3h37min	45	39	203	20.1	36	47	176	19.5
55 -> 65	32	26.5	87	11.6	2h21min	46	22	204	19	39	44	204	21.6
> 65	44	31.5	84	-	-	79	63.5	194		93	82	300	
Total	312.5	277	1262	100	20h	341	338.5	1467	100	322.5	408	1437.5	100
Focus group Total	1256.5					1427.5				1328.5			

Table 3. Number of people: age class in 2002, 2025 and 2050 [105].

Table 4: Time dispatching: according to age classes and A/V genres.

Type\Age	15 -> 20	20 -> 25	25 -> 35	35 -> 45	45 -> 55	55 -> 65		
Series	11,1%	11,5%	20,3%	14,2%	15,5%	10,3%		
Shows	11,1%	11,5%	20,3%	16,3%	13,3%	10,3%		
Movies	22,2%	23%	18,7%	24,5%	26,6%	41,2%		
Infos + Pol	0,0%	3,8%	1,6%	8,2%	8,9%	6,8%		
Mag	7,4%	7,6%	14,1%	16,4%	17,7%	24,1%		
Cartoons	22,2%	7,6%	4,6%	2,1%	0,0%	0,0%		
Sports	3,7%	11,5%	12,5%	12,3%	11,1%	3,4%		
Music	22,2%	23%	7,8%	6,1%	6,7%	3,4%		
Total (%)	100	100	100	100	100	100		
Total time (h)	2h15min	2h10min	5h20min	4h05min	3h45min	2h25min	~ 20h	%
				↓			~ 20h	100
Series	15 min	15 min	65 min	35 min	35 min	15 min	~ 3h	~ 15
Shows	15 min	15 min	65 min	40 min	30 min	15 min	~ 3h	~ 15
Movies	30 min	30 min	60 min	60 min	60 min	60 min	~ 5h	~ 25
Infos + Pol	0 min	5 min	5 min	20 min	20 min	10 min	~ 1h	~ 5
Mag	10 min	10 min	45 min	40 min	40 min	35 min	~ 3h	~ 15
Cartoons	30 min	10 min	15 min	5 min	0 min	0 min	~ 1h	~ 5
Sports	5 min	15 min	40 min	30 min	25 min	5 min	~ 2h	~ 10
Music	30 min	30 min	25 min	15 min	15 min	5 min	~ 2h	~ 10

The genre-wise distribution per age class is based on a user evaluation study (minimal 10 users per age class) that is done as part of this work. Moreover, the target time of each genre, see Table 4, is further distributed across the following groups:

- various geographic/national channels (USA, UK, France, China, Germany),
- analogue or digital transmission channels and
- public or commercial channels.

Concerning the distribution across the geographical/national regions the decision is (according to business interests, with an emphasis on Europe) to record[5]:

- 12 hours of European TV programs (French, German/Dutch, Italian and UK),
- 4 hours of US TV programs and
- 4 hours of Chinese TV programs.

As to distribution by TV source the decision is taken to record

- 50% from analogue, i.e. terrestrial, and
- 50% from digital, e.g. DVB-S, TV sources.

Finally, the AV corpus is split into public or commercial channels as follows:

- 30% of public channels and
- 70% of commercial channels.

[5] First iteration only, to keep it manageable.

All these percentages and distributions, presented above, are based on time dispatch as shown in Table 5. Moreover, the blue cells represent digital, e.g. DVB-S (satellite), channels and the pinks cells represent analogue channels, e.g. terrestrial or cable.
The following abbreviations are used in Table 5:

- EU_Pub: European program on public/national channels
- US_Pub: American program on public/national channels
- Ch_Pub: Chinese program on public/national channels
- EU_Com: European program on commercial channels
- US_Com: American program on commercial channels
- Ch_Com: Chinese program on commercial channels

Table 5 represents the final distribution of 20 hours of AV content used as evaluation and benchmark AV corpus to test the robustness of the various AV content analysis algorithms. The corpus does not represent all feasible geographic and demographic groups, but can be seen as a pragmatic approach towards a first meaningful and useful worldwide AV corpus for evaluation purposes. All content is stored in MPEG-2 MP @ ML program stream format with MPEG-1 layer 2 audio. EU and Chinese content is stored in PAL at 25 fps in D1 resolution (720*586) and US content in NTSC at 29,97 fps in D1 resolution.
In the following sections we will present three cut detectors we develop. Then in section 0 we will apply our AV corpus to benchmark the our cut detectors and to benchmark them against each other and another available cut detector from academic research.

Table 5: Time dispatching according to the transmission by region, (analogue: pink, digital: blue) and channel.

	Series (min)	Shows (min)	Movies (min)	Infos + Pol (min)	Mag (min)	Cartoons (min)	Sports (min)	Music (min)
EU_Pub	0	30	100	30	60	15	30	0
US_Pub	0	0	0	15	15	0	0	0
Ch_Pub	0	0	0	15	15	15	15	0
EU_Com	120	60	100	0	60	15	60	40
US_Com	30	60	60	0	30	15	15	25
Ch_Com	30	30	40	0	30	0	0	25
Total	180	180	300	60	180	60	120	120

Macroblock correlation cut detector (MBC CD)

State-of-the-art video compression systems such as video encoders, as shown schematically in Figure 33, contain among others a video compression block with *motion estimator* (ME) as further explained in [43] and [106]. The motion estimator identifies the best matching macroblock between current frame and successor (or predecessor) frame by means of minimizing the macroblock *mean absolute difference* (MAD) value (equation (4-1)). MAD is available at position *b* in a video encoder, as shown in Figure 33. The latter can be seen as motion compensated macroblock inter-frame correlation factor, which we introduced in 4.1.1 and explain in more detail in Annex 1 (annex MPEG-2). Consecutively, the total sum $MAD_{total}(N)$, which is the sum of all MADs of all macroblocks of all non-subtitle slices of the entire frame, the nominator in equation (4-2), is normalized with it's maximal achievable value, the denominator in equation (4-2), resulting in $MAD_{Norm}(N) \in \{0,..,1\}$ of equation (4-2) with *N* representing the frame instance in the video sequence.

At cut instances the inter-frame correlation decreases dramatically, which results in a dirac-like peaks of $MAD_{Norm}(N)$ at those abrupt cut transitions, as shown in Figure 34 (left). Hereon, for each frame instance *N* the $MAD_{Norm}(N)$ is compared to an adaptive threshold A_Th_N, which is based on a mean value of MAD_{Norm},

$$A\_Th_N = Th * \frac{1}{2(W_1 - W_2)} \left(\sum_{i=N-W_1}^{N-W_2} MAD_{Norm\ i} + \sum_{i=N+W_2}^{N+W_1} MAD_{Norm\ i} \right) \qquad (4\text{-}14).$$

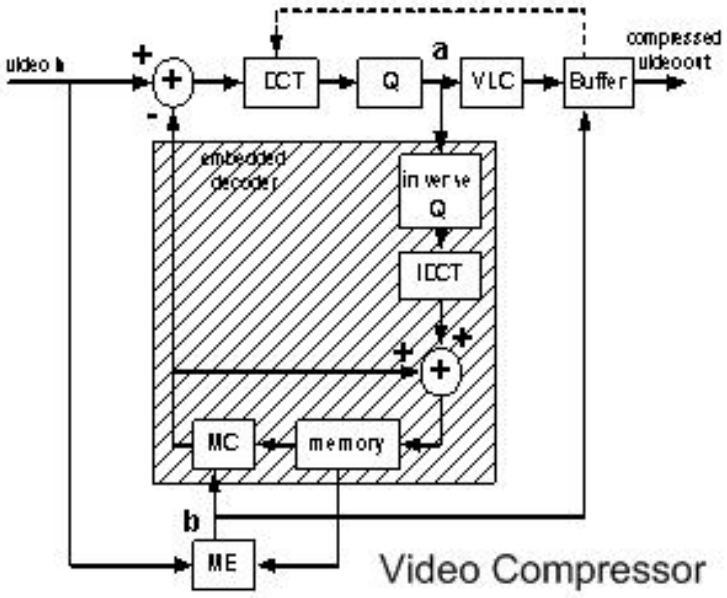


Figure 33. MAD-generating encoder.

Here $Th \in \{2..10\}$ is a fixed factor, by which $MAD_{Norm}(N)$ has to minimally exceed the local averaged value of MAD_{Norm}, with $W_1 \in \{3..41\}$ and $W_2 \in \{0..3\}$ representing a global and a local window length in number of frames, respectively, and with N being the index of the current frame investigated, as sketched in Figure 34 (right).

An inner window W_2 (here $W_2=1$) is used to reduce over segmentation, which can happen due to compression artefacts and illumination changes. Finally, instances, at which $MAD_{Norm}(N)$ exceeds A_Th_N, are indexed as cut transitions. The analysis of optimal settings, $W_2=13$ (further W) and $Th=3$, and results will be presented latter in this section.

Field difference cut detector (FD CD)

Another cut detector developed in this work is the *field difference cut detector* FD CD, applying technologies from the interlaced / progressive scan video domain. Here we calculate for each field, e.g. here field *n,* in an interlaced video signal (see *Annex 2)* an *inter-field dissimilarity IFD[n]* of the current field *n* and the predecessor field *n-1*. As example shown in Figure 35 (up right), this could be the *IFD[n]* between the odd field with field index 6 (field *n)* and the predecessor even field E with field index 5 (field *n-1)*.

The luminance signal *I(x,y,n)*, with the spatial coordinates *(x,y)* and the field index *n* (here in our example this is field index *6*), cannot be directly compared with the predecessor-field luminance value *I(x,y,n-1)*, at the same spatial position *(x,y)*, due to the different interlace phases of the two fields, as presented in Annex 2.

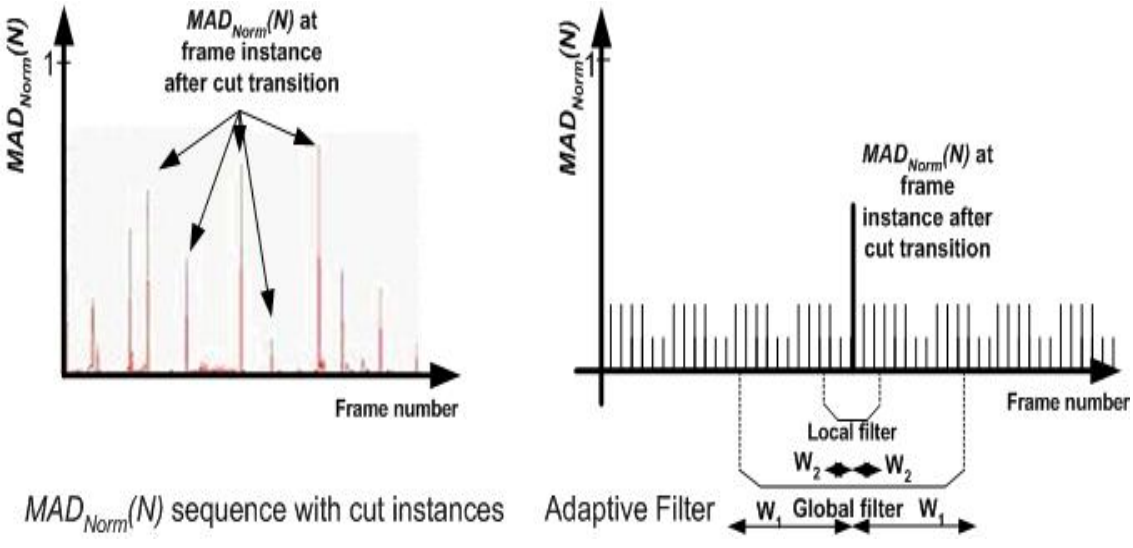


Figure 34. Video encoder output: $MAD_{Norm}(N)$ (left) and adaptive filter (right).

Instead, *I(x,y,n)* is compared with de-interlaced luminance value $I_{dei}(x,y,n-1)$,

$$I_{dei}(x,y,n-1) = median\ \left(I(x,y,n), I(x,y+1,n-1), I(x,y-1,n-1)\right) \quad \text{(4-15)},$$

using vertical temporal median as de-interlacing method, as explained in Figure 35 (top). The resulting *IFD[n],* shown in Figure 35 (top right), is defined as

$$IFD[n] = \frac{1}{N}\left|\left\{(x,y) \in P[n] : \left|I(x,y,n) - I_{dei}(x,y,n-1)\right| > T_{dis}\right\}\right| \quad \text{(4-16)},$$

with *P[n]* representing a pixel set with size *N*, containing the spatial positions (i.e. all pixels) in field *n*, and where T_{dis} is a preset threshold. We inherited the latter from the rendering domain, i.e. film mode detection in consumer rendering devices, where T_{DIS} is empirically chosen, as published in [107], with

$$T_{dis} = round\left(\frac{255}{10^{\frac{PSNR}{20}}}\right) \quad and \quad an \quad PSNR = 10 * \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \quad \text{(4-17)},$$

PSNR represents an heuristically derived peak signal-to-noise ratio *PSNR* (*noiselevel)* of 32 dB, using mean square error *MSE* of differences of pixels in fields and maximum pixel value MAX_I=255 (maximum luminance range of 255), which results in threshold of a T_{dis}=6 [107], i.e. pixel luminance differences of >5. *IFD[n]*$\in$*{0..1}* itself represents the relative percentage of dissimilar pixels counted.

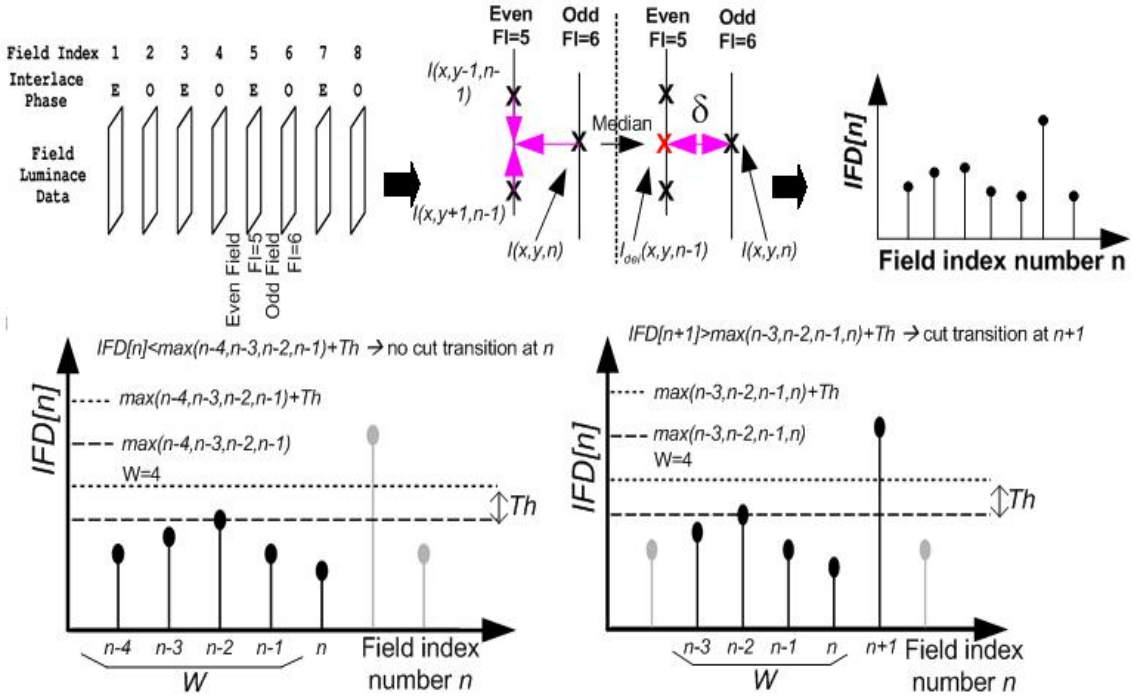


Figure 35. *Inter field difference* IFD calculation.

Finally, instances, here fields, at which the local *IFD[n]* exceeds the maximum *IFD[n]* value of the past $W \in \{2..15\}$ fields (*W* represents here a selected window length), increased by a chosen threshold value $Th \in \{0.1..0.7\}$, as defined by

$$IFD[n] > IFD[n-m] + Th, \forall m \in \{1,2,3,\cdots,W-1,W\} \quad \text{(4-18)},$$

are marked here cut instances. This is illustrated in Figure 35 (bottom). Such a cut detector is field level accurate, i.e. its cut instance resolution is field accurate. In our experience we achieved optimal results, by applying our AV benchmark corpus, with *W*=15 fields and *Th=0.2*, as presented in 0.

Colour segmentation cut detector (CS CD)

The *macroblock correlation* MB CD and *field difference cut detector* FD CD compare frames on macroblock and pixel level, respectively. Histogram-based cut detectors, on the other hand, compare frames on frame level. The third cut detector developed in this work resides on an intermediate level: it is based on RGB (or YUV) color segmentation, as published by us in [108], which we simply applied as such. Here, a watershed-like segmentation, presented in [49], is used as a pre-processing step, which does not result in object segmentation (as objects may have widely varying colors, as shown in Figure 36 - right). The main idea here is that at a cut instance the segmentations of two consecutive frames will be different while consecutively captured frames of a video sequence, i.e. a shot, have similar segmentations. Therefore, segmentation dissimilarity can be used for cut detection, as shown in Figure 36 (right) for an abrupt cut instance.

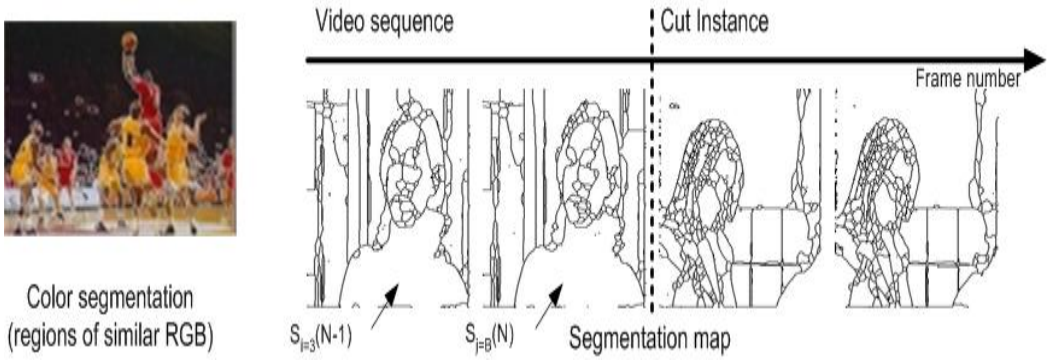


Figure 36: Frames around a cut transition (top) and corresponding segmentation[1].

The similarity of consecutive segmentation maps is quantified by a *consistency measure C*. For robustness reasons we also take motion into consideration, hence, we always motion compensate the segmentation of predecessor frames. The *consistency measure* C compares all motion compensated (to handle object motion) segments $S_i(N-1)$ of frame *N-1* with $S_j(N)$ of frame *N* (non-motion compensated), where *i* and *j* are segment indices. Each segment clusters a number of pixels, which are within its boundaries. Here after we define an overlap matrix *A*, which represents for all possible segments between frame *N* and *N-1* the number of joint pixels, i.e. pixels being member of segment $S_i(N-1)$ and of $S_j(N)$, as presented in Figure 37 (a), with A_{ij},

$$A_{ij} = \left| \{ (x, y) : S_i(N-1) \cap S_j(N) \} \right| \quad (4\text{-}19).$$

Subsequently, for each segment in frame *N* the best matching segment in frame *N-1* is selected that is those with most overlapping pixels. To formally define this, let us consider sets of segments of frame *N*, i.e. *S(N),* and frame *N-1*, i.e. *S(N-1).* Subsequently, segments of frame N-1, here $S_i(N-1)$, are mapped onto segment of frame N, here $S_j(N)$, if $A_{ij} \geq A_{iq}$ for all *q* and/or $A_{ij} \geq A_{tj}$ for all *t*. This results in two consistency measures $C_{AND}(N)$ and $C_{OR}(N)$, as explained in Figure 37 (b) and (c), resulting in:

$$C_{AND}(N) = \frac{1}{Pi} \sum A_{ij} : \forall \{i, j, q, t\} : A_{ij} \geq A_{iq} \wedge A_{ij} \geq A_{tj} \quad (4\text{-}20),$$

$$C_{OR}(N) = \frac{1}{Pi} \sum A_{ij} : \forall \{i, j, q, t\} : A_{ij} \geq A_{iq} \vee A_{ij} \geq A_{tj} \quad (4\text{-}21),$$

with *Pi* representing the total number of pixels of frame *N.*

To be more specific, the AND consistency measure is defined as the relative sum of all member pixels of all the mutual best matching segments between frame *N-1* and *N*, e.g. $C_{AND}(N)$ amounts in the example of Figure 37 (b) to (120 pixels + 250 pixels +115 pixels) / 1000 pixels = 48.5%, while $C_{OR}(N)$ results in 76.8%.

Frame N-1 / Frame N	$S_{i=1}(N-1)$	$S_{i=2}(N-1)$	$S_{i=3}(N-1)$	$S_{i=4}(N-1)$	$S_{i=5}(N-1)$
$S_{j=A}(N)$	120	50	0	0	0
$S_{j=B}(N)$	20	75	250	15	2
$S_{j=C}(N)$	0	5	125	80	10
$S_{j=D}(N)$	0	0	50	115	83

a) Overlap Matrix A

e.g. $A_{i=B,j=3}$: overlap of # of pixels between segment $S_{j=B}(N)$ of frame N and motion compensated segment $S_{i=3}(N-1)$ of frame $N-1$

Frame N-1 / Frame N	$S_{i=1}(N-1)$	$S_{i=2}(N-1)$	$S_{i=3}(N-1)$	$S_{i=4}(N-1)$	$S_{i=5}(N-1)$
$S_{j=A}(N)$	120	50	0	0	0
$S_{j=B}(N)$	20	75	250	15	2
$S_{j=C}(N)$	0	5	125	80	10
$S_{j=D}(N)$	0	0	50	115	83

b) AND consistency measure → $C_{AND}(N)$

e.g. $C_{AND}(N)$ = (120 + 250+115) / 1000 = 48.5 %

Frame N-1 / Frame N	$S_{i=1}(N-1)$	$S_{i=2}(N-1)$	$S_{i=3}(N-1)$	$S_{i=4}(N-1)$	$S_{i=5}(N-1)$
$S_{j=A}(N)$	120	50	0	0	0
$S_{j=B}(N)$	20	75	250	15	2
$S_{j=C}(N)$	0	5	125	80	10
$S_{j=D}(N)$	0	0	50	115	83

c) OR consistency measure → $C_{OR}(N)$

e.g. $C_{OR}(N)$ = (120 + 250+115 +75+125+83) / 1000 = 76.8 %

Figure 37. Consistency measure C_{AND} and C_{OR}.

$C_{AND}(N)$ measures the normalized area of all segments, which map bi-directionally onto each other (red numbers), whereas $C_{OR}(N)$ measures the area of all uni-directional mappings (red and blue numbers). A division by the total number of frame pixels *Pi* normalizes the consistency values $C_{AND}(N)$ and $C_{OR}(N)$. Notches in $C_{AND}(N)$ indicate a cut instance resulting in an accurate indicator for video cuts, as shown in Figure 38. However image noise or texture may cause segments to be split up or merged in subsequent frames, as visualized in Figure 38 (right). This decreases $C_{AND}(N)$, as for any split segment it will only count the area of the largest of the newly generated smaller segments. $C_{OR}(N)$, on the other hand, is insensitive to this effect as all smaller segments map uni-directionally to the big segment.

Hence, we can combine the two consistency measures, $C_{AND}(N)$ and $C_{OR}(N)$ into $C(N)$,

$$\alpha(N) = \left| C_{OR}(N) - mean\{C_{OR}(N - W_C), \ldots, C_{OR}(N-1)\} \right|$$
$$C(N) = C_{AND}(N) - \alpha(N) * (C_{AND}(N-1) - C_{AND}(N)) \qquad (4\text{-}22),$$

with W_c being the size of an averaging temporal window. This combination results in the fact, that notches - representing correct transitions - become more exposed in comparison to $C_{AND}(N)$, but for notches caused by segment splitting, as explained in Figure 38 (right), $C(N)$ remains more or less constant due to the insensitivity of $C_{OR}(N)$ to this particular cause, as shown in Figure 39 (left). Here, especially for difficult content, containing large textured areas, such as the grass in a soccer field or water surfaces, the performance of the cut detector improves. Finally, for each individual notch, a decision has to be taken as to whether or not it represents a cut instance.

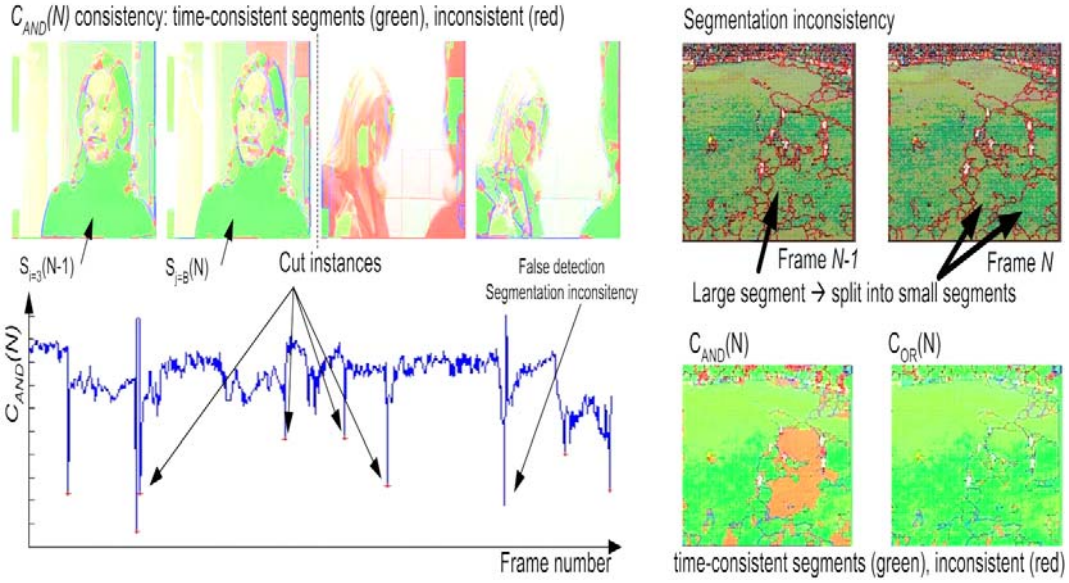


Figure 38. $C_{AND}(N)$ consistency measure and segmentation inconsistency (right)[1].

Adaptive threshold methods, as presented by Yussof in [53], are far more robust and flexible then fixed threshold ones. Three models were compared as follows,

(a) constant variance model: $Th_N = \mu_N - Th$ (4-23),

(b) proportional variance model: $Th_N = \frac{\mu_N}{Th}$ (4-24),

(c) Dugad model: $Th_N = \mu_N - Th * \sqrt{\sigma_N}$ (4-25),

with Th_N being the local threshold at instance N, Th a fixed threshold, μ_N and σ_N the mean and variance of $W+1$ consecutive consistency values of $C(N)$ around instance N, respectively. Proved to perform best, we adapt the proportional model of equation (4-24) slightly as had already been done for the MBC CD as shown in Figure 39 (right), with a sliding window of size $W+1 \in \{1..9\}$ and $Th \in \{1.4 .. 4\}$ resulting in the formula

$$SC(N) = \begin{cases} \text{cut} : C(N) < C(j)/Th, \forall j \in \{N - \frac{W}{2}, \dots, N + \frac{W}{2}\}, j \neq N \\ \text{no cut} : C(N) \geq C(j)/Th, \forall j \in \{N - \frac{W}{2}, \dots, N + \frac{W}{2}\}, j \neq N \end{cases} \quad (4\text{-}26).$$

The averaging window W should be smaller than the minimal duration between two consecutive cuts, as only one cut per window can be detected. Hence, a shot length analysis, presented in section 4.6.4, revealed that $W \leq 10$ frames helps to reduce the amount of missed transitions. On the basis of a benchmark analysis, as presented in the next section, the settings $W=8$ and $Th=1.4$ prove to perform best for the color-segmentation-based cut detector. We published this promising method in a European patent [108].

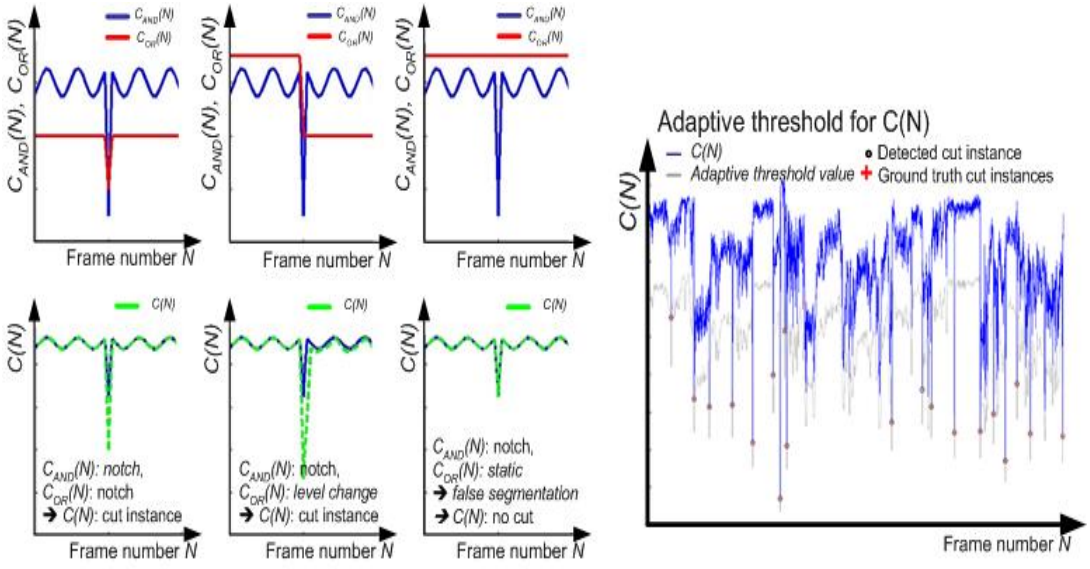


Figure 39. C(N) consistency measure (left) and adaptive threshold method (right).

Comparison of CDs

The next step is to analyze the performance of the three developed cut detectors, see Figure 40, and benchmark them against an available cut detector, see Figure 41, which participated in TrecVid 2004 [42]. The cut detector is called *rough indexing cut detector* RI CD, described in detail [56] and in the literature survey section 3.1.1.

Each detector is tested with several settings - Table 6, where one of the parameters is fixed at an optimal default value and the other one is varied, resulting in Figure 40. Finally, the three developed detectors are benchmarked against the RI CD using derived optimal settings for *Th* and *W*. In this way recall and precision can be tuned as desired.

Table 6. Cut detector settings.

	W [range]	*Th* [range]	Optimal *W*	Optimal *Th*	Overall *Re* [%]	Overall *Pr* [%]
MBC CD	3 .. 41	2 .. 10	13	3	92	83
FD CD	2 .. 15	0.2 .. 0.7	15	0.2	93	93
CS CD	1 .. 8	1.4 .. 4	8	1.4	93	90

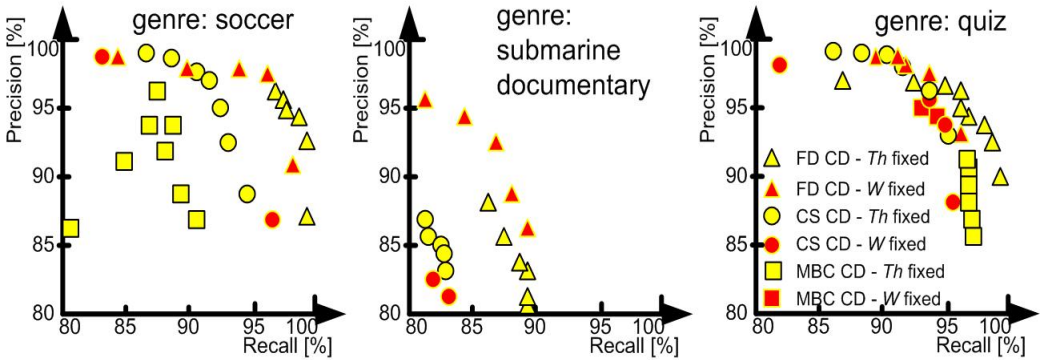


Figure 40. Cut detector results for three specific contents.

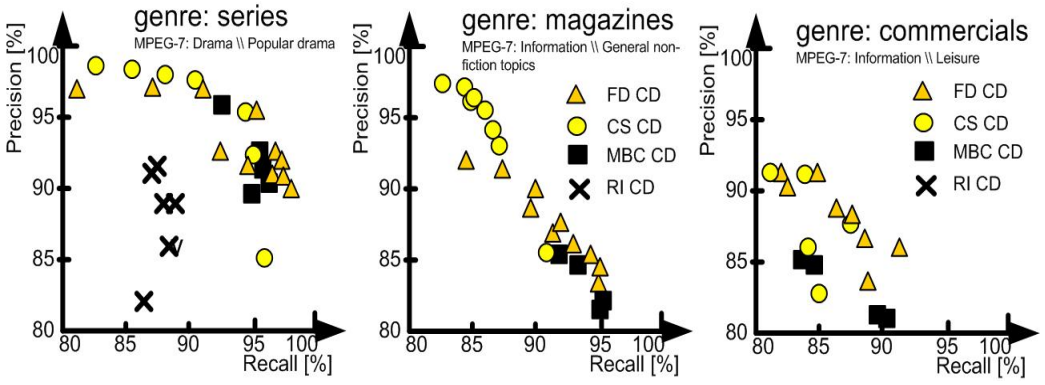


Figure 41: Recall and precision performance results of all four cut detectors.

All four detectors are tested on our AV content corpus, recorded in 25 frames per second PAL format horizontally sub-sampled to ½ D1 resolution, as presented in Annex 1. Content is carefully chosen to adequately represent real-world broadcasting material containing a variety of genres, such as series, magazines, commercials and sports events.

The performance is evaluated for each genre separately in terms of *precision* Pr, see equation (3-16), and recall Re, see equation (3-17) in section 3.1.1. Both, recall and precision are relevant to define the optimal parameter settings such as window size *W* and adaptive scaling threshold *Th*. They are relevant to assess the effectiveness of each individual cut detector, which can be done graphically by using so-called *Receiver Operator Characteristic* ROC graphs, i.e. plots representing true positive rates against false positive rates. Hence, tuning e.g. through the entire range of one algorithm variable results in series of Pr/Re coordinates.

The performance of the cut detectors presented relies on two parameters:

- *Th*, defining the required minimal (relative) difference of the current sample in relation to neighbouring samples in order to be indexed as a cut instance, and
- *W*, defining the number of neighbouring samples to be taken into account.

Performance

The evaluation revealed that the pixel- and segmentation-based detectors outperform the block-based methods, as expected. The rough indexing cut detector, which was tested on the TREC Video Corpus in the TRECVID2004 campaign, achieves a performance of 86.8% recall and 77.8% precision[6] there, and is used here as a reference benchmark. The rough indexing cut detector scores lower on recall, as shown in Figure 41 (left) for soaps/series, whereas the macroblock correlation cut detector scores lower on precision, as shown in Figure 41 (right) for commercials. On the other hand, the color segmentation cut detector, for which the ½ D1 content was further horizontally sub-sampled to 352-by-288 pixel frames, can reach high precision, but has a lower recall limit, as visible in Figure 41 (centre). The latter is especially sensitive to an inconsistent segmentation due to segment splitting, as shown in Figure 38 (right), a problem encountered especially in uniform and slightly textured regions. Examples of this are green light-textured soccer field sequences or submarine water sequences or dark video sequences sharing a similarly dominant color. This produces a noisy

[6] Due to the fact that it performs in compressed domain at I-P level and, therefore, can miss cuts at B frames.

constancy value, subsequently leading to missed and false detections, resulting in low recall and precision, as can be seen in Figure 40.

However, the detector can be improved in the future by (a) a better time-consistent segmentation[7], (b) a segment-size-dependent *Th*, (c) a motion-intensity-dependent *Th* or (d) a texture / homogeneity-dependent *Th.*

The results with the best performing *W* and *Th* of each detector for one genre, here series, are summarized in Table 7. We conclude that the pixel-based field difference cut detector performed best, and, hence, select this one for further evaluation in this work.

Table 7. Performance results for all four cut detector for genre series.

Genre: Series	Optimal *W*	Optimal *Th*	Overall *Re* [%]	Overall *Pr* [%
MBC CD	13	3	95.9	94.7
FD CD	15	0.2	96.1	96.3
RI CD	-	-	92.1	86.5
CS CD	8	1.4	93.4	97.0

Complexity, Latency and Robustness

Complexity: Two of the four detectors are cheap-to-compute by-products of existing solutions. The pixel-based field difference cut detector, a by-product of de-interlacing solutions, has the lowest complexity, as for each pixel only median and absolute difference values need computing. If done in software, the macroblock-based spatial correlation cut detector would be quite processing intensive but because it is a by-product of an already existing integrated circuit, as sketched in Figure 34, it provides an efficient video compression solution. The complexity of the rough indexing detector is intermediate, as it requires motion vectors and it robustly estimates an affine motion model. The measure ΔQ (see chapter 3) only needs to compute macroblock resolution. Finally, the color segmentation detector exceeds all others in complexity, as it requires costly image segmentation.

Latency: Both the field difference and rough indexing cut detector have zero-frame latency, as the resulting cut detection is immediately available for the current frame, making them suitable for on-line detection. The other two cut detectors use a symmetric window *W* surrounding the current frame, resulting in latencies of several frames.

[7] The overlap-probability, and hence robustness, increases with segment size, with decreasing motion errors and with clear texture. The latter decreases the probability of unstable segmentation.

Robustness: All cut detectors, with the single exception of the field difference cut detector require motion estimation (ME). Depending on the amount of motion as well as the quality of the ME, ME errors may become propagated into the resulting cut detector. The field difference cut detector is the most robust as it does not require ME, nor does it require any other parameters than *Th* and *W*. Using segmentation renders the color segmentation cut detector relatively resilient to small ME errors. The other two cut detectors are more critically reliant on accurate ME.

Overall

Considering the properties of all of the cut detectors, the field difference cut detector best meets the requirements exposing at real-time analysis high precision and recall. Nevertheless, depending on requirements (e.g. offline processing, low latency, low complexity) and available side information (motion vectors, segmentation, etc.), another cut detectors may be more suited to other specific applications. Moreover, Figure 41 reveals that the reliability of cut detectors is quite genre dependent. Each genre follows a fixed set of film (capture) rules, see section 4.5.1, and because of this each genre exhibits specific statistical attributes. Knowing what these statistical attributes are, e.g. the shot length can, in a subsequent step, be used to derive semantic information about new content.

Conclusions of the four cut detector benchmark

Two cut detectors are obtained as cheap-to-compute by-products of other video processing operations, e.g. MPEG encoding. The results on the AV corpus confirm that all cut detectors reach comparable levels of maturity/precision in detecting video cut transitions. Differences across genres are more pronounced than differences within genres for the different detectors. The field difference cut detector shows a good performance and has low complexity. Hence, it is in general preferred and, because it fulfils the requirements best, we select this detector as the cut detector of choice, with the default settings *W=15* and *Th=0.2* for subsequent analysis.

The user can control the precision/recall trade-off through a combination of threshold *Th* and window size *W* (tune-able precision). A topic for future research is gradual transitions detection. Whereas the macroblock- and field difference cut detectors compare subsequent frames, the color segmentation- (through segment tracking) and rough indexing one can handle larger inter-frame spacing, and as such may be more suitable for gradual transition detection.

Improvement of cut detector with feature-point-based similarity analysis

The field difference cut detector, as described in section 0, is due to its luminance (Y)- and pixel-matching-based nature very sensitive to impulsive illumination- and fast motion changes. Unfortunately, the producers of commercial content use those elements very often to augment the viewing experience. This includes light source variations such as the use of flashlights, sudden explosions or indoor illumination variations, but also abrupt transitions from indoor to outdoor settings. Moreover, commercial content very often contains special artistic effects such as steady or sudden camera vibrations and motions, and, furthermore, unexpected events such as objects or subjects passing the camera capturing space in close distance to the capturing unit. All these events mislead the field difference cut detector (*IFD[n]* exceeds at those moments *Th* as stated in (4-18)) resulting in a strong over-segmentation, reflected in a low shot boundary detector precision.

Hence, the aim is to use a cut detector setting from the previous section with high recall and medium precision, and, hereafter, to increase the precision by removing the over-segmentation instances, as sketched in Figure 42 (left-top).

The common nature of many of those above-mentioned events is their short visual deviation from the normal video flow. In other words, the video content before the specific event is often very similar to the video content after the event. Hence, we will exploit this property to increase the precision.

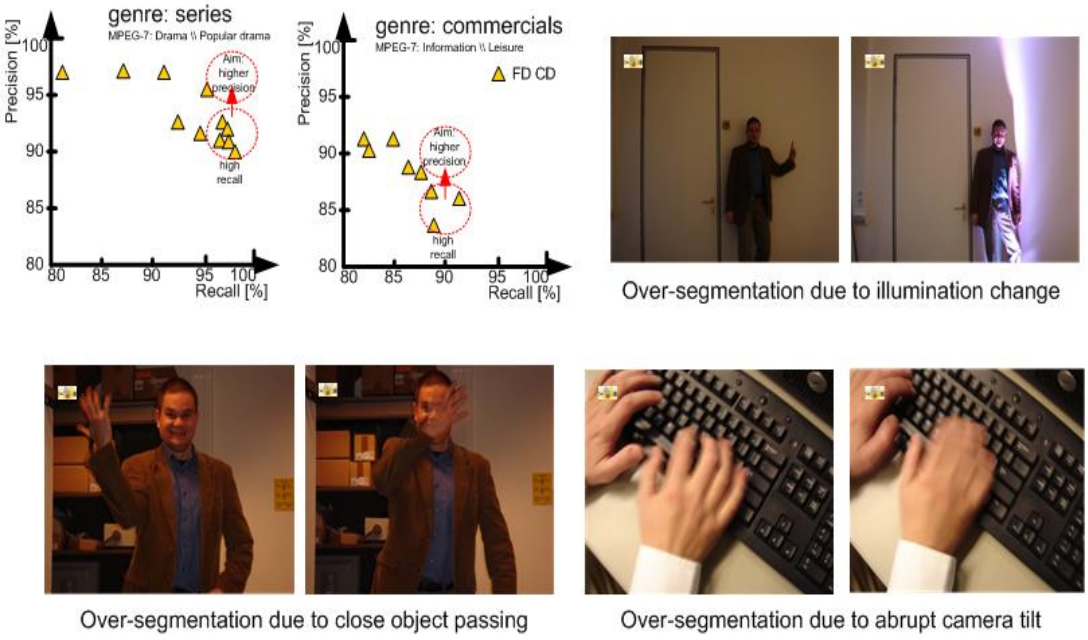


Figure 42. Over-segmentation with field difference cut detector[1].

Feature-point-similarity-enhanced cut detection verification

The aim is, therefore, to verify all field-difference-analysis-based cut instances by means of a similarity-based post-analysis between key frames pairs, i.e. key frames before and after the indexed cut instance (shot boundary, KF_x and KF_y), as shown in Figure 43. Comparing sets of feature points on key frames can do this verification.

The definition, detection and tracking of the here fore applied feature points are described in detail in the feature point section 4.5.2. Here, we suppose the availability of the required feature points $FP_i(KF_x)$ and $FP_j(KF_y)$. For the required key frame similarity check we use the percentage of tracked feature points (threshold $Th_{FP} \in \{0..100\}$ percentage of tracked feature points).

Hence, the aim of this post-processing is to eliminate false cut instance detections, which occur due to time-wise short interruptive events (e.g. flashlights). Because the video continuity of the video stream restores shortly after these instances, a correspondence can be established between the key frames flanking these short instances (i.e. one key frame before and after the event). This correspondence is characterized by a significant high number of tracked feature points based on the visual similarity between those key frame pairs (KF_x / KF_y). The average length of such temporal limited events can be taken into account by choosing an optimal symmetric window size W_1 ($W_1 \in \{1..3\}$) around each cut instance, as sketched in Figure 43.

Moreover, an optional extension to W_1 is introduced with window size W_2 increasing optimal key frame pair selection. If the selected key frames at $SB_{N-1} \pm W_1$ do not fulfil the criteria of a reasonable key frame (e.g. too blurry), it can be replaced with one within the range of $W_2 \leq 10$ (heuristically chosen). The evaluation and usage of W_2 will be further discussed in section 4.5.2.

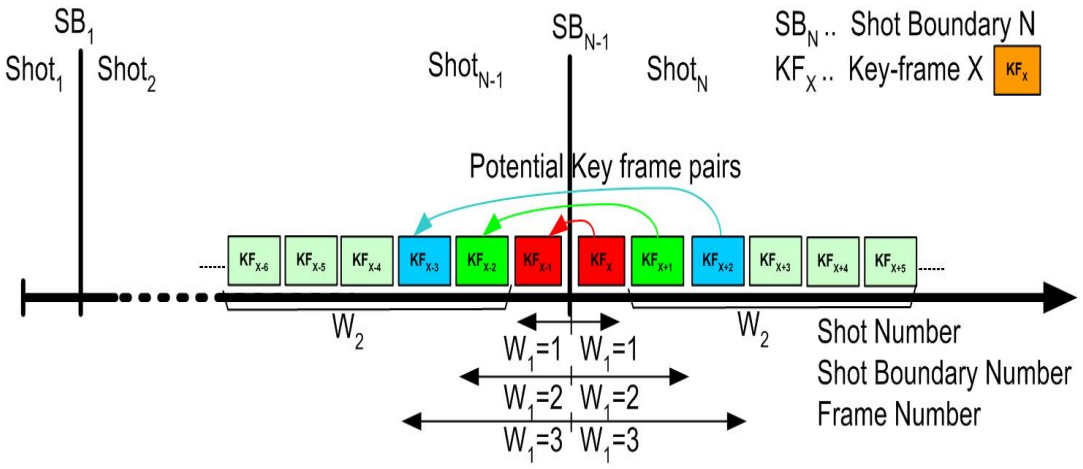


Figure 43. Cut detection verification through key frame similarity analysis.

A benchmark analysis on the AV corpus of section 0 (excluding series and movies) unveils, that the optimal trade-off between false cut detection reduction and loss of correct cut detections can be achieved with W_1=2 (optimal distance for temporal limited events) and Th_{FP}=*30* (optimal percentage of tracked feature points for key frame similarity analysis). Hence, at each cut instance the key frames pairs are provided as input to the feature-point-based similarity analysis, where at least 30% of all feature points have to be tracked to delete the cut index at this instance.

Performance of feature-point-similarity-enhanced cut detector

The elaborated nature of this work and the drive for quality required to focus finally on a subset of the entire AV corpus of section 4.1.2, in particular the genres movies and series are selected, as presented in Table 8 (the movies corpus was increased compared to section 4.1.2 due to its specific importance). The movies/series corpus contained in total a representative set of *4942+1591=6533* manually annotated ground truth cut instances (excluding non-content related inserts, e.g. commercial blocks). The results of the field-difference-based cut detector of section 4.1.2, with $W \in \{2..15\}$ and $Th \in \{0.1..0.7\}$, and a feature-point-based post processing of section 4.5.2, with $Th_{FP} \in \{0..100\}$, for the movies, series, commercials and channel adds genres are summarized in Figure 44.

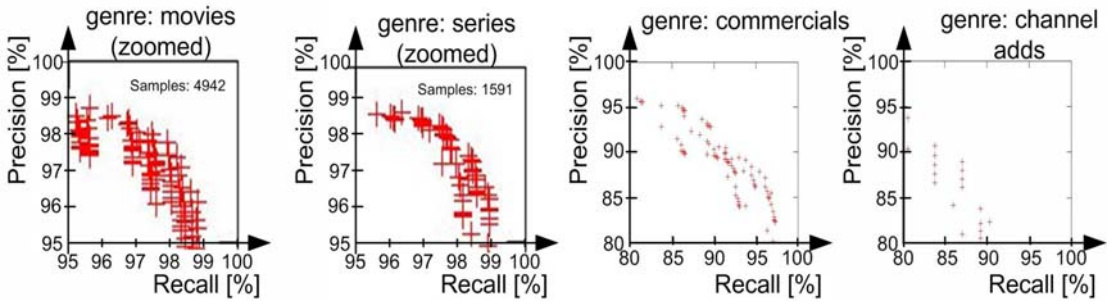


Figure 44. Cut detection results after FP-based post-processing.

Table 8. AV movies/series corpus.

Movies corpus	Total # GT SB (genre only)	# frames @ PAL
'ge1'	892	86700
'ge2'	314	65779
'nl'	1352	144359
'us_dig'	1208	156723
'us_ana'	1176	215370
Total	4942	668931
Total duration [hh:mm]		7 h 26 min

Series corpus	Total # GT SB (genre only)	# frames @ PAL
'nl1"	227	33970
'nl2'	212	33300
'ge1'	175	23008
'ge2'	495	59681
'gb'	482	37492
Total	1591	187451
Total duration [hh:mm]		2 h 5 min

Table 9. FD-based cut detection FDCD (*W=15*, *Th=0.2*) before (without) and after (with) feature-point-based post-processing (W_1=±2, Th_{FP} =30) results on movies/series corpus.

Genre	Total SBs (CDs)	Correct CDs	False CDs	Missed CDs	Recall [%]	Precision [%]
Movies before FP	4942	4855	194	87	98,2	96,1
Series before FP	1591	1551	34	40	97,5	97,4
M&S before FP	6533	6406	228	127	98,1	96,5
Movies after FP	4942	4855	134	87	98,2	97,3
Series after FP	1591	1551	24	40	97,5	98,5
M&S after FP	6533	6406	158	127	98,1	97,6

The results confirmed, that the independently, on the rest of the AV corpus derived optimal settings (field-difference settings: *W=15* / *Th=0.2, feature-point-similarity-based settings:* W_1*=2 /* Th_{FP}*=30*), do indeed perform best resulting in *R=98,2%* / *P=96,9%* for movies and *R=97,5%* / *P=98,3%* for series, summarized in Table 9, where '*False CDs'* represents over-segmentation.

Despite the fact that the increase in precision is rather limited, i.e. 1.1% to 1.2% in our experiments, as summarized in Table 9, this approach seems promising. In any case, the post-processing of detection results by comparing frames before and after a detected cut instance is a finding of this research work. The features for comparison have to be appropriately chosen and the decision rule has to be sufficiently discriminative. Feature points seem to us to be a good compromise between complexity and efficiency.

Resilience of feature-point-similarity-enhanced field difference cut detector

Over-segmentation (Precision)

An analysis of the resulting feature-point-post-processing cut detection results unveils, that the three main reasons for over-segmentation are

- temporal long 'illumination changes' (i.e. for movies the case for 50% of all over segmentation cases), as shown for two examples in Figure 45 (left),
- 'objects passing close' to the capturing device (i.e. in movies: 25% of all over-segmentation cases and in series: 35%), as shown in Figure 45 (right top), and
- '2/3-rule' (i.e. in series: 40%), as shown in Figure 45 (right bottom) of all over-segmentation cases. The exact distributions are summarized in Table 10 (left).

Over-segmentation, due to long-lasting illumination changes, is often not detected because of the optimal, but temporal short post-processing window W_1=±2, and the strong luminance dependency of the feature point. Illumination-adaptive scale invariant feature points, which are being under research at this present moment in time, could provide a viable solution here.

Table 10. Over-segmentation and missed cut detection evaluation.

	Over-segmentation (false detection) in % in relation to the total number of oversegmentation occurances					Missed cut detections in % in relation to the total number of missed occurances			
Genre	Number of occurrences	Illumination changes	Close objects passing	2/3-rule	Fuzzy content, fades,..	Number of occurrences	Motion & out-of-focus	Short shots w/ moving objects	Others
Movies	134	50%	25%	5%	20%	87	84%	5%	12%
Series	24	-	35%	40%	25%	40	60%	22%	17%

The second source of over-segmentation, the problem with out-of-focus objects passing close to the capturing device, will most probably remain for quite some time. But, more sophisticated methods, such as depth map analysis (used in 2D-to-3D video conversion) in combination with spatial texture information, could lead to promising results.

Over-segmentation due to the '2/3-rule' is a homemade problem based on the strict definition that at cut instances at least 2/3 of the frame content has to change (manual annotated ground truth). We witnessed that the end sequences of series and movies are very often downscaled and graphics at the frame boundaries display content related information, e.g. credits as shown in Figure 45 (right bottom). A spatial graphics-video discriminator, which is currently under development, will be able to identify the graphic bars enabling exclusive video part analysis and a consequent reduction in over-segmentation.

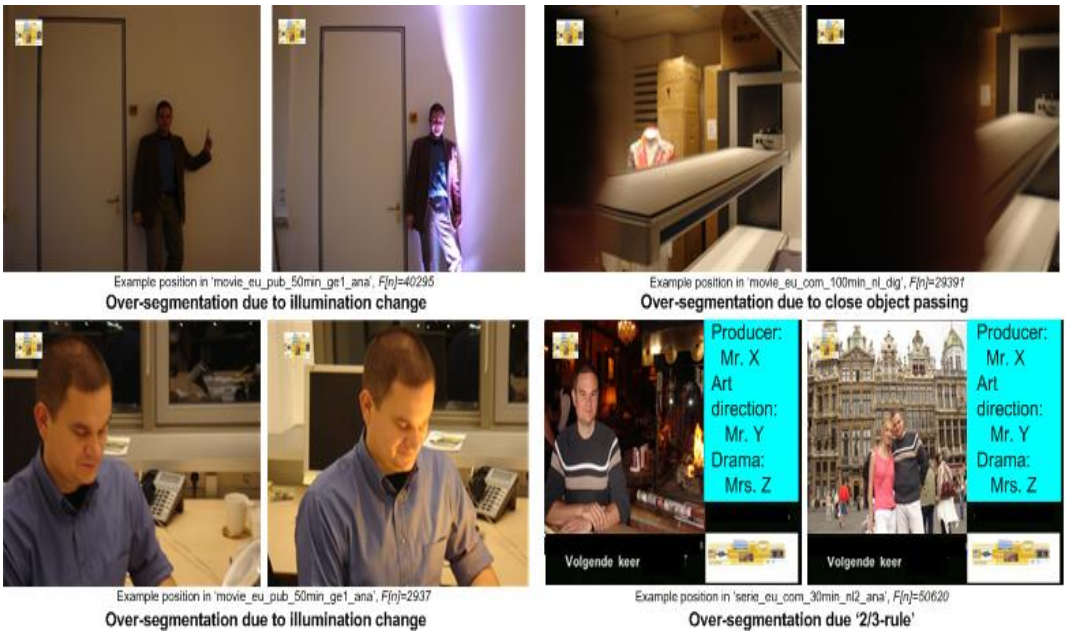


Figure 45. Examples of over-segmentation after feature-point-based post processing[1].

Missed Detections (Recall)

The analysis of the resulting feature-point-post-processing cut detection results, furthermore, unveils that the main reasons for missed cut detections, as stated in Table 10 (right), are for both movies and series instances with high motion in combination with blurry out-of-focus content before/after cut instances, i.e. for movies: for 84% the case and for series: in 60% the case. The field difference cut detector, with its pixel-similarity-analysis-based nature, is not able to cope with such high motion / out-of-focus instances. Representative examples are shown in Figure 46 (top). The spatial similarities of equation (4-16) often exceed the threshold T_{DIS} dis-proportionally, as shown in Figure 46 (bottom). Hence, inter-field dissimilarities *IFD[n]* prior to cut instances display relatively high values, which increase the cut detection threshold, as shown in Figure 35 and Figure 46 (bottom). This, unfortunately, leads to missed cut detections. This non-trivial problem will require additional low–level data, such as spatial texture (e.g. texture co-occurrence and co-variance), which could trigger a switch to an alternative, e.g. global, cut detector during these instances.

Resolution dependency of feature-point-similarity-enhanced field difference cut detector

The pixel-based nature of the field difference cut detector and the feature-point-based post processing are relatively processing intensive and, therefore, a short resolution dependency analysis is performed by us enabling a proper trade-off between reliability and processing costs if required.

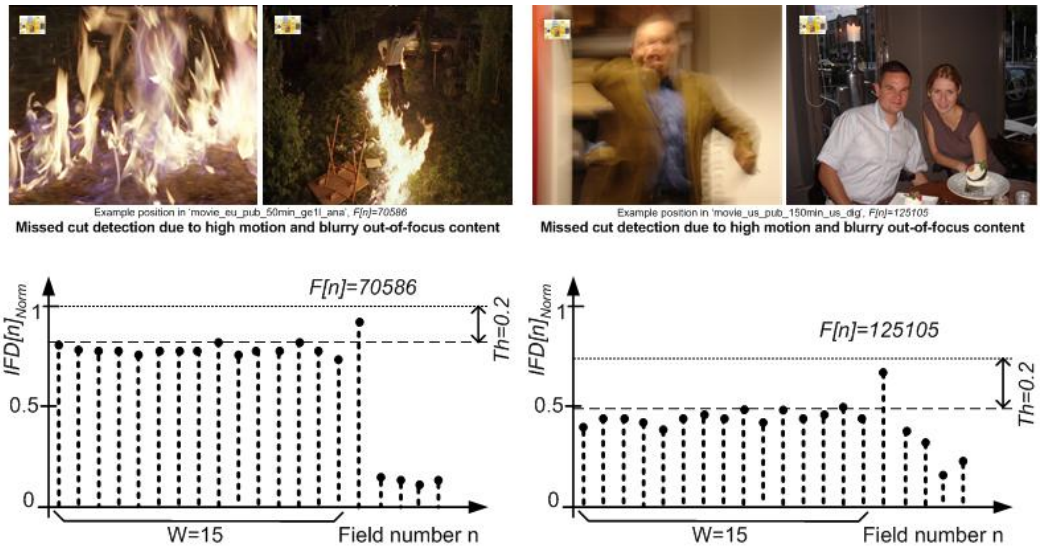


Figure 46. Missed cut detection examples after feature-point-based post processing[1].

Hence, we either simply sub-sample (using only every kth pixel – sub-sampling factor - in x/y direction) or averaged the luminance plane *Y(i,j)* of the frames of our D1 / SD-based content by

$$Block_{AV}(i,j) = \frac{1}{Block\ \mathrm{Re}s(x) * Block\ \mathrm{Re}s(y)} \sum_{m=1}^{\mathrm{Re}s(x)} \sum_{n=1}^{\mathrm{Re}s(y)} Y_{m,n}(i,j) \qquad (4\text{-}27),$$

which represents a block based interpolation similar to DCT-based DC-values of individual macroblocks. Applying cross-validation justifies using the entire data set for this analysis. The following parameter ranges are applied: $W \in \{2..40\}$, $Th \in \{50..450\}$ and $Th_{FP} \in \{0..100\}$, only $W_1=2$ is retained and the results are visualized in Figure 47.

As expected, due to missed cut detections, as shown in Figure 46, and as visible in Table 11, simple sub-sampling slightly outperformed averaging, because the former better preserves the intensity of edges and the latter blurs the content, a situation to which the field difference cut detector and the feature-point-based post processing are ultra sensitive. Hence, the proposed solution would probably work even better with high definition HDTV resolution. With decreasing resolution, as shown in Table 11, Th_{FP} (number of tracked feature points) decreases (D1: 30, HD1: 20, CIF:10) and, finally below CIF resolution, the post processing can no longer be applied, as feature points can no longer be identified. The field difference method, contrariwise, exhibits an impressive robustness against resolution reduction, i.e. a decrease by 1000x results in only ~2% recall/precision loss, as shown in Figure 47.

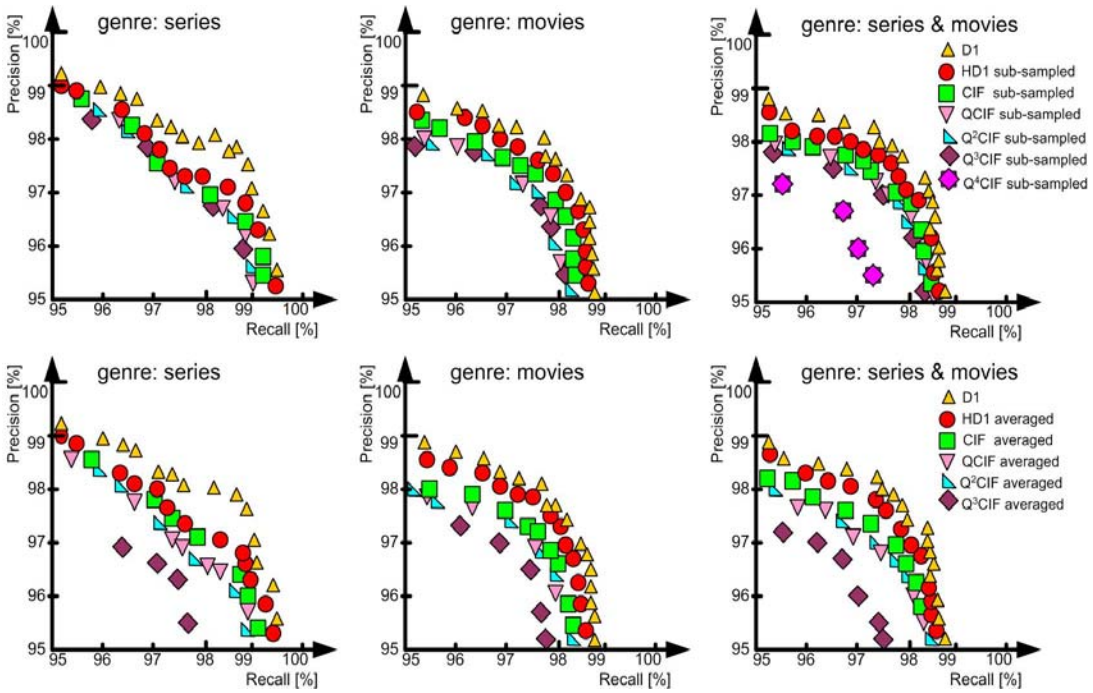


Figure 47. Resolution dependency of shot boundary detector.

Table 11. Cut detector resolution dependency [(Quarter) Common Intermediate Format (Q)CIF].

Resolution	W	Th	Th_{FP}	R	P	W	Th	Th_{FP}	R	P
D1 (720 * 576)	12	0.2	20	98,1	97,7					
	15	**0.2**	**30**	**98,1**	**97,6**					
	18	0.2	30	97,6	98,0					
	Sub-sampled					Averaged				
HD1 (352 * 576)	15	0.2	20	97,9	97,5	15	0.2	20	97,9	97,5
	18	**0.2**	**20**	**97,7**	**98,0**	18	0.2	20	97,6	98,8
CIF (352 * 288)	15	0.25	10	97,4	97,6	15	0.25	10	97,3	97,3
	18	**0.2**	**10**	**97,8**	**97,4**	18	0.2	10	97,7	97,2
QCIF (176 * 144)	**18**	**0.2**	**-**	**97,8**	**97,0**	18	0.2	-	97,6	96,7
Q^2CIF (88 * 72)	**18**	**0.2**	**-**	**97,8**	**97,0**	18	0.25	--	97,5	96,6
Q^3CIF (44 * 36)	**18**	**0.2**	**-**	**97,7**	**96,9**					
Q^4CIF (22 * 18)	**18**	**0.2**	**-**	**96,7**	**96,6**					

Recall enhancement of forward-based field difference cut detector with feature-point-similarity post-processing with backward-based analysis

Here we will consider a case, where the cut detector has to be semi-automatic. That means a cut transition detector, which can be used by a manual annotator, who identifies manually the correct cut instances. We make a further step to improve recall. As stated above (in Table 10), the common reason for missed detections is high motion, and therefore, blurry sequences prior the cut instance, as presented in Figure 46.

Hence a backward-based field difference cut detector is added to the forward approach, as schematically shown in Figure 48. For this purpose we simply reapply the forward cut detector technique, but then from the opposite side, i.e. from the end of a content analysis window towards the beginning. For simplicity reasons we apply for both, the forward and backward approach, the same values for *W* and *Th*.

The analysis on the corpus, applying cross validation, unveils that recall can be increased to *R=98,9%* at the cost of precision *P=97,5%*, as presented in Table 12 and Table 13, with *W=18*, *Th=0.2* and Th_{FP}*=20*. The comparison between forward against forward / backward approach is visualized in Figure 49, showing a clear shift towards higher recall.

The two-side cross-validation requires marginal increase in computational costs and memory buffer, but the results are reason enough to include this approach into our final cut detector.

Table 12. Forward- and backward-based cut detector results.

Resolution	W	Th	Th_{FP}	R	P	
D1 (720 * 576)	15	0.3	20	98,1	98,0	max R/P
	18	0.2	20	98,9	97,5	max R

Table 13. Missed cut detection improvement with backward-based field difference.

	Missed cut detections **before** backward improvement and distribution in % across reasons for failure				Missed cut detections **after** backward improvement			
Genre	Number of instances	Motion & out-of-focus	Short shots w/ moving objects	Others	Number of instances	Motion & out-of-focus (blurry → blurry	Short shots (< 15 frames) w/ moving objects	Others (e.g. 2/3 rule)
Movies	**87**	83%	5%	12%	**37**	85%	5%	10%
Series	**40**	60%	23%	17%	**33**	52%	27%	21%

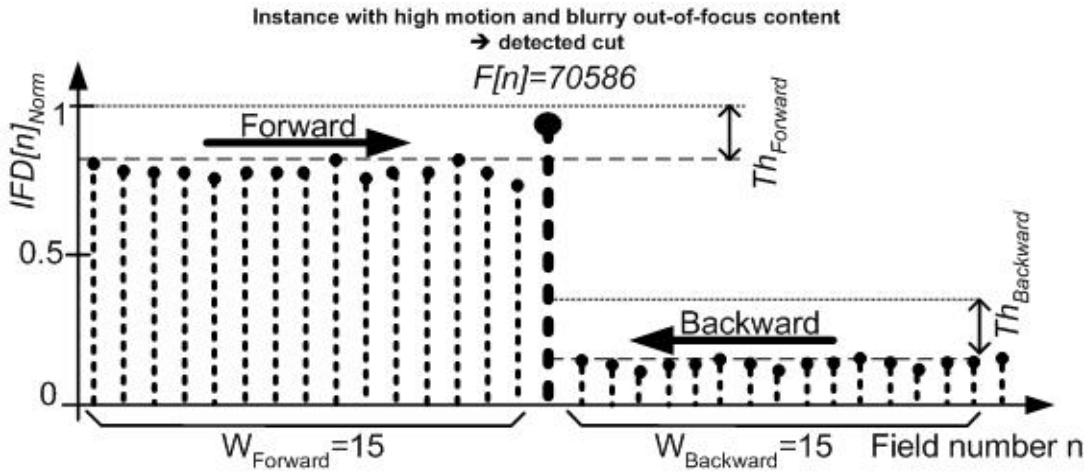


Figure 48. Forward- and backward field difference cut detector.

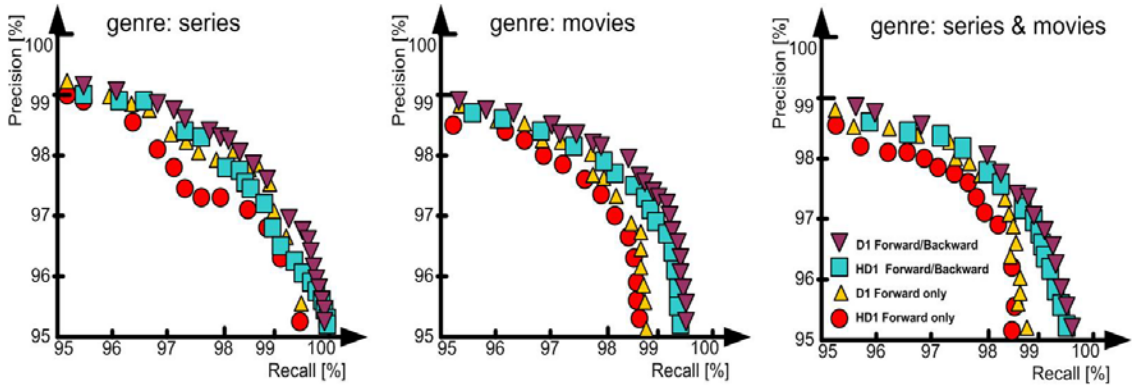


Figure 49. Recall improvement with forward- and backward-based field difference cut detector with feature-point-based post-processing.

Frame based Cut Detector

In order to reduce the computational cost in the industrial system, in the final version of the service unit *cut detector* the field difference method is replaced by a simple frame based cut detector. Hence, the median calculation of missing pixels is replaced by using only pixels of same location in subsequent frames. The *IFD[n]*, applied for the frame based cut detector, hence becomes

$$IFD[n] = \frac{1}{N}\left|\left\{(x,y) \in P[n] : \left|I(x,y,n) - I(x,y,n-1)\right| > T_{dis}\right\}\right| \quad (4\text{-}28),$$

with *P[n]* representing the pixel set with size *N*, containing the spatial positions (i.e. all pixels) in frame *n* (attention: not field), and where T_{dis} is the same preset threshold as applied for the field difference method (see equation (4-17)). Instances, i.e. frames, at which the local *IFD[n]* exceeds the maximum *IFD[n]* value of the past $W \in \{2..20\}$ frames (*W* represents here a selected window length), increased by a chosen threshold value $Th \in \{0.1..0.7\}$, as defined by

$$IFD[n] > IFD[n-m] + Th, \forall m \in \{1,2,3,\cdots,W-1,W\} \quad (4\text{-}29),$$

are marked here as cut instances. Optimal results are achieved, by using the AV benchmark corpus of 0, with *W*=9 frames, *Th=0.2* and Th_{FP}*=30*. The results of the sub-sequent post-processing steps are summarized in Table 14.

The drawbacks of such a simple cut detector are the same as for the field-based cut detector. Its sensibility to fast motion, flashes and object motion is obvious. This is why the same post-processing and cross-validation schemes are of use here.

Table 14. Frame based cut detector with post-processing steps.

Frame based cut detector	W [frames]	Th	Th_{FP}	Correct	False	Missed	Re [%]	Pr [%]
Forward	6	0.2	-	6354	245	169	97.4	96.3
	7	0.2	-	6337	225	186	97.2	96.6
	9	0.2	-	6316	169	207	96.8	97.4
Forward & Backward	6	0.2	-	6450	221	73	98.9	96.7
	7	0.2	-	6437	199	86	98.7	97.0
	7	0.2	-	6269	103	254	96.1	98.4
	9	0.2	-	6414	155	107	98.4	97.6
	15	0.2	-	6285	101	238	96.4	98.4
F&B with FP post-processing	9	0.2	30	6414	108	109	**98.3**	**98.3**

System compliant integration of cut detector into framework

Finally, to integrate the feature-point-enhanced frame difference cut detector in a framework compliant way, two Service Units (see section 2.3) are created, as shown in Figure 50. The forward (backward) frame difference cut detector indexes with zero-frame-delay ($W_{Backward}$-frame-delay, respectively) cut instances and triggers a transfer of two selected key frames at this cut instance to the feature-point-based post-processing

units. At key-frame-similarity instances, i.e. false cut instances, the cut instance index is withdrawn from the metadata output stream, which has now a frame delay of $max(W_1,W_2)$. The MPEG-7 compliant data metadata output streams are summarized in Annex 9.

The results of the cut-detection-based segmentation are shown in Figure 51 for one broadcasted movie, i.e. movie_ge2, which we will apply across this work as reference to show the individual processing results of individual service units. Each vertical bar in the graph of Figure 51 represents one shot of the content item (with an increasing shot number on the x-axis) and the height of each bar represents the duration of the shot (in logarithmic scale on the y-axis). Red bars identify shots, which are delimited at their beginning by a cut, and, on the contrary, blue bars are delimited at their beginning by a gradual transition. We apply this graphical representation as well in our manual annotation tool, which we developed to allow efficient and intuitive manual post-annotations.

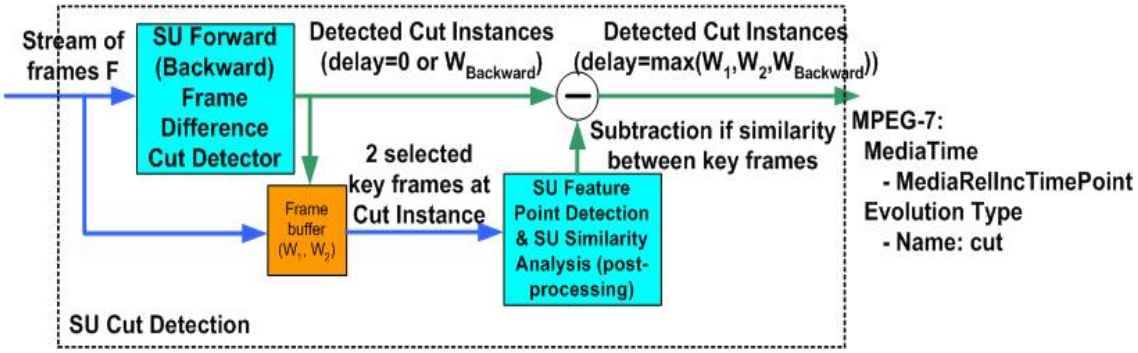


Figure 50. System integration of feature-point-enhanced cut detector.

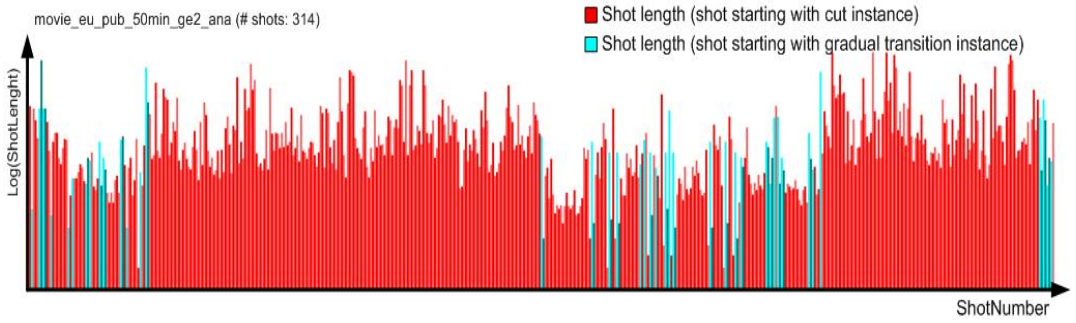


Figure 51. Cut- and gradual-detection-based segmentation of one corpus item (x-axis: shot number; y-axis: shot length in frames in logarithmic scale).

Conclusions for shot boundary detector: cut transition detector

In this section we introduced three newly developed cut detectors, the comparison with a 4th one, the benchmark results on a cultural diverse, multi-genre AV corpus and proposed improvement steps using the most promising cut detector. The results unveil that the simple frame difference method proves to be currently the best trade-off. A subsequent analysis of the detection results unveils further improvement options, which are presented in section 4.1.2, i.e. feature-point-based post-processing and backward cut detection. Finally, we describe in section 4.1.2 a framework compliant integration of the cut detector.

Our simple processing efficient cut detector reached detection results comparable to the highest results achieved in TRECVid with processing-wise expensive methods. Nevertheless, the broad range of algorithms requiring a resilient content segmentation algorithm justifies further investment into the improvement of the current cut detector. Either luminance-, motion- and spatial-texture-adaptive components have to be combined with the field difference detector and feature point analysis, e.g. adaptive SIFT methods, or 3D segmentation in combination with depth analysis could be applied. Especially what concerns the 3D segmentation, the chrominance-only-based AND/OR-consistency-measure (Figure 37) segment analysis of the colour-segmentation-based cut detector, claimed by us in our patent [108], could be enhanced by means of luminance weighting, i.e. additional usage of luminance to identify correctly matching pixels between two temporally-distant segments. We foresee that recall of the colour segmentation cut detector, which performed precision-wise well, will increase and subsequently could be applied as post-processing for the current solution to increase precision. Nevertheless, for the scope of this work the results achieved with the feature-point-enhanced frame difference cut detector fulfil the requirements mentioned in the introduction and, hence, we decided to focus our attention on the other Service Units required to realize our target application.

Shot Boundary Detection – Gradual Transition Detection

The set of professional content production and editing tools at the broadcaster and content producer side not only offer the option to simply concatenate shots together. Those tools also allow to combine shots in a gradual manner by means of e.g. dissolves, fades and wipes as presented in section 3.1.1. Editors use these artistic gradual transitions to create specific situations e.g. to create a flow from a shot of an actor towards a shot of his virtual dreams. Hence, gradual transitions transmit the message of an abstract connection between these two shots. But, as stated by Borecky in [110], ‘Gradual transitions are often used at scene boundaries to emphasize the

change in content of the sequence', and, hence, being a valuable feature for semantic content segmentation. Because of their semantic value gradual transitions are used during production selectively and with caution. The selected corpus of 5 movies and 5 series contained in total 6533 cut transitions, but only 67 ground truth gradual transitions (series: 34, movies: 33). The ground truth gradual transition group contains 35 dissolves, 25 fades and 7 generated, as presented in Table 15 and Table 16. 17 of the 67 gradual transitions instances temporally correlate with ground truth scene boundaries, which supports Borecky's statement in [110] and proves the value of gradual transitions detection for our task. Because the viewer needs time to realize the artistic complex transition, gradual transitions are rather long ranging from 7 to 50 frames (average duration: ~30 frames). Hence, our task's semantic nature stimulates considering as well a gradual transition detector. Nevertheless, we do not propose a new, but apply one available to us from Naci [59], as introduced in section 3.1.1.

Table 15. Ground truth Gradual Transitions in series.

Series	'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total
Dissolves	7	2	0	6	2	17
Fades	0	3	2	7	1	13
Wipes	0	0	0	0	0	0
Computer Generated	1	1	0	2	0	4
Total Gradual Transitions in genre	8	6	2	15	3	34
Gradual Trs at Genre Boundary	0	2	2	4	0	0
Average Duration [frames]	30	31	20	42	29	30

Table 16. Ground truth Gradual Transitions in movies.

Movies	'nl'	'ge1'	'ge2'	'us_com'	'us_pub'	Total
Dissolves	11[8]	5[8]	1	1	0	18
Fades	3	0	2	4	3	12
Wipes	0	0	0	0	0	0
Computer Generated	1	0	1	1	0	3
Total Gradual Transitions in genre	15	5	4	6	3	33
Gradual Trs at Genre Boundary	0	0	4 (Fades)	0	0	0
Average Duration [frames]	12	41	29	64	30	35

Gradual transition detection improvements

The analysis of the detection results of Naci's simple detector [59] unveils that the method is very sensitive to small changes, e.g. compression noise, in areas with almost no motion resulting in F_1~1 (equation (3-9)) due to its adaptive behaviour, which results in a high over detection (352 false detections in total). A detailed analysis of correct, false and missed gradual detections for series and movies is given in Table 17, Table 18 and Table 19.

[8] All gradual transitions are clustered in one short sequence of a photo slide show.

Table 17. Gradual Transition Detection of [59] on series and movies.

Series	**'n1'**	**'nl2'**	**'ge1'**	**'ge2**	**'gb'**	**Total**
Total # of ground truth GradTs in genre series	8	6	2	15	3	34
Correctly detected	3	4	2	4	1	14
False	0	2	0	10	3	15
Missed	5	2	0	11	2	20
Movies	'nl'	'ge1'	'ge2'	'us_com'	'us_pub'	Total
Total # of ground truth GradTs in genre movies	15	5	4	6	3	33
Correctly detected	4	0	4	3	1	12
False	84	43	180	28	2	337
Missed	11	5	0	3	2	21

Table 18. Gradual Transition Detection of [59] on series.

Series	False detections		Missed Detections		
	Static sequence	Others	Gradual high motion	Dissolve static sequence	Computer animations
'nl1'	0	0	4	0	1
'nl2'	1	1	0	1	1
'ge1'	0	0	0	0	0
'ge2'	10	0	4	7	0
'gb'	2	1	2	0	0
Total in series	13	2	10	8	2

Table 19. Gradual Transition Detection of [59] on movies.

Movies	False Detections					Missed Detection	
	Ss & Ml & B with cut transition	Static sequen	Mono lur	Blurry	Others	Gradual high motion	Others
'nl'	10+19+13	13	7	19	3	9	2
'ge1'	17+5	14	7	0	0	5	0
'ge2'	50	128	1	1	0	0	0
'us_com'	2+3	2	18	2	1	1	2
'us_pub'	1	1	0	0		1	1
Total in movies	120	157	34	22	4	16	5

False gradual transition detection in series appeared mainly during static sequences (static pictures, computer animated static sequences, small object moving in static background), as shown in Table 18. Missed detections appeared, according to Table 18, during dissolves between motion loaded colour wise similar sequences or fades to static dark sequences. False gradual transition detection in movies appeared during static sequences with embedded cuts (cut transition embedded by static pictures), static sequences (small object moving in static background, small moving background area with big static object computer animated static sequences, small object moving in static background), monotonous luminance changes (slowly dimmed light) and blurry sequences (fast camera motion, light spot towards camera), as can be seen in Table 19. Missed detections appeared, according to Table 19, mainly during gradual transitions between motion loaded colour wise similar sequences.

120 of these false instances correlate with cut instances identified by our own cut detector of section 4.1.2. Hence, we propose the following post-processing: we discard gradual transition instances if a gradual spans across one of our cut instances. Graduals are simply discarded when the simple TimeStamp (TS) rule,

'CD reduction': $GradualTrT\ S_{start} < CutDetecti\ onTS \leq GradualTrT\ S_{end}$ (4-30),

is satisfied. This reduces the false detection to 232 instances, as stated in Table 20.

Here after, we applied another step to improve the robustness of Naci's method [59], i.e. by post-filtering all remaining gradual transition instances applying a temporal minimal length threshold. Gradual transitions contain a certain semantic meaning and, hence, have to have a certain length, i.e. in average 30 frames, to be perceivable by human observers. Mainly video artefacts triggered the method of [59] to falsely detect very short gradual transitions, which we discard if their length fall-short of an experimentally chosen temporal threshold length, i.e. Th_W=7 frames, resulting in 74 false detections, as summarized in Table 20. Out of the 74 instances 55 instances are embedded in a static sequence and another 17 instances during which the illumination changes smoothly, as shown in Figure 52. Hence, we apply our feature point based similarity post-processing (with Th_{FP}=30), as applied for the cut detector, reducing the false detections to 17, i.e. a precision of 60.5%.

Table 20. Gradual Transition Detection results.

Genre	Total GradTs	Correct GradTs	Missed GTs	False GTs	Recall [%]	Precision [%]
Series - method [59] only	34	14	20	15	41.2	48.3
Movies – method [59] only	33	12	21	337	36.4	3.4
M&S - method [59] only	67	26	41	**352**	38.8	**6.9**
Series after CD reduction	34	14	20	15	41.2	48.3
Movies after CD reduction	33	12	21	217	36.4	5.2
M&S after CD reduction	67	26	41	232	38.8	10.1
Series after windowing (W=7)	34	14	20	15	41.2	48.3
Movies after windowing (W=7)	33	12	21	59	36.4	16.9
M&S after windowing (W=&)	67	26	41	**74**	38.8	26
Series after FP-analysis	34	14	20	1	41.2	93.3
Movies after FP-analysis	33	12	21	16	36.4	42.9
M&S after FP-analysis	67	26	41	**17**	38.8	**60.5**

Figure 52. Examples of gradual transition false detection instances[1].

The analysis of missed detections shows that 26 of the 41 missed gradual transitions occurred during blurry, motion loaded sequences, which decrease the crucial reliability of the motion estimator.

In the end the previous described blocks forms the service unit *Gradual Transition Detector*, as sketched in Figure 53, which communicates its results in an XML-based MPEG-7 compliant way to the subsequent service units, as described in Annex 9.

Conclusions on Gradual Transition Detector

The importance of gradual transitions for semantic segmentation justifies to integrate a state-of-the-art gradual transition detector, i.e. the one from Naci, into the framework and to elaborate accuracy (precision) improvements using derived statistics. Our improvements increased the precision to an acceptable level of 60.5% containing false detections mainly based on smooth illumination changes. We expect that the latter could probably be eliminated by checking each gradual transistion instance applying our segmentation map based consistency method, presented in our shot detection section.

We also did not tackle the challenging question of increasing recall. Mainly, because the detection of the majority of the remaining 41% missed detections, i.e. 26 individual instances, which are embedded in blurry, motion loaded sequences would require additional features such as more advanced foreground / background segmentation solutions enabling better camera motion predictions and texture to define the blurriness.[9]

To avoid that the accuracy achieved with the gradual transition detector influences subsequent analysis results we consider applying gradual transition ground truth as input for the subsequent service units and, hence, to separate the results from each other.

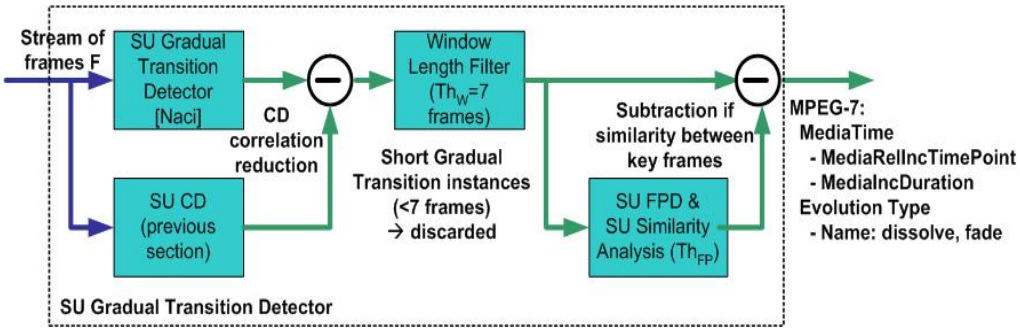


Figure 53. System integration of gradual transition detector.

[9] There are some recent works in TRECVid dealing with luminance changes applying linear regression, which could be of use for this purpose.

Camera Motion Analysis

Literature differentiates between two main camera motion analyses models, as described in section 3.1.2, a feature- and the intensity-based one. For the purpose of audiovisual segmentation analysis an intensity-based camera motion model, derived from [111], is implemented as service unit into the framework with a 2-parameter translation estimation $\underline{d}=(d_x,d_y)$, which will be shortly described in this section. First of all, the camera motion algorithm relies on a sequence of related video frames, i.e. frames within a shot, e.g. *F(n)* and *F(n+d)*. Hence, shot boundary information, here from the shot boundary detection service unit, is provided as input to the camera motion analysis unit used to reset the camera motion parameters $d=(d_x=0,d_y=0)$ at each shot boundary securing small motion parameter values. The latter are usually in the range of ± 20 pixel / frame for both directions, *x* and *y*. Unfortunately, we do not have a reliable foreground / background differentiating algorithm to our disposal, which could help to reduce the motion noise introduced by foreground object motion as described in section 3.1.2.

Camera motion analysis

First and foremost, a set of N_{SP} uniformly distributed sample points, here a simple uniform grid is applied, extracted from the Y-plane, i.e. the intensity out of YCbCr, of the initial frame *F(n)* and the successor frame *F(n+1)*, as described in [111], resulting in N_{SP} samples $Y_{F(n)}(x_i,y_i)$ and N_{SP} samples $Y_{F(n+1)}(x_i,y_i)$. Subsequently, only for frame *F(n)* the x- and y-gradients (derivatives) are calculated per sample point, by means of a simple "–1;0;1" mask, i.e. for each sample $Y_{F(n)}(x_i,y_i)$ the horizontal predecessor sample $Y_{F(n)}(x_{i-1},y_i)$ is subtracted from its successor $Y_{F(n)}(x_{i+1},y_i)$, as well as for the vertical direction ($Y_{F(n)}(x_i,y_{i+1})$- $Y_{F(n)}(x_i,y_{i-1})$). This results in an $N_{SP}x2$ matrix *M* (consisting of a x- and a y-gradient per sample point).

Now a matching procedure, including several iterations, starts to estimate the *2x1* camera motion vector $\underline{d}=(d_x,d_y)$ minimizing the $N_{SP}x1$ error vector $\underline{e}$ of equation

$$M\Delta\underline{d} = \underline{e} \tag{4-31},$$

with $\underline{d}_{New}=\Delta\underline{d}+\underline{d}_{Old}$, wherein $\Delta\underline{d}=(\Delta d_x,\Delta d_y)$ represents the update vector of $\underline{d}$. The elements of $\underline{e}$ are the luminance difference values between the sample points $Y_{F(n)}(x_i,y_i)$ and the motion compensated sample points of $Y_{F(n+1)}(x_i,y_i)$, as defined by

$$e = \begin{bmatrix} Y_{F(n)}(x_1, y_1) - Y_{F(n+1)}(x_1 + d_x, y_1 + d_y) \\ \vdots \\ Y_{F(n)}(x_{N_{SP}}, y_{N_{SP}}) - Y_{F(n+1)}(x_{N_{SP}} + d_x, y_{N_{SP}} + d_y) \end{bmatrix} \tag{4-32},$$

with an initial *d=(0;0)*. Since *M* represents an over-specified system (many more rows than columns) the equation may not have a solution and, therefore, a least-square calculation is required using *M'*, i.e. the transpose of *M*, converting the equation into

$$M'M\Delta\underline{d} = M'\underline{e} \quad (4\text{-}33),$$

wherein *M'M* represents a *2x2* matrix and *M'e* a two-element vector, resulting in

$$\Delta d = \left(M'M\right)^{-1} M'\underline{e} = P\underline{e} \quad (4\text{-}34),$$

with *P* representing the pseudo-inverse of matrix *M*. At the end of each iteration the motion vector $\underline{d_{New}}=\Delta\underline{d}+\underline{d_{Old}}$ is updated, which represents the end of an iteration. Minimizing $\Delta\underline{d}$, i.e. $\Delta\underline{d}\sim 0$, leads, therefore, to an optimal camera motion estimation, which is done running the previous calculation for several iterations. In test eight iterations have performed best and has been used subsequently. Finally, the resulting $\underline{d_{New}}=(d_x,d_y)$ is provided as output of this service unit per frame instance.

Conclusion on Video mid-level features

In this section we have introduced firstly our approach to construct an AV corpus for objective development of content analysis algorithms. The latter we applied to benchmark three of our cut detectors, i.e. macroblock correlation CD, field difference CD and color segmentation CD, against each and against a rough indexing CD. The field difference CD performed best and, hence, we improved the robustness by feature-point-based similarity analysis post-processing and a forward-backward-based filtering approach. Furthermore, we investigated the robustness of the detector by decreasing the resolution and replacing the field difference CD by a frame difference CD. The latter we integrated as our cut detector of choice into our framework as Service Unit CD.

Hereafter, we improved an inherited gradual transition detector. The latter we implement as Service Unit GDT into our analysis framework.

Finally, we elaborated a camera motion analysis algorithm, which we implemented as Service Unit CM.

4.1.3 Conclusions on video low-level and mid-level feature

In this section we presented several low-level and mid-level analysis algorithms, which we consider useful and necessary to elaborate a system solution allowing us to segment content automatically into its semantic entities.

Because we intend to apply low-level feature extraction in a processing constraint environment we constraint ourselves in this section to compressed domain solutions. Nevertheless, the concepts introduced can be applied as well in the baseband domain if required.

For our task of content segmentation we identified as well several mid-level video analysis features, such as cut detection, gradual transition detection and camera motion. Our specific robustness and platform requirements forced us to develop own solutions for these mid-level features, which we met as e.g. presented for cut detection and gradual transition detection.

In the next section we will introduce several audio-based low-level and mid-level analysis features, which we consider as necessary to accomplish our task.

4.2 Task-oriented low-level and mid-level audio analysis

In this section we will investigate low-level and mid-level features, which we aim to apply for audio-based segmentation and for a dedicated content classification task, i.e. commercial block detection.

The promising results achieved by fusing audio- and video features for audiovisual content classification and segmentation, as described in sections 3.2 and 3.3, motivated us to elaborate specific audio low-level and mid-level features. In this section we present various low-level and mid-level audio analysis algorithms starting with a compressed domain silence detector. We develop this application-oriented silence detector specifically for our commercial block detector at which we merge audio silences and several video features. Here after, we present in more detail our inherited audio classifier, i.e. the classifier of McKinney [74], which we introduced in section 3.2.2 and aim to apply as pre-processing for audio-based segmentation.

4.2.1 Commercial block silence detection

The equivalent to shot boundaries in video are in the audio domain silences. The latter allow segmenting an audio signal content into its elementary audio units. Especially in message loaded content such as commercial adds, individual messages (i.e. individual adds) have to be audio-visually observably disconnected. Dedicated audio silences, further referenced as commercial block silences, exhibit this distinctive behaviour and due to their specific observable nature we aim to identify them for indexing non-content related inserts, i.e. commercials.

Compressed domain commercial block silence detection

Our analysis on commercial blocks embedded in our AV corpus content, described in section 4.1.2, unveils that silences separating individual commercial adds from each other propagate three specific characteristic and distinctive attributes. These three attributes, as published in our patent application [33], are (a) a steep short-term signal power decreases and (b) a long duration, i.e. in the range of 0.1-1.0 seconds and (c) a distinct local signal power in relation to the global averaged signal power level, i.e. distinctive local minima. This is why we propose a method, which is based on a simple thresholding. The sub-task's specific technical requirements, i.e. processing efficient implementation of a commercial block detector, are the reason that we focus on two compressed domain audio parameters in parallel, i.e. MPEG-1 layer 2 (scale factors) and AC-3 (exponents), to realize this specific silence detector.

MPEG-1 layer 2

The core of an MPEG-1 layer 2 codec [114], which covers a bit range from 8 to 384 kbps, samples an incoming pulse code modulated *PCM* audio signal at frequencies of 48 kHz, 44.1 kHz or 32 kHz. Here after, it decomposes the signal into individual equidistant audio frames with distances of 24, 26.12 or 36 msec, respectively. Each audio frame is sub-divided into three time-wise consecutive blocks *BL*, i.e. BL_1 to BL_3, consisting of 32 frequency-wise equidistant narrow sub-bands SB_{Ml}, i.e. SB_{M0} to SB_{M31}, and containing two channels *Ch*, i.e. left channel CH_L and right channel CH_R, as shown in Figure 54 (left). This results in 3*32*2=192 sub-band units per audio frame and each of them contains 12 quantized samples, which is represented by one corresponding intermediate factor. The latter one corresponds to the upper-bound estimate of the 12 sample values. The resulting intermediate factor is, here after, clipped to one of the 59 MPEG-1 layer 2 defined scale factor values, i.e. 0 to 58. Hence, each sub-band unit, as shown in Figure 54, is represented by its value and one scale factor. This results in *sfv=32*3*2=192 scale factor values* (sfv) per audio frame and each scale factor value represents a pseudo dB representation.

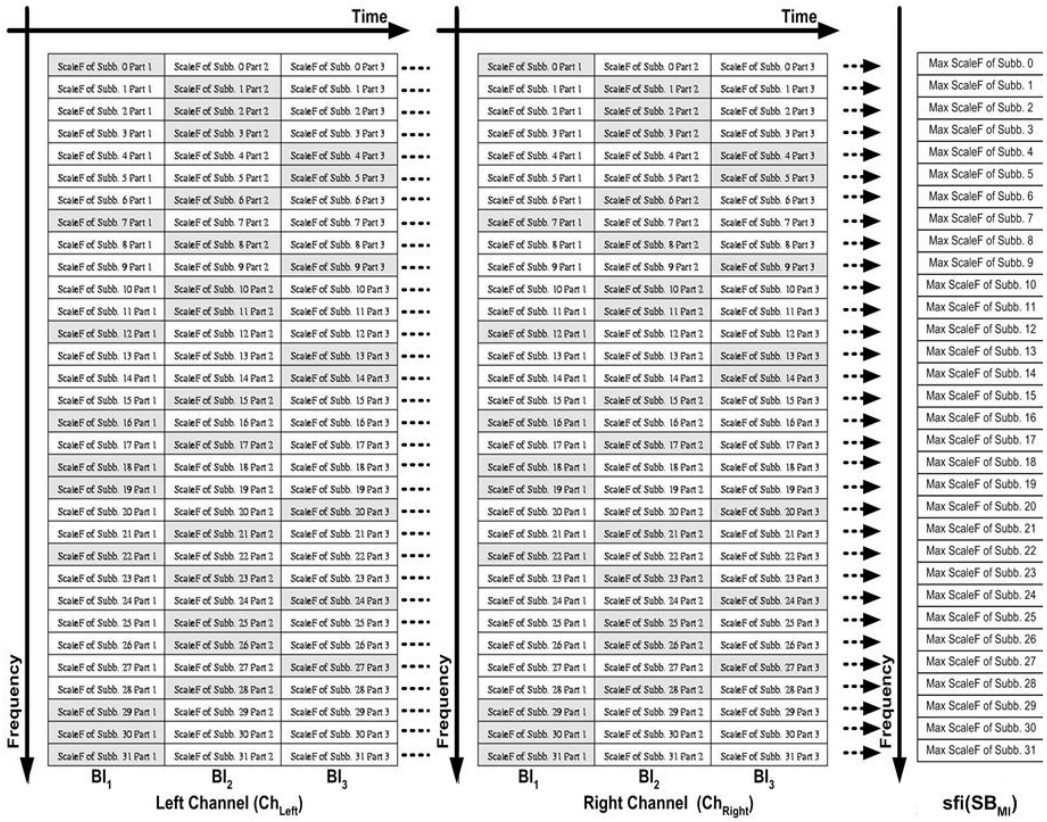


Figure 54. MPEG1 layer 2 audio frame with maximum scale factor selection.

Furthermore, the specific task of computing the maximal signal power allows compressing the available data of each audio frame even further. It is sufficient to keep, for each sub-band, only the maximum scale factor value, as shown in Figure 54 (right). Hence, for each of the 32 triplets and two channels only the maximum *scale factor index* value per sub-band *sfi(SB$_{Ml}$)* is needed with

$$\mathrm{sfi}(\mathrm{SB}_{\mathrm{Ml}}) = \max\begin{Bmatrix} sfi_{\mathrm{Ml}\,BL=1,CH=L}, sfi_{\mathrm{Ml}\,BL=2,CH=L}, sfi_{\mathrm{Ml}\,BL=3,CH=L}, \\ sfi_{\mathrm{Ml}\,BL=1,CH=R}, sfi_{\mathrm{Ml}\,BL=2,CH=R}, sfi_{\mathrm{Ml}\,BL=3,CH=R} \end{Bmatrix} \quad (4\text{-}35).$$

This results in 32 scale factor values per audio frame.

AC-3 (Dolby)

For AC-3 we take an equivalent approach to MPEG-1 layer 2. In AC-3 [115], a compression developed by company Dolby with a bit rate ranging from 32 to 640 kbps, the audio signal is characterized by 21 exponents, i.e. *exp$_0$* to *exp$_{20}$*, corresponding to a signal power ranger of 0 to –120 dB. The sampling rate here is 33.25 kHz. Here after, each resulting audio frame is sub-divided into six time-wise consecutive blocks *BL*, i.e. *BL$_1$* to *BL$_6$*, consists of 253 frequency-wise equidistant narrow frequency sub-bands *SB$_{Ml}$*, i.e. *SB$_{A0}$* to *SB$_{A253}$*, and contains two channels *Ch*, i.e. left channel *CH$_L$* and right channel *CH$_R$*. This results in 6*253*2=3036 sub-band units per audio frame and each sub-band unit is represented by a mantissa and exponent product, i.e. mantissa*2$^{(-\text{exponent})}$. Equivalent to the compression done in the MPEG-1 layer 2 case, we select, per sub-band, one maximum exponent value across all blocks and channels, which results in 253 exponent values per audio frame. This is still of unnecessary fine granularity, hence, we compute the average across eight consecutive narrow sub-bands, i.e. sub-band clustering, which results in *L=upperbound((253-5)/8)=32* maximal exponent values *exp$_{N,i}$*, wherein the last cluster contains only 5 sub-bands.

Conversion from MPEG-1 layer 2 scale factors to AC-3 exponents

At this stage we have with MPEG-1 layer 2 32 maximum pseudo scale factor values and with AC-3 32 maximum exponent values. In order to continue with one common solution for both we map the scale factor values onto AC-3 exponent values. MPEG-1 layer 2 applies step sizes of –2 dB per scale factor with the highest scale factor value, i.e. 58, corresponding to -116 dB, as shown in Table 21. In AC-3 21 exponents are applied with the highest exponent, i.e. 21, also corresponding to –116 dB. The step size is, therefore, in AC-3 -5,8 dB. With Table 22 we convert the scale factors into exponent values *exp$_{N,i.}$*

Table 21. Look-up table for MPEG-1 scale factor to AC-3 exponent conversion.

AC-3			MPEG-1 layer 2		AC-3			MPEG-1 layer 2	
exp	dB		dB	scale factor	exp	dB		dB	scale factor
0	0	←	0	0 (1,2)	11	- 63,8	←	- 64	32 (33,34)
1	- 5,8	←	-6	3 (4,5)	12	- 69,8	←	- 70	35 (36,37)
2	- 11,6	←	- 12	6 (7,8)	13	- 75,4	←	- 76	38 (39,40)
3	- 17,4	←	- 18	9 (10,11)	14	- 81,2	←	- 82	41 (42)
4	- 23,2	←	- 24	12 (13)	15	- 87	←	- 86	43 (44,45)
5	- 29	←	- 28	14 (15,16)	16	- 92,8	←	- 92	46 (47,48)
6	- 34,8	←	- 34	17 (18,19)	17	- 98,6	←	- 98	49 (50,51)
7	- 40,6	←	- 40	20 (21,22)	18	- 104,4	←	- 104	52 (53,54)
8	- 46,4	←	-46	23 (24,25)	19	- 110,2	←	- 110	55 (56,57)
9	- 52,2	←	- 52	26 (27,28)	20	- 116	←	-116	58
10	- 58	←	- 58	29 (30,31)			←		

Table 22. Mapping of MPEG-1 layer 2 scale factor to AC-3 exponents.

Scale factor	Exponent	Scale factor	Exponent	Scale factor	Exponent
0,1,2	0	3,4,5	1	6,7,8	2
9,10,11	3	12,13	4	14,15,16	5
17,18,19	6	20,21,22	7	23,24,25	8
26,27,28	9	29,30,31	10	32,33,34	11
35,36,37	12	38,39,40	13	41,42	14
43,44,45	15	46,47,48	16	49,50,51	17
52,53,54	18	55,56,57	19	58	20

Silence detection with relative ratio of signal strength – compressed domain

Hence, from the previous step we derived the maximal exponent values $exp_{N,i}$ of the clustered sub-bands, i.e. *L=32* per audio frame. The latter we apply to calculate the local signal strength *S(N)* of audio frame *N*. Here fore, we sum across all 32 clustered sub-bands with

$$S(N) = \sum_{i=0}^{L} \left(2^{-\exp_{N,i}}\right)^2 \quad \text{(4-36)},$$

wherein $exp_{N,I}$ denotes the i-th exponent. Here after, we calculate the average signal strength $S_A(N)$ by averaging across all *S(N)* values of all audio frames located within a sliding time window of size *W* of consecutive audio frames. Experiments prove that logarithmic values of *S(N)* pose good discriminative power with respect to silence detection. We compute $S_A(N)$, therefore, with a delay of *W/2* frames, with

$$S_A(N) = 2^{\frac{1}{W}\sum_{i=0}^{W-1}\log_2\left(S\left(N+\frac{W-i}{2}\right)\right)} \quad \text{(4-37)}.$$

Each audio frame instance is then evaluated by applying a simple threshold based rule with threshold *Th*, i.e.

$$\frac{S(N)}{S_A(N)} \leq Th \quad \text{(4-38)},$$

and instances satisfying this equation are indexed as potential cut silence instances.

Hereafter, we cluster consecutive silence instances together and close short outliers, i.e. closing gaps by morphing. The latter instances are defined by a signal strength $S_{outlier}$, which falls short of the product of a fixed factor M and the signal power of a neighboring silence instances S_{NCS}, i.e.

$$S_{outlier} = M * S_{NCS} \quad (4\text{-}39).$$

Our method for silence detection in the compressed domain can be easily applied as well in the baseband domain as claimed by us in [33] and [114] and published by Louis and us in [37] and [36], respectively. In the context of this work we also consider to combine the baseband silence detection method for scene boundary detection.

Commercial block silence detection parameter

The specific distinctive nature of commercial silences allows excluding the signal strength of higher frequency sub-band clusters, i.e. SB_{15} to SB_{31}, because of their neglectable contribution to the silence detector. Hence, we apply only the lower frequency sub-band clusters, i.e. SB_0 to SB_{14}, without scarifying detection robustness. The analysis of the detection results unveils limited over-detection, i.e. false detection, during speech silences. An analysis of instances of the latter showed that these sequences exhibit compared to non-speech sequences a significantly higher variance of $S_A(N)$ within windows larger than 5 seconds, which we applied to exclude speech sequences. Our experiments provided optimal results with an empirically chosen window of W=*256* audio frames, i.e 8.2 seconds for AC-3 at an audio sampling rate of 31,25 Hz, a threshold value Th=1/512*(2^{-9}) and a 'morphing' factor of M=8.

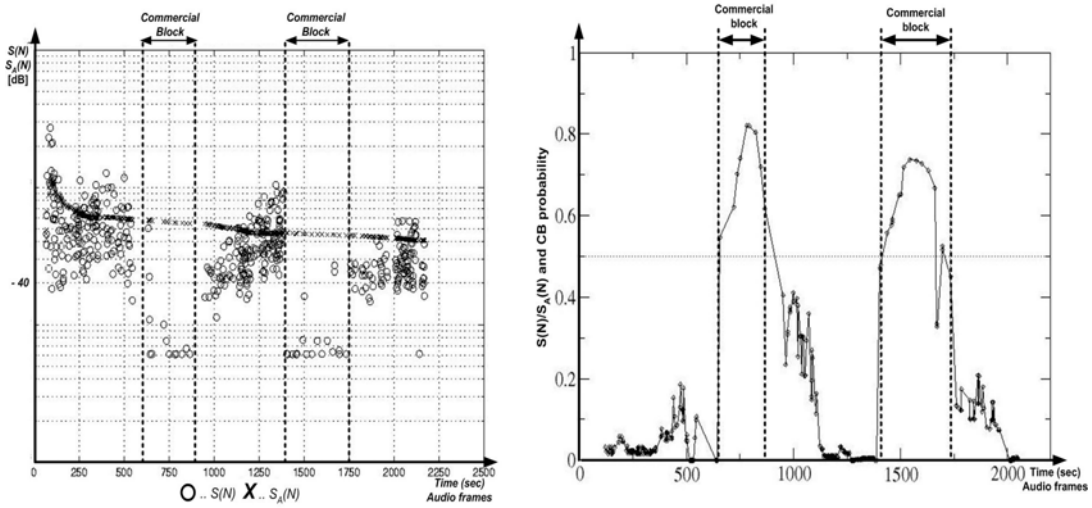


Figure 55. Left: *S(N)* and $S_A(N)$; Right: CB propability with *S(N)/SA(N)* and duration.

In Figure 55 (left) we show the results the local signal strength *S(N)* and the average signal strength $S_A(N)$ on a AV corpus movie sequences containing two embedded commercial blocks. $S_A(N)$ adapts quickly the audio signal and *S(N)* instances cluster themselves close to $S_A(N)$ during movie sequences, but at commercials containing characteristically deep low-signal-strength silences the $S(N)/S_A(N)$ ratio exhibits significant changes. Hence, we index all audio instances, which fulfil our $S(N)/S_A(N)$ ratio based equation (4-38) as potential commercial block cut silences, fill gaps if the instance fulfil our morphing condition of equation (4-39) and delete all silence instances with a duration shorter than two audio frames. Subsequently, we apply the commercial block cut silence frequency to derive a Commercial Block Probability, as visualized in Figure 55 (right), which we only calculate to check the strength of the cut silence detector but do not use any further. What we aim for is to fuse our commercial cut silence detector with visual commercial block detector features.

System integration of service units commercial cut silence detection

In the final setup we integrate the service unit commercial cut silence detection into our framework with an MPEG-7 compliant output. The latter is specified in Annex 9. For the required synchronization with the video stream we replace the audio media time of the commercial cut silence detector output with the appropriately video media time, as shown in Figure 56.

Conclusions

Our analysis proved that the audio signal of commercial blocks contain, as published by Marlow in [90], valuable distinctive information, i.e. characteristic deep and long-lasting silences.

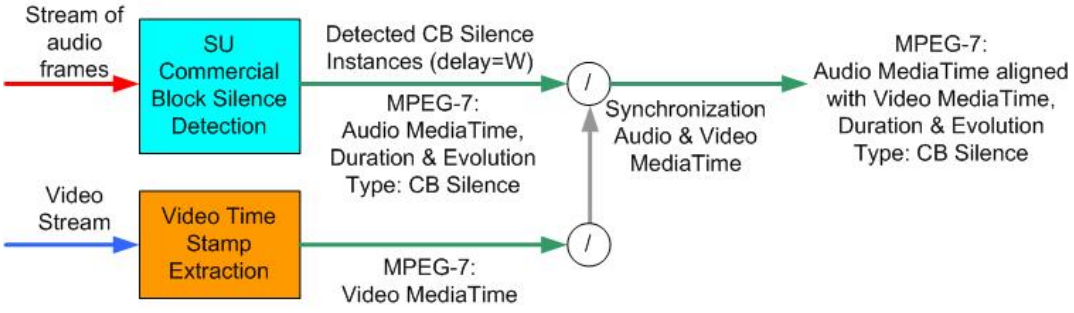


Figure 56. System integration of commercial block silence detector.

The technical requirements of our task, i.e. processing inexpensive and real-time, stimulated us to realize a new commercial silence detector, which exploits features of compressed domain audio, i.e. MPEG-1 layer 2 and AC-3. From the latter we applied scale factors and exponents, respectively, to compute local and average signal power, which in relation, i.e. $S(N)/S_A(N)$, provide a good commercial silence indicator. The latter representss a strong feature for commercial block classification, as we aim to develop in section 4.4.

4.2.2 Audio classifier

For scene segmentation we consider fusing visual segmentation with audio scene segmentation, as proposed already in [77],[78] and [80]. Similar to the Nitanda's method [79], we intend to apply audio segmentation as preprocessing step for audio segmentation. Here fore, we consider applying McKinney's audio feature extractor [74] and classifier. The feature extraction unit is based on a selected group of distinctive audio features, i.e. (a) low-level signal properties, (b) *mel-frequency spectral coefficients* (MFCC), (c) psychoacoustic features including roughness, loudness and sharpness, and (d) an auditory model representation of temporal envelope fluctuations. McKinney's classification unit applies these features to classify the audio signal into six independent class probabilities, i.e. class independent probability values between zero and one, for six audio classes, i.e. *speech*, *music*, *noise*, *crowd*, *silence* and *unknown*, as shown in Figure 57 for a short AV corpus movie content for various classes in various colors.

4.2.3 Conclusions on task-oriented low- and mid-level audio analysis

In this section we introduced two task-oriented audio analysis mid-level features, i.e. commercial block silence detection and audio classification, which we consider to fuse with video analysis features for content classification, i.e. commercial block detection, and content segmentation, respectively.

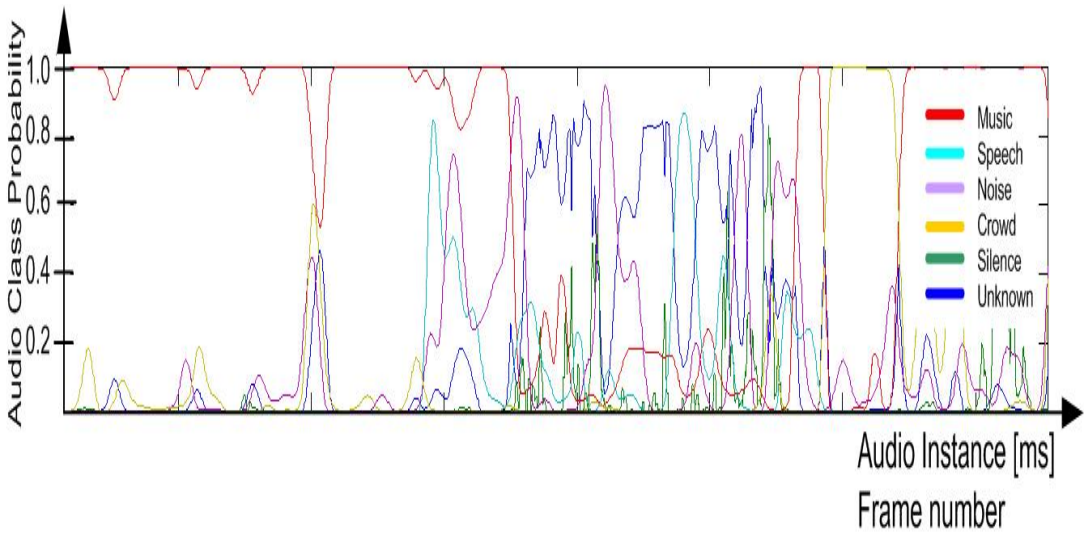


Figure 57. Audio class probability results obtained with McKinney's audio classifier.

4.3 Generic High-Level AV Segmentation

So far in this work, we segmented content items up into their elementary video-related units, i.e. shots, as presented in section 4.1.2, and into its audio entities as explained in section 4.2. The next step is to cluster these elementary units, i.e. shots, which are comparable to words in a spoken language, into bigger meaningful entities such as audiovisual chapters, which could be seen as entire spoken sentences each having a defined structure and semantic meaning. Fortunately, audiovisual productions follow well-specified (production) rules, comparable to a human language being defined by a specific language grammar. In the same way as human languages are based on different grammars and rules, also the audiovisual content genres have their genre-specific rules, which an analyzer has to be aware of before trying to interpret the audiovisual content.

Hence, in a first step it is important to identify and filter-out non-content-related entities (non-genre related contents), which were incorporated into the content, i.e. commercial-block- and channel-advertisement inserts. This is a necessary step, because the grammar of the latter is very uncorrelated with the grammar of the narrative content it is embedded in. The author developed, therefore, a specific filter for commercial block inserts, further described in the next section, permitting a more focused approach.

Knowing that non-content related inserts are indexed as such, one can assume that the remaining content parts follow some genre specific production rules also known as *film grammar*, as published by Beaver [15] and Bordwell [16], which will be elucidated in detail in section 4.5. The aspired quality of the final segmentation solution and the variety of genre specific rules compels to narrow the genre space. Hence, we consider narrowing the solution to two narrative genres, i.e. series and movies. Fortunately, both narrative-based genres contain specific film-grammar-based clusters, i.e. *Cross-cuttings* and *Shot-Reverse-Shots*, further explained in section 4.5.1. The latter could be seen as 'subordinate clauses' when using the analogy to language grammar. In section 0 we intent to present various methods, which facilitate parallel shot detection through which more than half of the content can be clustered together into meaningful audiovisual segments.

The remaining small non-clustered part of the content is the input for the final step, which is to segment the content into its semantic meaningful audiovisual chapters. We present various clustering and segmentation methods and combinations thereof in the remaining sections of 4.5 and 4.6, aiming to detect automatically the boundaries of audiovisual scenes (chapters).

4.4 Audiovisual Content Filtering: Commercial Block Detection

Robust AV segmentation, as described in the previous sections, and the related automatic abstraction of video information from e.g. TV broadcast requests for robust methods to distinguish between program related and non-program-related content. This is specifically needed as broadcast stations insert non-content-related elements such as commercial advertisements due to their specific business model. Because of its dissentient nature, these non-content-related inserts have to be identified in a pre-processing step and they have to be excluded from the further AV segmentation process to secure the reliability of the segmentation solution.

These non-content related inserts mainly consist of individual *commercial clips* CC. The latter aim conveying in a very short time a distinct and recognizable marketing message of a product or a service. Their content is, therefore, heavily loaded. It consists of very short shots, high motion, appealing color distribution, strong embedded text messages, persons and objects in focus, dominant speech/music sequences, and the absence of channel logos. Furthermore, commercial clips distinct themselves, by intention, from the rest of the content. Hence, intuitive and recognisable audiovisual *commercial clips separators* CCS enable viewers to distinguish between individual commercial clips, as shown in Figure 58. Several commercial clips are clustered into *commercial blocks* CB. The latter are often flanked on either side by *commercial block identifiers*, which are defined by legislation to index the beginning and end of a commercial block, and / or *trailers*. The latter are often channel related announcements and promotions of upcoming events or programs, schematically shown in Figure 58. Sometimes the latter can also appear independently of the commercial blocks. The specific nature of commercial blocks distinguishes them from other genres.

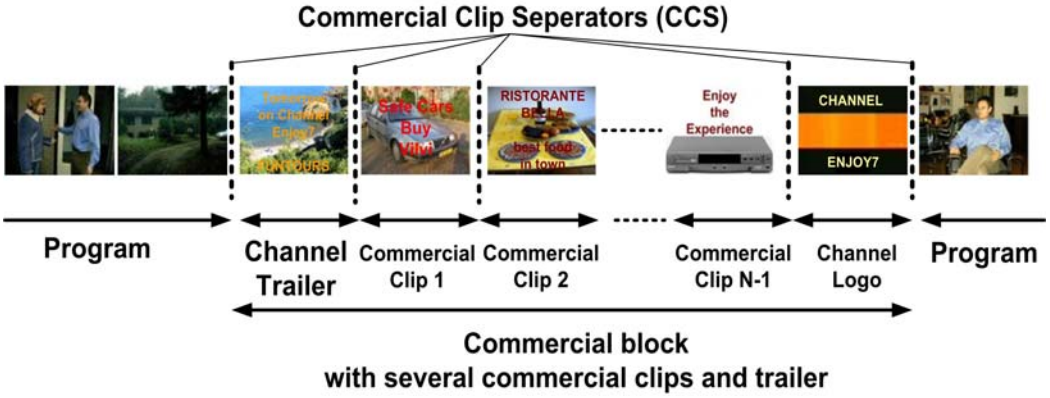


Figure 58. Commercial block with commercial clips embedded in a program content[1].

4.4.1 Commercial block properties

In section 3.3.1 various published methods have been presented exploiting the specific nature of commercials to detect them, such as the fact that individual commercial clips of a commercial block have to be distinguishable from each other. Hence, an intuitive discontinuation between successive clips is required realized by an audible and visible disruption of the audiovisual signal. A common approach of broadcast stations in the past was applying audio silences black frames here for. From chapter 3 we know that especially the latter feature was used by e.g. Blum [87] and Iggulden [88] in combination with specific temporal thresholds, i.e. the time-wise distance between successive black frames, in order to identify commercial advertisement sequences. The need to convey an important marketing message in an appealing way in about 30 seconds, i.e. the average duration of a commercial clip, forces producers of commercials to apply an unusual high video cut frequency transmitting memorize-able appealing content. An example of the latter can be seen in Figure 59 for a news-culture-weather block, which is embedded within several commercial blocks. The short shot duration in combination with colorful, sharp and focused content provide distinctive video cues not only to increase the attractiveness of commercial clips, but at the same time to automatically identify them, e.g. by exploiting the high variance of $Y_{DV\_AV\_Norm}$, as shown in Figure 59 (up left).

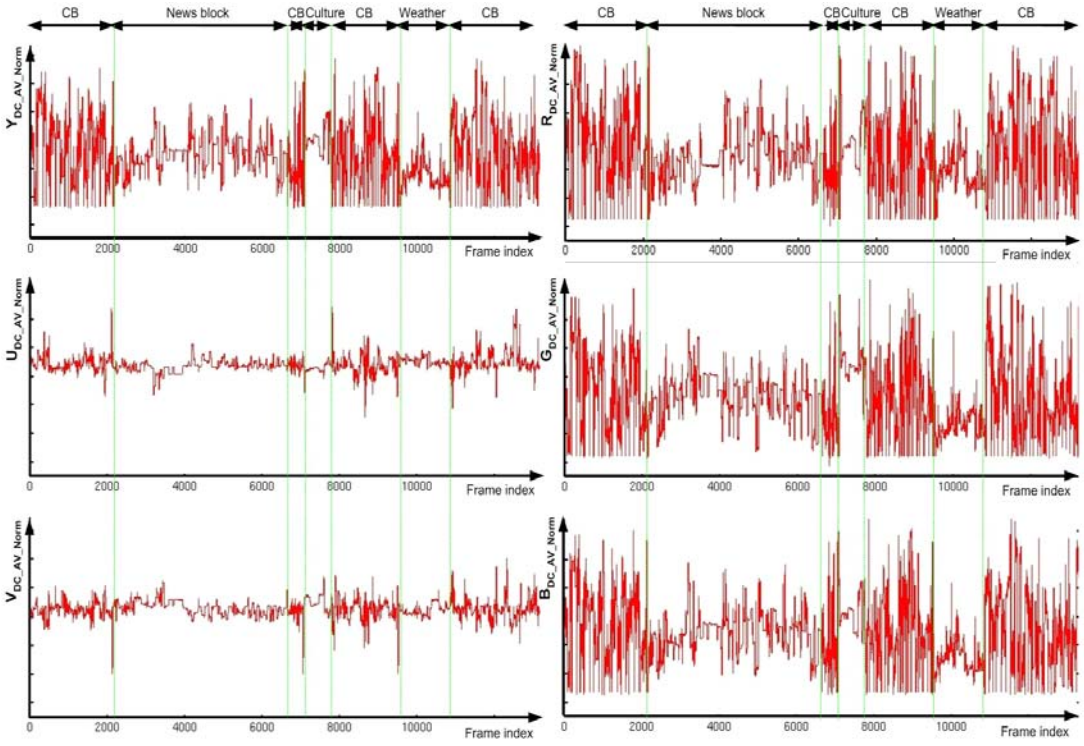


Figure 59. Behaviour of average YUV and RGB values during commercial blocks CB.

As presented in chapter 3, Lienhart exploited in [89] the high cut frequency in combination with the frequent re-appearance of black frames to detect advertisements. We aim in this section combining our application-oriented commercial cut silence detector from section 4.2.1 with several distinctive video features, similar to Marlow's method [90] but then extended with concepts from Iggulden [88] and Lienhart [89].

Scarce computational processing resources in the target consumer platforms and the need for a near real-time performing commercial block detector forced us narrowing our search mainly to the compressed domain of video. Luckily, MPEG-2 codecs, i.e. an encoder-decoder, offer internally a variety of compression parameters, which we reuse to calculate our specific low-level and mid-level features, as published by us in [121]. Hence, we implement several compressed domain video low-level features, which we introduced in section 4.1.1, i.e.

- normalized complexity COM_{Norm} of equation (4-3),
- progressive detector $SUM_{Prog/Inter}(N)$ of (4-5),
- black frame detector and mono-chrome frame detector with $Y_{DC\_AV\_Norm}(N)$ and $Y_{DC\_VAR\_Norm}(N)$ of (4-6)/(4-9),
- letterbox detector $Y_{DC\_VAR\_LB\_Norm}(N)$ of (4-13) and the video mid-level feature,
- shot boundary detector of section 0 with MAD,

and try applying them to identify commercial blocks, as shown in Figure 60.

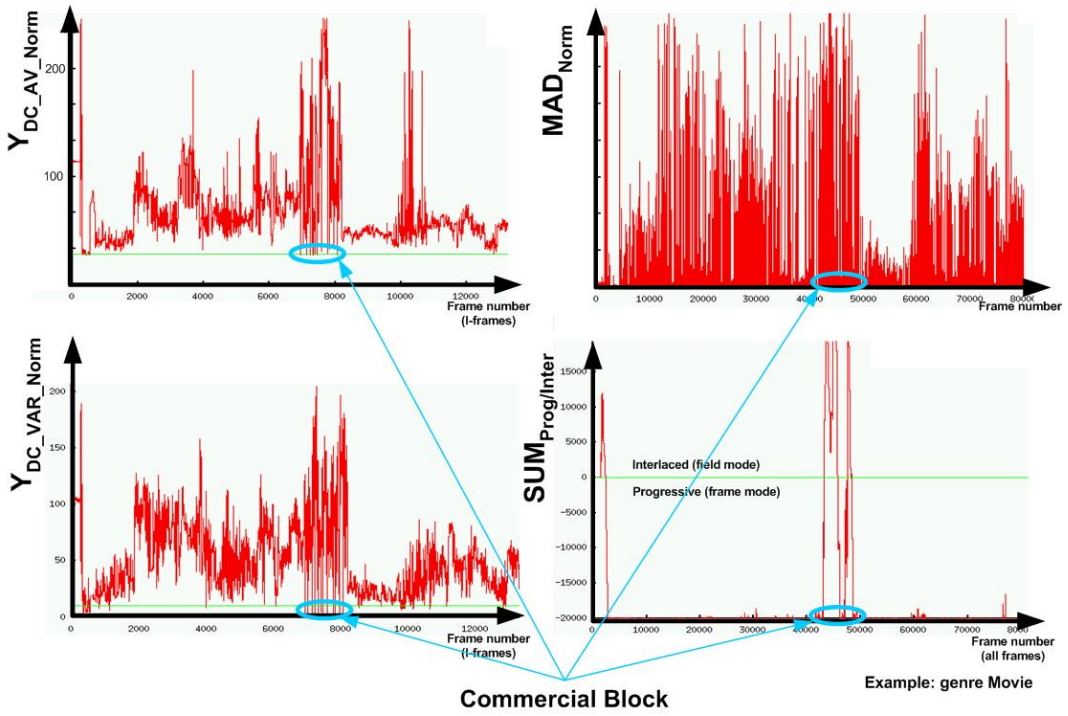


Figure 60. Distinctive behaviour of features for genre 'commercials'.

The nature of the high-level features commercial clips and commercial blocks vary, unfortunately, between cultures, i.e. nations, but also between commercial and national authority owned channels and even within channels dependent on the day of the week and the time of the day. An example for the latter is that during prime time, i.e. between 6 and 9 p.m., shorter commercial clips appear in more frequent commercial blocks e.g. compared to noon time. The variant nature and the need for a processing-efficient solution forces us to select the most salient commercial block detection features, which appear to be features enabling the detection of commercial clip separators. Because broadcaster replaced black frames by monochrome frames as visual commercial clip separator we select our monochrome frame detector, based on $Y_{DC\_VAR\_Norm}(N)$ of equation (4-9), and index all instances as *monochrome frames*, which fall bellow the in section 4.1.1 specified threshold of $Y_{DC\_VAR\_Norm}(N) \leq 0{,}015$. Hereafter, we index all monochrome frame instances as *commercial clip separators*, if they correlate time-wise with a commercial cut silence instance, as visualized schematically in Figure 61 (left).

In a final step we empirically analyse the statistical nature of commercial clip separators within our AV corpus, which we use to implement our commercial block detector. The analysis shows that the minimum and maximum length of individual commercial clips and, hence, the distance between commercial clip separators is $CC_{MinLength}=22$ seconds and $CC_{MaxLength}=640$ seconds. Hence, the first identified CCS is applied as reference and if at least two succeeding CCS instances, i.e. in total three CCS in row, appear within the CC boundaries, i.e. $CC_{MinLength}$ and $CC_{MaxLength}$, than we index the first CCS as start of the commercial block sequence. Here after, all subsequent CCS instances, which fulfil the CC boundary conditions, are added to the commercial block and the last CCS is indexed as end instance of the automatically detected commercial block.

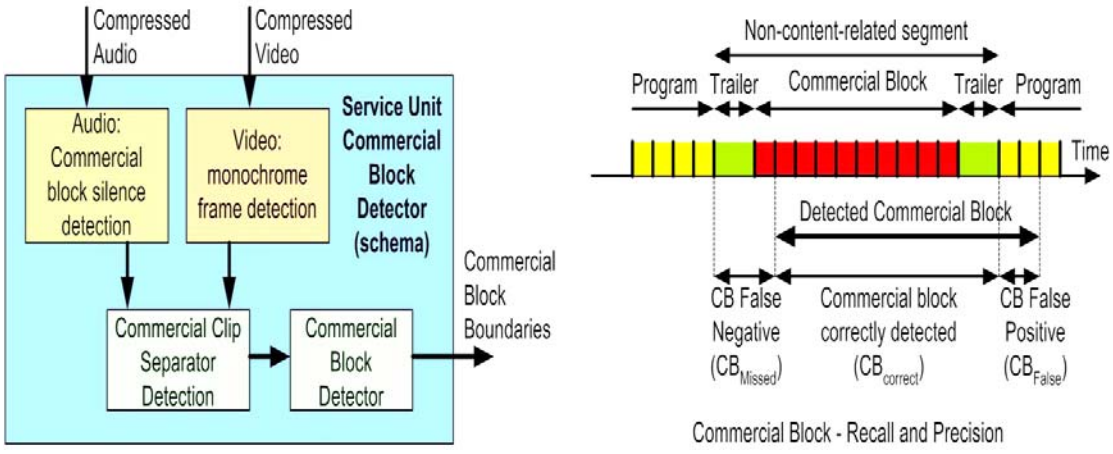


Figure 61. Commercial block detection - schema.

4.4.2 Results of commercial block detection

The application of commercial block detection in consumer products leaves us with the question of the specific usage of it in the consumer product. The desired solution would be to implement a fully automatic commercial block detector and, hence 'automatic commercial skip'. Nevertheless, user studies unveiled that also a 'lean forward' based, i.e. user initiated, commercial skip would fulfil the requirements. The user initiates the commercial skip as soon he encounters a non-content related insert. Hence, the detection of the beginning of a commercial block sequence is less important than the correct end detection of the sequence, because the system automatically jump to the automatically identified end point. Therefore the latter should rather be within the non-content related sequence than outside. In the case of the latter desired program content would be missed. We conclude that our system should achieve high precision, i.e. the duration of false positive instances F_P (CB_{False}) in seconds should be very low, rather than high recall, i.e. the duration of false negative instances F_N (CB_{Missed}) in seconds. False positives are those part(s) of program content(s), which are falesy identified as commercials, and false negatives F_N are those part(s) of commercials(s) and trailers, which are not identified as non-content related inserts, as sketched in Figure 61 (right).
The specific nature of this analysis motivates us to adapt the recall and precision equations (3-16) and (3-17), respectively. For recall we define $CB_{TotalTime}$ in seconds, which represents the accumulated duration of all commercial clips including other non-content related inserts such as trailers. On the contrary, for precision we define PC in seconds, which represents the accumulated total time of the program contents, with

$$\mathrm{Re} = \frac{CB_{TotalTime}}{CB_{TotalTime} + F_N} * 100, \qquad \mathrm{Pr} = \frac{PC}{PC + F_P} * 100 \qquad \text{(4-40).}$$

With this definition our precision should reach close to 100% to be useful for consumer devices, i.e. Pr~99.99% which results in 3.6 seconds of program content falsely indexed as non-content related inserts for every 10 hours of program content, as summarized in Table 23.

Table 23. Duration of falsely identified content for a set of precision values.

Precision	98.00%	99.00%	99.50%	99.90%	99,95%	99.99%
F_P per 10 hours	12 min	6 min	3 min	36 sec	18 sec	3.6 sec

For the evaluation of our commercial block detector we record 96 hours and 16 minutes of AV broadcast content (excluding non-content inserts) with 12 hours and 18 minutes of non-content related inserts, i.e. commercials clips and trailers. The latter are clustered in 204 individual commercial blocks. The corpus contains recordings from 23 commercial and national channels distributed across 7 countries. For the analysis we decided not to differentiate between trailers and commercial advertisements. The results of our commercial block detector, unfortunately, only reached a precision of 99.93%, as shown in Table 24, i.e. ~30 seconds of 10 hours of content are falsely indexed as non-content related inserts. The recall reaches about 91.4% and Table 24 summarizes as well the average and maximum duration of false positives and negatives, respectively.

Table 24. Commercial block detection results.

Pr [%]	Re [%]	Average F_P	Average F_N	Maximum F_P	Maximum F_N
99.93	**91.4%**	3 sec	29 sec	35 sec	59 sec

For the analysis we also classified each commercial block sequence into one of the following five groups, i.e.

- *complete correct detection* CCD of the non-content related insert,
- *entire commercial detected* ECD, but pre-deceasing and/or succeeding trailers are not (entirely) detected,
- *partial commercial detected* PCD, i.e. part of the commercial clips are missed,
- *non-relevant program content* NPR included, i.e. program content such as the program credits or program intros are included,
- *relevant program content* RPR included, i.e. relevant program content was indexed as non-content related inserts.

The distribution of the 204 commercial block sequences across the five groups is summarized in Table 25. The last two columns, i.e. NPR and RPR, represent the instances of severe problems, i.e. content identified as non-content related inserts.

Table 25. Results of analysis of total 204 commercial block sequences.

Group	CCD	ECD	PCD	NPR	RPR
Number of sequences	105	60	29	6	4
Percentage	51.5%	29.4%	14.2%	3.0%	1.9%

The analysis of the NPR and RPR cases shows that (a) content of individual music channels, i.e. music clips, exhibit commercial clip characteristic behaviour, and that (b) some commercial low budget movie channels do not separate program content from commercial blocks. The six NPR cases contain two cases at which credits (offset ~15 seconds), two cases at which program intros of soaps (offset ~34 seconds) and two music clip intro (offset ~12 seconds) are indexed as non-content related inserts. The four RPR cases contain three instances at which the program start (offset ~42 seconds) and one time the music clip itself (offset ~55 seconds) is identified as non-content related insert.

Applying a dedicated program credit detector, i.e. a scrolling text detector as we published in [118], which is based on our embedded text detector [119], could solve the problem of false positives during program credits, i.e. NPR cases. Furthermore, repetitive intros and outros of programs could be memorized with e.g. Forbes method [120], i.e. signal signatures of repetitive TV signals could be stored automatically and used for identification. The same method could be applied to increase recall, i.e. learning the signature of individual commercial clips, which are then used to identify missed commercial clips during a post-processing step. Another extension of our method would be to include a dedicated logo detector, because in many countries channel logos have to disappear during non-content related inserts and reappear after them.

4.4.3 Service Units commercial block Detection CBD

Finally we implemented both, our commercial cut silence detector of section 4.2.1 and our monochrome frame detector, and provide the resulting commercial clip separators to our commercial block detector unit. The latter generates an MPEG-7 compliant commercial block data output stream.

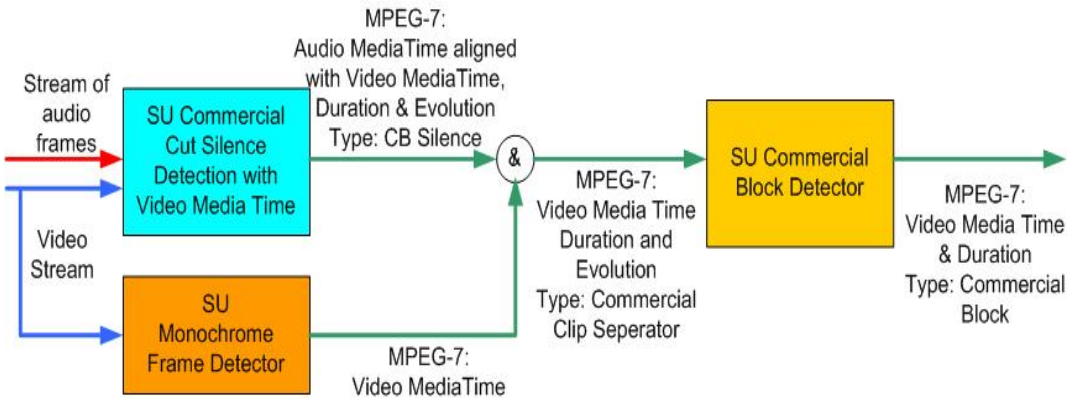


Figure 62. Service unit Commercial Block Detection CBD.

Hence, our service unit commercial block detection classifies all non-content related inserts in our AV corpus appropriately, as shown in Figure 63, and identifies both boundaries instances, i.e. *commercial block start* instance CBS and *commercial block end* instance CBE.

4.4.4 Conclusions

In this section we presented a commercial block detector, which we required to filter out all non-content related inserts before starting to segment the content into semantic meaningful chapters. We implemented here fore several audio and video low-level and mid-level features exploiting data most efficiently available within a video compressor, as we describe as well in [121]. Here after, we selected the most salient features, i.e.

- commercial cut silence detection (as we describe as well in [33] and [116]) and
- monochrome frame detection (also published in [122], [123] and [124]),

which we applied for our commercial clip separator detector. Subsequently, we used a statistical analysis to derive appropriate rules for our commercial block detector. For the benchmark of the latter we recorded a broader test set of broadcast content. The analysis unveiled that the precision of the detector reached about 99.93% and a recall of 91.3%. Because the achieved robustness did not satisfied the requirements of an automatic commercial skip application we decided applying the detector for a manual commercial skip application. Furthermore, we developed an intuitive user annotation tool, as we described in Annex 10, with which consumers can easily shift the automatically detected boundaries to the appropriate location. Only recently we saw that a few CE device makers provided on very few of their PVRs commercial block detection functionality, but no skip based application mainly due to the low detection rate of their solution.

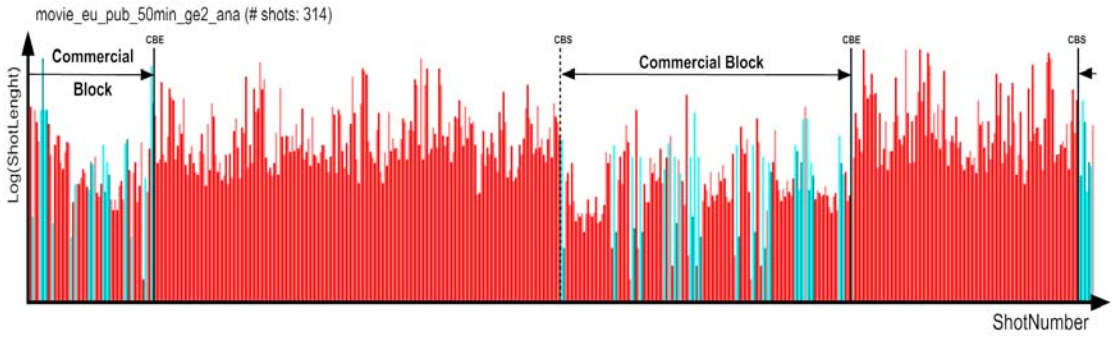


Figure 63. Shot segmented and non-content indexed content item (movie_ge2).

4.5 Film-grammar based audiovisual content clustering

The service units of the preceding chapters, i.e. shot boundary- and commercial block detection cleaned the content and segmented the content-only sequences into its atomic units, i.e. shots. For the final aim of retrieving semantic audiovisual discontinuities, i.e. scene boundaries, the author decided to include an intermediate step, i.e. retrieving and clustering of visually correlated content sequences based on commonly applied content production rules. Sequences of the latter category exhibit such a strong internal correlation that from a comprehensive point of view it is impossible to divide such a sequence, e.g. by a scene boundary. Hence, we decide to investigate those rules and to develop an appropriate clustering method in order to reduce the potential scene boundary instance space, i.e. to pre-cluster the content as much as possible before searching for scene boundaries in the remaining parts.

In the following section, in 4.5.1, we describe the internationally acknowledged and commonly applied, e.g. by content production industry, film-grammar based production rules. The latter include cinematographic features, shot classification and parallel shots. The reader should be aware that those rules hold true only for a sub-set of all audiovisual genres, i.e. narrative content such as soaps, series, special magazines, some sub-groups of documentaries, cartoons and movies. Hereafter, we propose methods for content clustering, i.e. parallel shot detection, in section 4.5.2, and for categorization, in 4.5.3, respectively.

4.5.1 Film Grammar for AV scene segmentation: Introduction

4.5.1.1 Production metadata

The production of TV broadcast- and cinema content underlies a handful of common conventions often referenced as *film grammar,* Bordwell [16]. In this section the basics of film grammar and its related domain-specific terms are explained in more detail to sketch the opportunities for multimedia content analysis.

First of all, the reader or those who work with film analysis have to realize that film production is an art on its own. As such, it is very difficult to analyze film content objectively, because it is based on and makes usage of the creativity and subjectivity of the content producer or director, who has his/her own style. Fortunately, almost every producer or director commits himself to follow the above-mentioned conventions during the production of multimedia content, as sketched in Figure 64.

First of all a *scenario* is sketched containing a short overview in terms of scenes, their location, their settings, the actions contained and its involved characters. The segmentation into its narrative elements, i.e. scenes, is explicitly present in the scenario, which as such is mainly used to estimate budgets and agree on, hence, required changes. Thereafter, a *script* is written defining the semantic concept and the screenplay. It contains objective description elements, such as location / environmental setting, time of the day, dialogues, shot composition and video editing transitions. The script, which incorporates film grammar rules, is still readable like a novel and provides the *producer* or *director* with an initial layout for the production. During the production the director or producer may adapt the concept if required, e.g. due to budget cuts, and those decisions in combination with the now and then updated script result in the final production decisions. Those production decisions together with the script information are very valuable *production metadata*, as shown in Figure 64, providing content- and context-relevant information about the content. Unfortunately, those *production metadata* are not standardized and even if available not stored appropriately with the *produced content*.

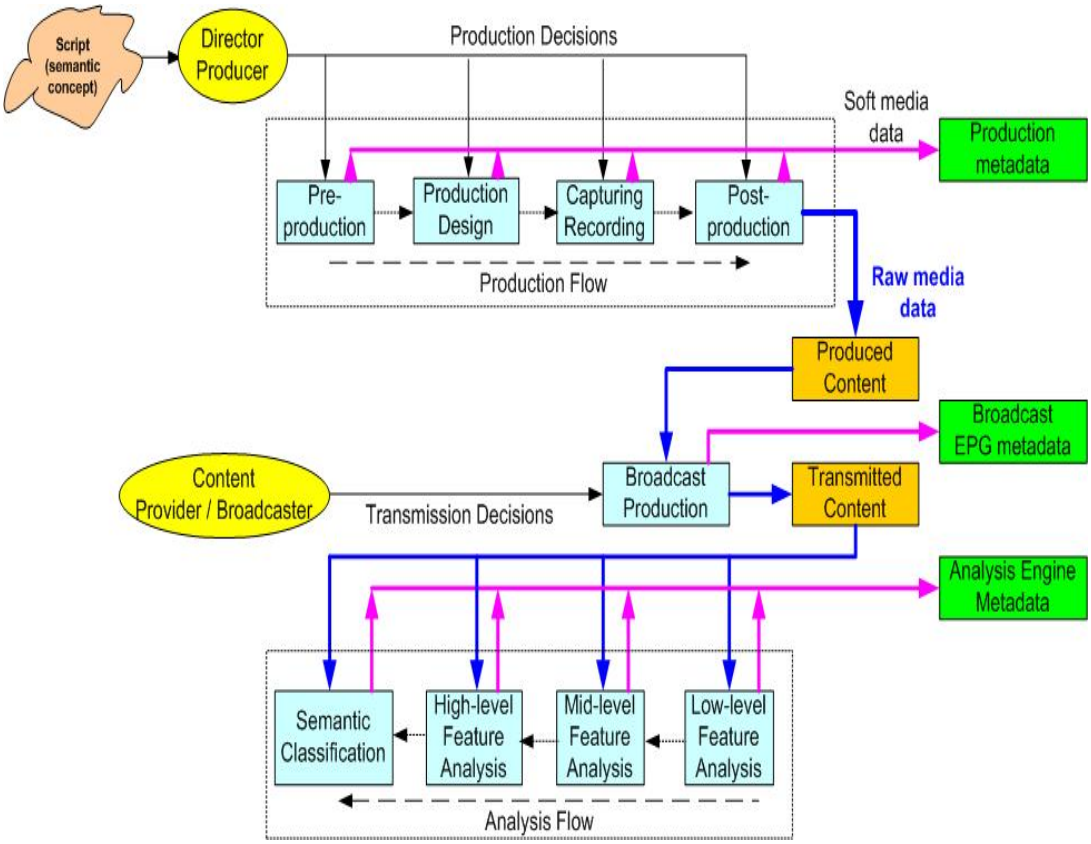


Figure 64. Production and analysis flow[1].

Before capturing individual takes the script is used to make a storyboard composed of individual sequential graphics, as sketched in Figure 65, depicting the narrative story line supporting the director to imagine the settings, e.g. arrangements inside a shot, exact camera positions, shot type, costumes, make-up, lighting and other cinematographic components, explained in more detail later in this section.

Subsequently, *content providers* or *broadcasters* change the flow of the produced content according to their business models. Broadcasters create a continuous broadcast flows by concatenating various produced content items, which to some extend are captured by *Electronic Program Guide* (EPG) metadata and transmitted by means of e.g. *Digital Video Broadcast – Service Information* (DVB-SI), containing data such as title, genre, channel id, start time and duration, synopsis (abstract) and key words.

Figure 65. Storyboard of a scene with crosscutting - realized in Figure 78[1].

Optionally they can also contain data such as the name of the director and actors, country of origin and language. Unfortunately, the current business models of broadcasters, which to some extend is based on commercial inserts, inhibit the open and accurate transmission of *broadcast metadata*, sketched in Figure 64. Often commercial blocks flank content items and therefore EPG start- and end times are kept fuzzy. In addition, commercial blocks are inserted into content items, but due to the business model no information is available about the location of the commercial inserts.

The task of content *analysis engines* can be seen as a kind of reverse engineering. Production rules, production decisions and transmission decisions are not properly maintained or provided with the produced content and therefore the metadata have to be re-generated by analyzing the incoming AV signal with an analysis engine resulting in *analysis engine metadata*, as shown in Figure 64.
In chapter 3 we have already introduced some prior works, which implicitly or explicitly apply production rules in order to segment content into video entities. We apply in this work additional film grammar knowledge to elaborate new methods for the final aim of clustering and segmenting audiovisual content. Hence, in the rest of this section more insides in the film grammar rules will be given to understand the common production rules, which were necessary to define proper analysis engine algorithms for this work. It is needless to say that the ultimate goal is to generate all relevant production metadata.

4.5.1.2 Film grammar

The task of director and editor is to use script and storyboard to produce an audiovisual sequence telling a logical and comprehensible story, which should fulfill certain *'film grammar rules'* and which should be to some extend uniform to the chosen film genre. For us film grammar is a set of rules we can apply for this work. Film grammar often has specific patterns, which have established themselves in the course of time e.g. the narrative structure of classical movies consists of three *acts* separated by two plot points, as sketched in Figure 66 and explained by Bordwell in [16] and Beaver in [15]. Hence, the production of raw material starts with capturing *takes*, as described in 4.1.2, which are composed matching as much as possible to the sketches described in script and storyboard.

Mise-en-Scene production rules

Directors, here for, use a well-known technique called *Mise-en-Scene*, i.e. French for 'putting into the scene', which covers from a cinematographic point of view all what a viewer sees, i.e. spatial compositions, settings, camera position, make-up, light settings and space-time relations. Properly applied Mise-en-Scene enables to convey emotional effects, symbolic messages and meanings, but also to concatenate content instances on an abstract level together.

Setting, Costumes, Make-up, Lighting, Space and Time

Director or script defines where an event should take place, which is called the *setting*. Everything, which surrounds the actor, is part of the setting. Outdoor scenes, for example, are often shot at authentic locations to preserve authenticity. Indoor scenes are predominately produced in studios due to access to professional lighting and equipment facilities. In general nothing is left to chance, what is shown in a movie, and even if something should occur random it is most probably intended. Through settings directors are enabled to deliver certain messages to the audience. A scene with e.g. impressive buildings and a gaggle of supernumeraries shall deliver a mind-blowing atmosphere. In contrary, a scene with a monochrome background should direct the viewer attention to e.g. the specific gestures and mimics of an actor.

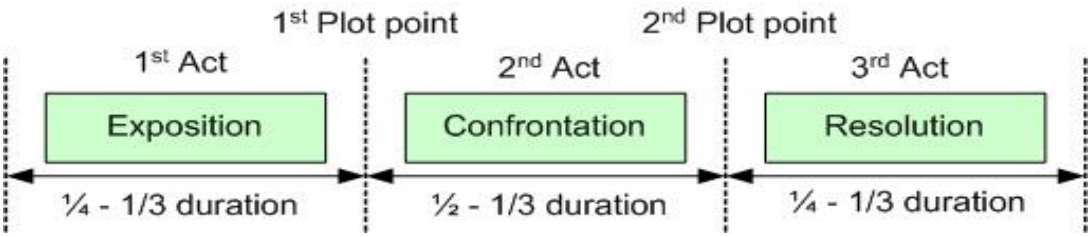


Figure 66. Narrative structure of classical movie consisting of three acts.

Costumes have the purpose to support the authenticity created by the setting, e.g. emphasizing the era of the event. Moreover, costumes have the function to define certain actor's roles, e.g. the viewer will immediately uncover a magician as such through his long dark robe and a soldier through his uniform. *Make-up* has a similar function in defining an actor's role. It can convert an actor into anything and anybody such as a beast as well as a famous composer.

"Light expresses ideology, emotion, color, depth, and style. It can efface, narrate, describe. With the right lighting, the ugliest face, the most idiotic expression can radiate with beauty or intelligence", stated by director Frederico Fellini. Light has a strong impact on the interpretation of a scene and with light certain subjective feelings can be triggered. Furthermore, light can have the function to stimulate the viewer to focus on specific areas, objects / subjects or actions. Light can be separated into *key light* and *fill light*, which are the two basic sources used to lighten a scene. Key light is the strongest possible light and usually corresponds to the actual light source in the scene, which is often also visible to the viewer. Fill light is used to illuminate the entire setting and to soften existing shadows. Lighting graduations are often used to differentiate between day and night settings. "*There is a strong shadow where there is much light*." (J.W. v. Goethe, "Goetz von Berlichingen", 1st act). In cinematography there are two kinds of shadows the *attached shadows* or *shading* and the *cast shadows*. When obstacles hinder light to illuminate the entire setting shadows occur. The human nose causes for example almost always slight shadows in the face of the actor, which is called shading. On the other hand specific objects can create strong visible meaningful shadows called cast shadows.

Figure 67. Setting, Make Up and Lighting examples[1].

Another aspect of film grammar is the creation of *a continuous story line*, which has to be done carefully because the human visual system and brain is very well trained to perceive changes, both in *space and time*, much better than uniformity. This fact will be further exploited and therefore the non-uniformity aspects of *Mise-en-Scene* will attract our further attention, especially those related to changes in light, shape and movement. Essential here fore is the understanding of some *cinematographic* rules.

Cinematographic rules

Cinematography, literally the *writing in movement*, provides the director with tools to manipulate the viewer's experience and to create (non-)uniform impressions, using e.g. range of tonalities, speed of motion and transformation of the perspective.

Range of Tonalities

The director has for example a *range of tonalities* choice of creating either color or black-and-white content. In addition, he can regulate the contrast in the content stream, which refers to the degree of difference between the darkest and the lightest areas in the singular frame, but also across the shot and the entire content.

Speed of motion is mainly used to create an impression of action or suspense, but also to focus the viewer's attention on something. In live sport events the point of interest is often passed before the viewer is able to absorb the information, e.g. a goal in a soccer match. Hence, *slow motion replays* are used to flashback onto the instance of interest and at critical moments a freeze-frame may be used to enable the viewer to digest the instance. On the other hand, an increase of speed of motion is often used to create an impression of action and activity. In the extreme case, if increased to comic-like speeds, it is often used to give an overview of a long-windowed process, such as a sunrise.

Perspective relations can be applied to achieve e.g. the impression of 3-D spaces. Those relations deal with the spatial relation of objects / subjects inside a setting. In a nature setting for example, trees located close to the viewer are perceived much bigger then mountains, which are far back in the scenery. The kind and the settings of the camera lens enable to simulate the human eyes focusing capabilities changing the perspective relations. A wide-angle lens, for example, tends to hyperbolize depth in a recording, whereas a telephoto lens drastically reduces the depth.

Shot Distance – Shot Type

Perspective relations are created e.g. with properly selected camera angle, height and *shot distances*, i.e. the position from which a setting is captured in relation to the setting. The latter are used to separate shots into *Establishing Shots*, *Long Shots*, *Medium Long Shots*, *Medium Shots*, *Medium Close Shots*, *Close-Ups*, *Big Close-Ups* and *Extreme Close-Ups*, as summarized in Figure 68. When starting with e.g. a new scene, the

director has to introduce the viewer to the new setting. Hence, the opening shot, also called *Extreme Long Shot* (ELS), *Establishing Shot* or *Objective Shot*, often presents a general view of the setting from a far distance, where the protagonist only plays a secondary role, whereas the surrounding is the main focus at this moment. Attention has to be paid to the fact that Establishing Shots can also occur during a scene e.g. when (re-) establishing an 180º system (see next section). *Long Shots* (LS) are defined by the fact that the entire actor is frame central. On contrary, in *Medium Long Shots* (MLS) the lower frame line cuts off the actor's feet and ankles. Documentaries with social themes e.g. mainly make use of Long Shots to focus on social circumstances rather than on individuals. In *Medium Shots* (MS) object or subject of interest and its surrounding setting share equal frame areas, e.g. in the case of a standing actor the lower frame line passes through his/her waist, providing sufficient space to follow his/her gestures. Reducing further the distance leads to *Medium Close Shot* (MCS) level, wherein the lower frame line passes e.g. through the chest of an actor often used for a tight presentation of two persons. *Close-Ups* (CU) are covering extreme close distances, showing only e.g. the character's face and its shoulders in great detail so that it fills the screen. Those shots abstract the subject from the context. *Big Close-Ups* (BCU) show only an actor's forehead and chin, focusing the attention of the viewer on a person's feelings and reactions. They are sometimes used in interviews to show participant's emotional excitement state, grief or joy. Finally, *Extreme Close-Ups* (ECU) isolate a portion of an object or subject to magnify the moment.

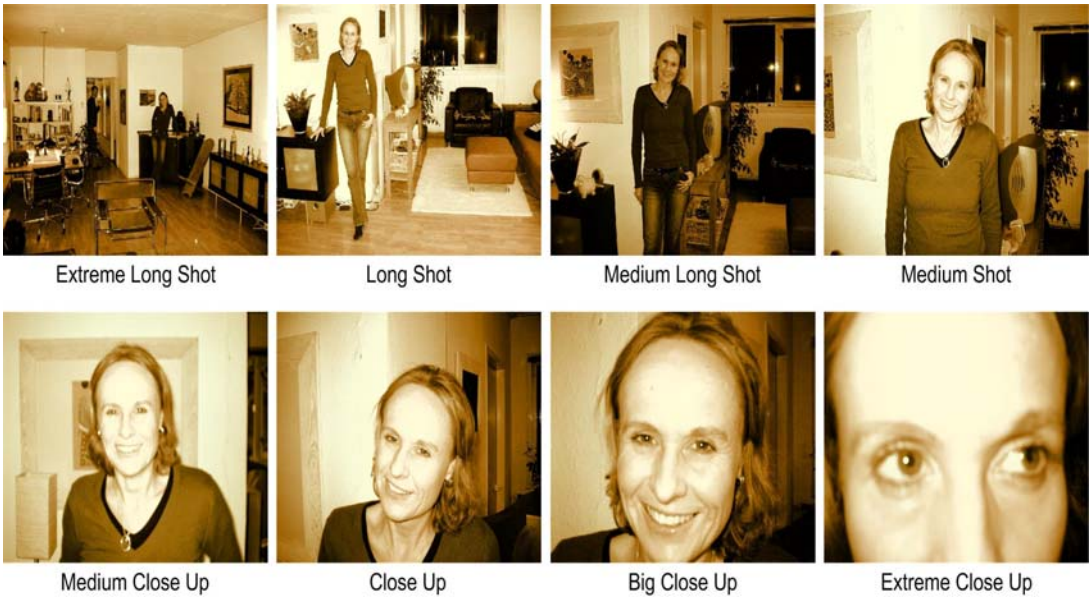


Figure 68. Examples for various distance shots[1].

Angle of shot: The *angle of the shot* is determined by the direction and *height* from which the camera captures the setting. The convention for e.g. news programs is, that the camera is on eye-level with the anchorperson (*Eye-Level*). Particularly dialogues make use of this technique to maintain the eye-level when switching between speaker A and speaker B (*eye-line* match, as published by Boggs in [85] and shown in Figure 69). Special cases are *Close-Ups* in which males are usually shown from just below eye level and females from just above eye-level.

Level Height: In a high angle setting the camera looks down on the actor (*High Angle*, see Figure 70*)*, which puts the viewer in a (psychological) stronger position than the actor. A shot from below puts more impact on the actor's importance (*Low Angle*). An overhead shot, also called *Bird's Eye*, is made from a position located directly above the scene. The opposite is called the *Worm's Eye*. A *Tilted Shot*, also called chanted shot, is created when the camera is tilted to its own axis. In this way normally vertical lines appear slant, which creates an unease feeling at the viewer and is often used in mystery content. A special case are the *Point-of-View Shots* were the scene or setting is shown out of the eyes of another actor and is therefore shot in sight to this person.

Figure 69. Dialogue with Eye Line[1].

Figure 70. Examples for various shot angles and various positions[1].

Camera Motions

Camera motions specify the trajectories of a camera in relation to the setting and can be classified into *Zoom In*, *Zoom Out*, *Tilt*, *Pan*, *Dolly* and *Tracking*, sketched in Figure 71 (left), *Crab* and *Crane*. During a zoom the camera position does not change, but the focus of the lens is adjusted, as shown in Figure 72. An example for a *Zoom In* is the continuous transition from a *Long Shot* to a *Close Up*, used to guide the attention of the viewer to something, which might be invisible in the Long Shot. On contrary, the *Zoom Out* uncovers areas in the setting, which were previously not visible. In the case of a tilt the camera retains at a fixed position, but it experiences a vertical rotation around the horizontal axis either upwards (*Tilt Up*) or downwards (*Tilt Down*), shown in Figure 73.

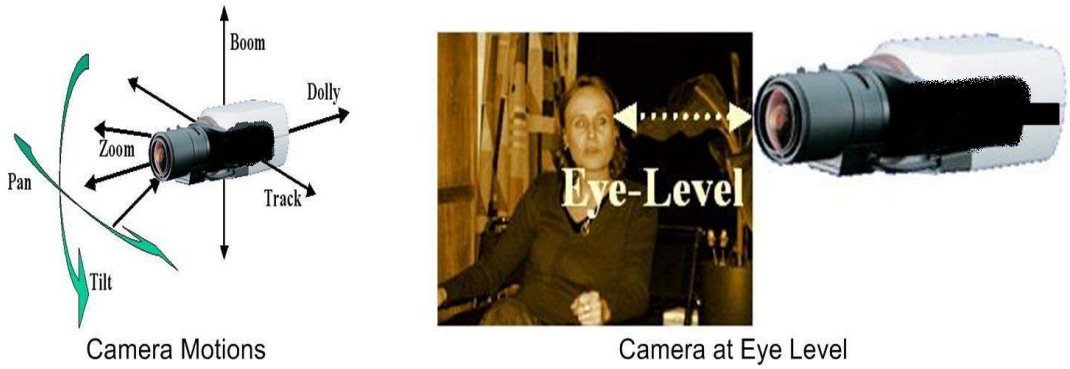


Figure 71. Camera Motions and Eye-Level[1].

Figure 72. Examples for zooming[1].

Figure 73. Examples for tilting[1].

A *pan* is defined by the fact that the camera retains at a fixed location, but it sways in horizontal direction, i.e. rotates around its vertical axis, either to the left (*Pan Left*) or to the right (*Pan Right*), as shown in Figure 74. The pan leads more than it trails and therefore a space is always left in front of the moving object or subject. In films a pan is usually flanked at both sides (beginning and end) by some seconds of a still picture to increase the impact of the pan. The mood of the content can be influenced through the speed of the panning as is often done in action feature films. Incidentally it happens that inexperienced operators pan too fast, which cause an effect called strobing or tearing.
A *dolly* is a small-wheeled vehicle, piloted by a dolly grip that is used to move a camera around in a scene. A dolly shot is a move in (*Dolly In*) and out (*Dolly Out*) of a scene, i.e., the movement is parallel to the camera lens axis. Fast *Dolly In* creates excitement at the viewer, whereas a *Dolly Out* relaxes the interest.
A *tracking* (or crab) shot is a movement perpendicular to the camera lens axis, i.e. the camera moves horizontally right or left in relation to the setting, sketched in Figure 75, whereas during a *crane* (or boom) the camera executes exclusively vertical movements.

The presented mis-en-scene and cinematographic rules enable the director to record individual takes matching script and storyboard and conveying abstract massages, but also time wise constituencies have to be satisfied and are applied as described next.

Figure 74. Examples for panning[1].

Figure 75. Examples for tracking[1].

Post-Production Editing Rules

During the capturing / recording part of the production flow in Figure 64 raw video material has been produced, enabling the cutter, editor and/or director (further only referenced as cutter) to select appropriate material to create a consistent flow. The cutter has the task to select and concatenate the raw material appropriately fulfilling the above-described cinematographic rules of (non-) uniformity. Hence, first of all he/she selects appropriate raw material and removes unwanted footage like unnecessary takes of clips or the clapboard in the beginning of each take. Thereafter, he creates individual shots, which are continuous camera recording events, as described in 4.1.2, and concatenates those together using traditional techniques such as hard *cuts*, but also artistic transitions, i.e. *dissolves, fade ins*, *fade outs* and *wipes*, as described in 4.1.2.

Film Editing

The dense way transmitting semantics, nevertheless, demands from the cutter to densely pack shots with information and, increasing the compression even further, to capitalize on the viewers intelligence of self-conclusions and semantic interpretations. The cinematographic *Kuleshov Effect*, named after Russian director Lev Kuleshov, is a good example exploiting the viewer's intellectual capabilities. A shot containing a neutral expression of Ivan Mozzhukhin's face has been combined alternatively with shots of a plate soup, a girl and a child's coffin. The audience believed that Mozzhukhin's face expressed associated feelings as hunger (soup shot), desire (girls shot) and sorrow (coffin shot), not knowing that the face shot was always the same one, which proofed the effectiveness of film editing. Hence, the essence is not only presented through the shot's content, but also through its time wise compilation with each other. In general there are four groups of such relations used by filmmakers, which are (a) *graphic relations*, (b) *rhythmic relations*, (c) *spatial relations* and (d) *temporal relations*.

During the post-production process the cutter (as well as the director or editor) has to secure continuity and uniformity of the stream using those relations. Shots are either concatenated to obtain a soft continuity, i.e. uniformity is present across shots, or to create abrupt changes, i.e. a strong mismatch between consecutive shots.

What concerns *graphical relations* the cutter has various graphical elements to influence the continuity and uniformity, e.g. continuity of visible shapes, colors, range of tonalities, and movements. Fulfilling all the above graphical rules leads to a perfect uniformity, also called *graphic match*, but in reality such a perfect match is difficult to achieve and therefore editors mainly concentrate to keep at least the centre of interest at the same location.

Another instrument to guarantee uniformity, but also to create certain feelings, is the shot-length-based *rhythmic relation*. Shots can span several frames to several thousand frames as we describe in 4.1.2. The cutter controls the length of shots and with concatenating them he/she also controls the rhythmic potential. Increasing the duration of a shot gives the viewer time to process the information of the passed scene, but also to increase the tension as often used in dramatic scenes. For the latter an increasing shot length builds up a certain suspension before the climax of the scene is reached, which is often followed by a long shot to enable the viewer to prepare him/herself for the next highlight. On contrary, short shots are either used to create action and excitement, as e.g. in action movies or music clips, but also to transmit a lot of information, as usually used in commercial blocks. Important to realize is that the shot length, and therefore the rhythm relation, is proportional to the camera's distance, i.e. *Long Shots*, at which a viewer has more to absorb, are in average longer than *Close Ups*.

A viewer also expects uniformity what concerns *spatial relation*, hence, it is of importance to create first a sense of the scenery by means of an establishing shot and shots covering the scene-relevant locations and / or actors. Those spatial relations are secured following the *180º system*.

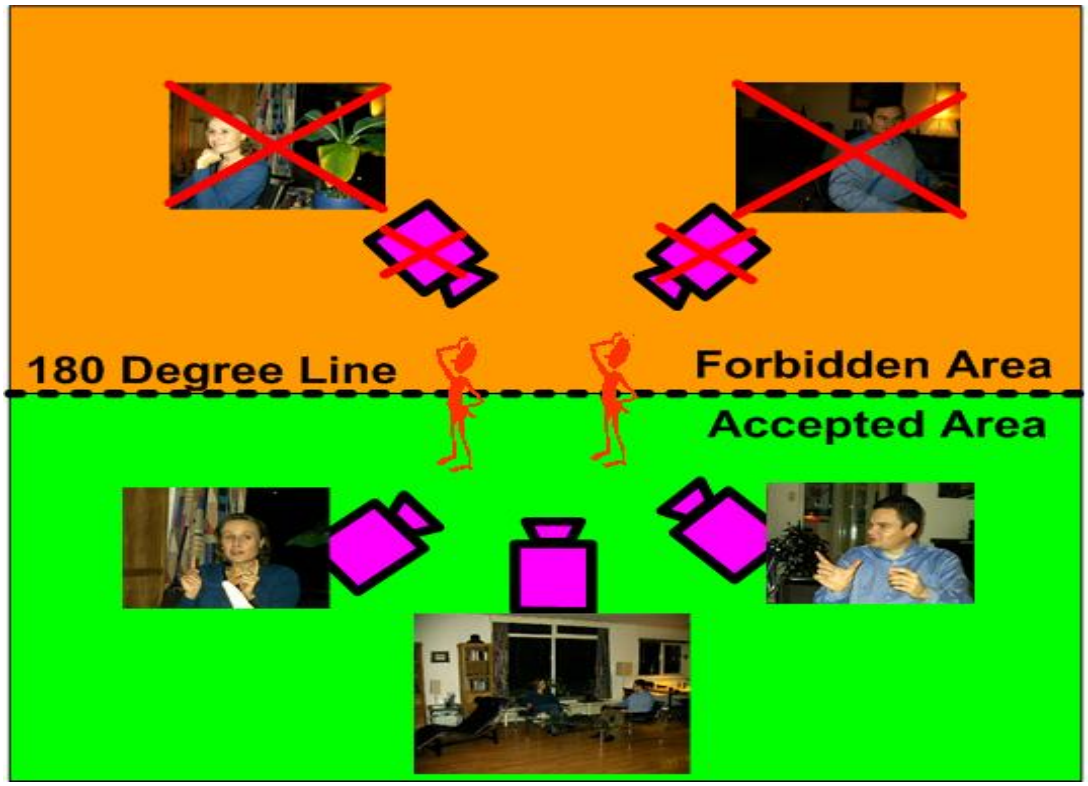


Figure 76. The 180° System[1].

While watching a certain activity in a setting the viewer expects certain uniformity in terms of the camera's location, i.e. the action should take place along a so-called *axis of action* also referenced as *180º line*. This *virtual* (imaginary) line cuts the setting into a forbidden camera location area and an accepted area, as sketched in Figure 76. This rule is important to avoid confusing the viewer.

Hence, the camera has to capture the scenery, here as an example the dialogue between two persons, permanently from the accepted side. Furthermore, it is important that the objects of interest, here the dialogue partners, have to be positioned spatial-conform inside the captured frame, as the examples of Figure 76 shows, to support the spatial relation to each other.

But there are also artistic exceptions, e.g. Lev Kuleshov used a technique to only create the impression of an uniform scenery, by actors virtually 'looking at each other' in concatenated shots, who were in reality physically dislocated. Spatial relations can also point out activities, which happen at the same time, or even at different times, at different locations, which we will describe in more detail later addressing them as *parallel shots*. The most drastic way to use this technique is to create an uncertainty of the location by showing little of the scenery. This is sometimes used in suspense movies.

The seldom-used *temporal relation* technique covers situations were a continuity of an action is interrupted by temporal dislocated instances such as flashbacks or flash-forwards, activities that happened either in the past or will happen in the future, respectively. To establish a relation, editors use for this often dissolves.

The so far described production rules, i.e. mis-en-scene, cinematographic rules and post-production rules including several relation rules, guarantee uniformity and narrative continuity. As a result, viewers perceive the result of the production as smooth and uniform, and, hence, not distractive allowing them to follow the narrative content of the story.

Semantic segments

According to script writer Syd Field [125], good story scripts are subdivided in 3 acts separated by two plot points, as already depicted in Figure 66, starting with the introduction of the actors, scenery and story in the exposition act. The latter is followed by the confrontation act containing the battle of the protagonist loaded with action and tension, and, finally, followed by the resolution act, letting the protagonist reach a certain aim. Those acts are further subdivided into semantic scenes, which in the ideal case start with an *establishing shot*, followed by several sometimes interleaved *narrative elements* following a certain *peripety* and finalized by a *conclusion shot*.

Establishing and conclusion shots

Some general rules on the types of shots used during film production can be given. A scene has to be established, for example, first giving the viewer the chance to accommodate with the new setting. Hence, a scene is established with an *establishing shot*, normally the first shot(s) of a scene. Giving an overview requests for great depth achieved by using *Extreme Long Shot* ELS or *Long Shot* LS sometimes in combination with slow panning. Hence, these types of shots give insights about the setting and its spatial relations. Due to their information richness the *Shot Length* SL of theses shots, as we describe later in this chapter, is in general proportional to the shot depth, enabling the viewer to absorb the details. We experienced as well that when scenes / settings are reestablished, i.e. settings already established in a preceding scene, those reestablishing extreme long shots have a rather short shot length or are left out. In the case of the latter the director counts on the viewers memory.

Conclusion shots, on the other hand, applied to conclude scenes and used to show that a certain action came to an end, often have to recall the setting, which is achieved through extreme long shots (high shot length). Moreover, conclusion shots are used to relax the tension of an e.g. action loaded scene lowering the viewer's excitement to a level, which allows him to enter the new scene. Nevertheless, we witnessed that conclusion shots are often skipped by directors, as we will prove later in this chapter.

Interleaved narrative elements – Parallel Shots

Each individual scene can contain one or more embedded *interleaved narrative elements*, which we further address as *parallel shots*, and each individual scene is encapsulated by one or more establishing and conclusion shots. Furthermore, in some genres these scenes follow a certain *peripety*.

Each *narrative element* represents one individual event consisting of strongly related but not necessarily connected shots, i.e. shot fulfilling the previous described relations. Usually two or more narrative elements are interleaved with each other, forming parallel shots. We introduce the following definitions for parallel shots:

- Parallel shots are interleaved narrative elements.
- *Parallel shots*, can be divided into two main groups, i.e. *cross-cuttings* and *shot reverse shots*.
- *Cross-Cuttings* visualize, in general, either (a) time-wise correlated, location-wise disjoined parallel running narrative events, i.e. same time, but different location were interaction is not obligatory, as shown in Figure 77 and Figure 78, or (b) time-wise uncorrelated events such as one event and a flash-back, i.e. different time at the same or different location.

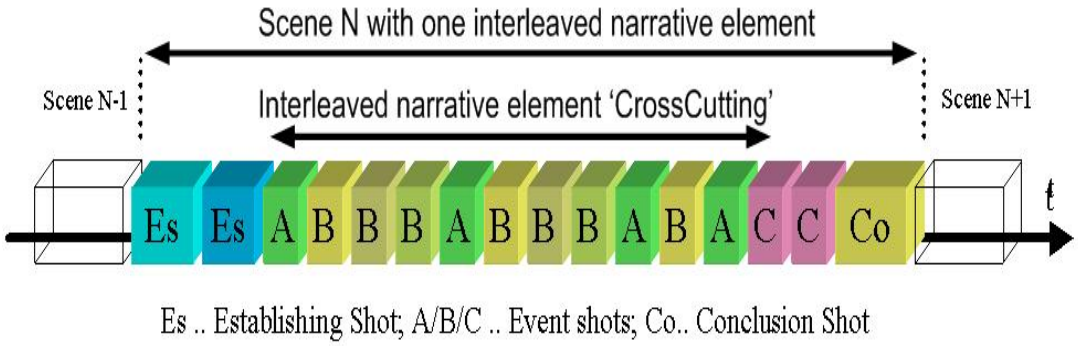


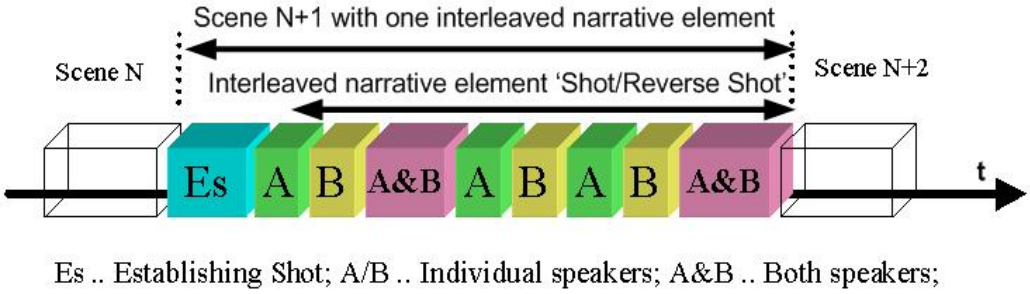
Figure 77. Schematics of a scene with cross-cutting.

Figure 78. Scene with cross-cutting: depicts two events (A and B) that unfold simultaneously. Interleaved rendering of A & B[1].

- On contrary, *shot reverse shots* are used to visualize events happening at the same time at the same location, captured from two or more camera positions and rendered in an interleaved manner, e.g. A-B-A-B, such as a dialogue between two actors. An example is given in Figure 79. In-between the interleaved sequences distant shots are used, e.g. an AB shot, to introduce spatial relations. Here after, A and B shots follow the spatial relation rules. Essential for *shot reverse shots* is to fulfill the earlier described eye-line match.

Moreover, two additional semantic structures are often applied inside parallel shots, i.e. *peripety* and *four-action sequence*. *Peripety*, defined by the poet Aristotle, is the interplay between sensations increasing success passages and crushing disappointment ones. *Four action sequences*, on the other hand, are semantic logical sequences of *perceiving*, i.e. shot of an actor in perceiving pose, *perception*, i.e. shot depicting the actor's attention, *pondering*, i.e. shot showing the actor's pondering, and *action*, i.e. shot with the final action taken.

Figure 79. Scene with shot reverse shot: dialogue between two individuals (A & B) shown in an alternating fashion[1].



Es .. Establishing Shot; A/B .. Individual speakers; A&B .. Both speakers;

Figure 80. Schematics of a scene with shot reverse shot as used in Figure 79.

The film grammar, including its artistic rules, is essential to semantically analyze audiovisual content, which includes classification and segmentation of AV content into its semantic entities, i.e. scenes. In the following sections various AV analysis tools will be described, which the author chose based on the knowledge described in this section.

4.5.2 Film grammar rule based content clustering into parallel shots

With the work of the previous section (4.1.2) the content is so far segmented into its elementary shots, i.e. a kind of reverse engineering of the editing work done by the content editor, by means of shot boundary detection. The latter contain cut transition detection and gradual transition detection. Furthermore, non-content related inserts, i.e. commercial- and channel adds, inserted by the broadcaster (transmission decision), were identified and indexed by the commercial block detector (as explained in section 4.4).

In the present section, in addition, we apply film grammar based rules to accomplish higher-level content analysis, embedded in higher-level service units. The latter, e.g., cluster intentionally interleaved shots of two or more narrative events together, i.e. dialogues, a.k.a. shot reverse shots, and cross-cuttings, as introduced in 4.5.1.

Ground truth and statistics of parallel shot sequences

The objective evaluation of the parallel shot detection methods on its robustness requires a manually generated ground truth. Hence, objective rules were defined for the manual annotation. Knowing that directors (editors) establish a bridge between related, but not consecutive, shots through a continuation of the audiovisual story flow, the rule has been established to visually compare one of the last key frames ($W_{KF}>1$, i.e. excluding at least the last one) per shot with one of the first key frames ($W_{KF}>1$) of a certain number of consecutive shots ($W_{sh}>2$, i.e. excluding the proceeding one), as sketched in Figure 81. In the case that key frames at $W_{KF}=2$ do not fulfill the requirements of a representative key frame the best-fitting one within the range $W_{KF} \in [2..10]$ is chosen.

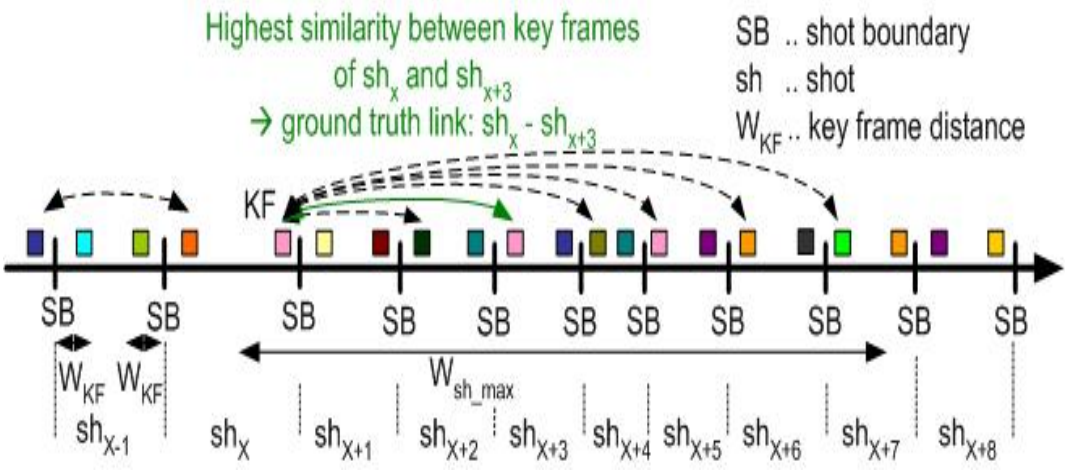


Figure 81. Rules for parallel shot ground truth annotations.

The analysis or a large amount of content, i.e. more than 12 hours, unveils that editors do not separate semantically connected shots by more than ~7 shots. Hence for this work we select the maximal distance W_{sh_max}=7 for the manual annotation. In the next step, the end key frame of e.g. shot *x* is compared with the start key frame of all subsequent W_{sh_max} shots, and the two shots with the highest key frame pair similarity are linked together (as shown in Figure 81). The similarity is specified, based on the definitions of cinematographic rules and relations of section 4.5.1, as simultaneous strong visual correlation of foreground and background information and similar spatial layout.

The only restriction we apply is that shots can only have one single forward and backward link. Finally, shots linked to subsequent shots, but not to predeceasing ones, are determined as the beginning of a parallel shot. On contrary, shots linked to predeceasing shots, but not to successive ones, are indexed as the end of a parallel shot. All shots between the beginning and the end shot of a parallel shot sequence are considered as members of that parallel shot sequence. In an additional step of the manual annotation, individual linked shots are either indexed as shot reverse shots, i.e. dialogues, or cross-cuttings based on the rules given in 4.5.1, according to the location of the scene criterion. Furthermore, we different between links between individual SRS shots '*SRS links*' and CC shots '*CC links*' and apply them for statistical purposes.

Ground truth GT results covering the series/movies AV corpus are summarized in Table 26. The statistical analysis, summarized in

Table 27, shows that the ratio between SRS and CC sequences is about 3:1 in series and 0.8:1 in movies (1st column), which fulfils the expectations that series contain more dialogues.

Columns two and three show that in series about 60% of all shots are member of *shot reverse shot* SRS sequences (dialogues with an average duration of 9.5 shots) and ~13 % of *cross-cuttings* CC (average duration of 6.4 shots).

On contrary, in movies only 46% of all shots are members of shot reverse shots (with an high average duration of 14.8 shots) and almost 30% of cross-cuttings (average duration of 7.4 shots). Finally, columns four and five in

Table 27 unveil that the ratio of parallel shot links compared to the number of member shots is for both, series and movies, about 0.7 for shot reverse shots and ~0.45 for cross-cuttings, i.e. link bridges in movies are almost twice the size in terms of shots compared to those in series.

Table 26. Ground truth numbers of SRSs and CCs of series /movies AV corpus.

Content		# of SRSs	# of shots in SRSs	# of GT SRS links	# of CCs	# of shots in CCs	# of GT CC links	# of shots total
Series	'nl1'	9	104	56	11	60	23	227
	'nl2'	12	127	84	8	62	30	212
	'ge1'	15	145	96	0	0	0	175
	'ge2'	28	298	197	0	0	0	495
	'gb'	28	297	198	10	63	28	482
	Total	92	871	631	29	185	81	1591
Movies	'ge1'	36	486	337	35	203	99	890
	'ge2'	17	163	122	10	93	54	314
	'nl'	24	335	258	83	588	297	1352
	'us_dig'	48	950	726	11	102	43	1208
	'us_ana'	28	337	239	50	404	178	1176
	Total	153	2271	1682	189	1390	661	4940

Table 27. SRS and CC statistics for series and movies.

Genre	Ratio # SRSs:CCs	Shots member of SRSs	Shots member of CCs	Average length of SRS	Average length of CC	Ratio SRS links : number of SRS shots	Ratio CCS links : number of CC shots
Series	~ 3 : 1	60%	13%	9.5 shots	6.4 shots	0.7	0.4
Movies	~ 0.8 : 1	46%	28%	14.8 shots	7.4 shots	0.74	0.48

Key frame pair similarity analysis methods for parallel shot detection

In this section we will present the methods we developed to identify and index parallel shots. Accordingly to the definition of parallel shots, this task consists in detecting similar shots time-distanced by at least on dissimilar shot.

We will declare two shots similar, if their representative key-frames are similar in a specified feature space according to a specified similarity criterion. Hence, in this section we will consider four methods we developed for measuring key-frame similarity (i.e. HSV, HY, ScaFT, SIFT based key frame pair similarity analysis) for parallel shot detection. Subsequently, we will benchmark the methods using our AV corpus and specify the winning method in more detail.

HSV based key frame pair analysis

In order to make a proper choice for the first key frame pair similarity analysis method, several different color spaces and feature models have been evaluated, including uniform quantized histogram intersection models with (a) global histogram or (b) local histogram, (c) non-uniformly histogram intersection models, (d) color coherence vectors and, finally, with (e) color auto-correlogram, which led to the selection of HSV-based uniform histogram intersection method with spatial separation. For this purpose video frames, e.g. YUV coded ones, are converted into HSV (*hue*, *saturation* and *value*). Subsequently, three spatial frame areas are defined, as sketched in Figure 82, i.e. (a) the entire frame further references as global area *GA*, (b) the region of interest area further referenced as foreground *FG* for which the frame is cropped by 20% at the right, left and top, and, finally, (c) the remaining area further referenced as background area *BG*.

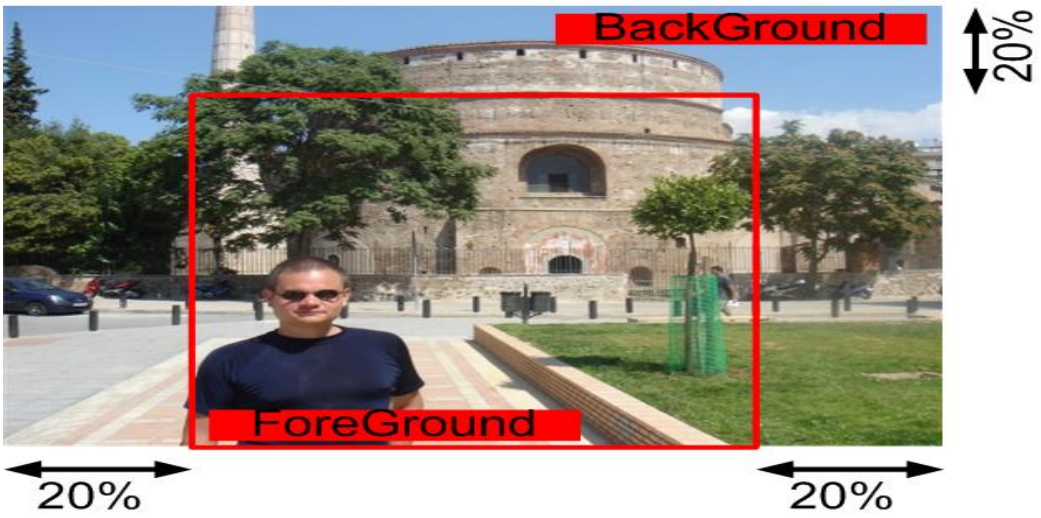


Figure 82. Spatial frame segmentation for HSV based key frame pair similarity analysis[1].

Moreover, for each of three frame areas (*areaє[GA,FG,BG]) of* each frame of the key frame pair (F_N, F_M) an uniformly distributed 256-bin HSV histogram is calculated, which is schematically shown for *HSV* $HIST_{k\_FN\_area}$ in Figure 83. The hue space is divided in l=16, saturation in m=4 and value in n=4 intervals, which results in k=256 uniform bins. A bin index k is obtained as

$$k = 16*l + 4*m + n, \quad l = [0..15],\ m = [0..3],\ n = [0..3],\ k = [0..255] \tag{4-41}$$

In order to evaluate the relationship, i.e. visual similarity, between two key frames (F_N, F_M) of two shots, various histogram distances were introduced in the past. For this work we select the histogram intersection distance *HID*, presented by Jeong in [126]. In his method the color histograms of two frames ($HistF_N$, $HistF_M$) are compared calculating the normalized sum of the lower bin values between two corresponding bins ($Hist(k,F_N)$, $Hist(k,F_M)$ across all bins,

$$HID(F_N, F_M) = \frac{\sum_{k=o}^{bin_{\max}-1} \min\left(Hist(k, F_N), Hist(k, F_M)\right)}{\min\left(\left|Hist(F_N)\right|, \left|Hist(F_M)\right|\right)} \tag{4-42}$$

with $|Hist(F_x)|$ representing the number of pixels present in frame (region) F_x, and binmax-1 representing the index of the highest bin. With the spatial separation in three areas this results in three *HIDs* (*HIDєf0…1]*), which are

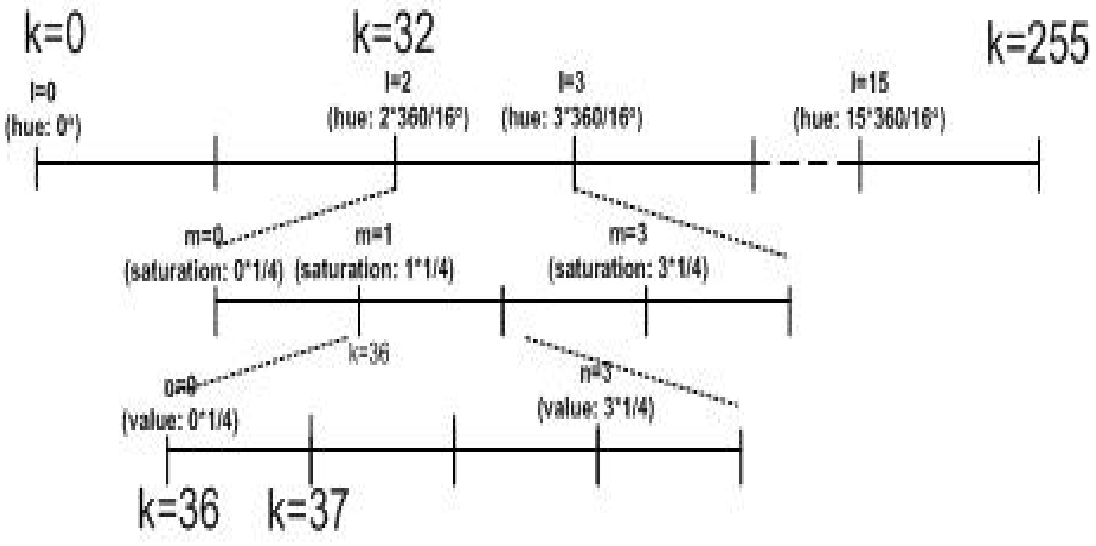


Figure 83.Uniformly distributed HSV histogram.

$$HID_{FG}\left(F_N\left(FG\right),F_M\left(FG\right)\right)=\frac{\sum_{k=o}^{bin_{\max}-1}\min\left(Hist_{FG}\left(k,F_N\right),Hist_{FG}\left(k,F_M\right)\right)}{\min\left(\left|Hist_{FG}\left(F_N\right)\right|,\left|Hist_{FG}\left(F_M\right)\right|\right)} \quad (4\text{-}43)$$

$$HID_{BG}\left(F_N\left(BG\right),F_M\left(BG\right)\right)=\frac{\sum_{k=o}^{bin_{\max}-1}\min\left(Hist_{BG}\left(k,F_N\right),Hist_{BG}\left(k,F_M\right)\right)}{\min\left(\left|Hist_{BG}\left(F_N\right)\right|,\left|Hist_{BG}\left(F_M\right)\right|\right)} \quad (4\text{-}44)$$

$$HID_{GA}\left(F_N\left(GA\right),F_M\left(GA\right)\right)=\frac{\sum_{k=o}^{bin_{\max}-1}\min\left(Hist_{GA}\left(k,F_N\right),Hist_{GA}\left(k,F_M\right)\right)}{\min\left(\left|Hist_{GA}\left(F_N\right)\right|,\left|Hist_{GA}\left(F_M\right)\right|\right)} \quad (4\text{-}45)$$

In a subsequent step, the HID with the highest value among the three HIDs, i.e. foreground, background and global, is selected, and stored as S_{prob},

$$S_{prob}\left(F_N,F_M\right)=\max\left(HID_{FG},HID_{BG},HID_{GA}\right) \quad with \quad S_{prob}\left(F_N,F_M\right)\in[0..1] \quad (4\text{-}46)$$

S_{prob} is then compared against a derived key frame pair similarity threshold $Th_{PS\_HSV}\epsilon[0\ldots1]$,

$$\begin{cases} S\left(F_N,F_M\right)=1 & if \quad S_{prob}\geq Th_{\text{PS\_HSV}} \\ S\left(F_N,F_M\right)=0 & if \quad S_{prob}<Th_{\text{PS\_HSV}} \end{cases} \quad (4\text{-}47)$$

Finally, two individual shots with a key frame pair similarity of $S(F_N,F_M)=1$ are indexed as sufficiently similar and, hence, linked together. They will be further used to cluster them into parallel shot sequences.

Hue-luminance (HY) based key frame pair analysis

Our study of color spaces unveiled that the maximal discriminative power of the YUV and HSV space of individual frames (F_N) is mainly contained in the values hue H and luminance Y_{lum}. Hence, instead of comparing key-frames in a single color system, we decided to use both, i.e. HSV and YUV, but using only their most discriminative components, i.e. H and Y. After normalization of hue to 0..360 degrees, and luminance to values between 0..255, hue and luminance plane similarity analysis are applied in this section on selected key frame pairs (F_N, F_M) of two selected shots, which are W_{sh} shots distanced from each other, for parallel shot detection. This results in hue- and luminance difference values $\Delta H(F_N,F_M)$ and $\Delta Y_{lum}(F_N,F_M)$, respectively. For the analysis the normalized hue and luminance planes F_N^H and $F_N^{Y_{lum}}$ of individual key frames (e.g. taking as example frame F_N with frame resolution x_{res} and y_{res}) are subdivided into p columns ($i\epsilon[0\ldots p-1]$) and q rows ($j\epsilon[0\ldots q-1]$), which results in p*q blocks (heuristically chosen) each indexed through i and j. Columns identified as letterboxes are excluded from sub-sequent steps here. Subsequently, for each block of the frame one normalized

average (mean) hue and luminance value is calculated, which results in two *p*q* matrices H^{F_N} and Y^{F_N},

$$H^{F_N} := \left(h_{i,j}^{F_N}\right)_{p*q}, Y^{F_N} := \left(y^{F_n}{}_{i,j}\right)_{p*q} \quad (4\text{-}48)$$

with H^{F_N}'s and Y^{F_N}'s matrix elements (mean /average values)

$$h_{i,j}^{F_N} = \frac{1}{xb * yb}\sum_{a=1}^{xb}\sum_{b=1}^{yb} F_N H\left(x = i * xb + a, y = j * yb + b\right) \quad (4\text{-}49)$$

$$y_{lum\,i,j}^{F_N} = \frac{1}{xb * yb}\sum_{a=1}^{xb}\sum_{b=1}^{yb} F_N Y\left(x = i * xb + a, y = j * yb + b\right) \quad (4\text{-}50)$$

where block width and height is defined by the resolution dependent parameter

$$xb = \left\lfloor \frac{x_{res}}{p} \right\rfloor, \quad yb = \left\lfloor \frac{y_{res}}{q} \right\rfloor \quad (4\text{-}51)$$

Hereafter, time-wise hue- and luminance derivatives, i.e. *ΔH and ΔY,* are calculated of selected key frame pairs (F_N, F_M) with

$$\Delta H\left(F_N, F_M\right) = \left|H^{F_N} - H^{F_M}\right|, \Delta Y\left(F_N, F_M\right) = \left|Y\left(F_N\right) - Y\left(F_M\right)\right| \quad (4\text{-}52)$$

In a sub-sequent step, the individual block based hue- and luminance matrix elements, $\Delta h(F_N,F_M)_{(i,j)}$ and $\Delta y(F_N,F_M)_{(i,j)}$ of the matrices $\Delta H(F_N,F_M)$ and $\Delta Y(F_N, F_M)$, are compared with empirically derived thresholds Th_{HY_hue} and Th_{HY_lum},.

Finally, the comparison resulted in a set of binary value elements representing key frame pair block-by-block similarities, i.e. $r(F_N,F_M)_{(i,j)}$, with

$$\forall(i,j): r\left(F_N, F_M\right)_{(i,j)} \begin{cases} =1 & if \quad \left(\Delta h\left(F_N, F_M\right)_{(i,j)} < Th_{hue}\right) \cap \left(\Delta y\left(F_N, F_M\right)_{(i,j)} < Th_{lum}\right) \\ =0 & else \end{cases} \quad (4\text{-}53)$$

The latter elements are used to calculate a normalized key frame pair similarity S_{prob},

$$S_{prob}\left(F_N, F_M\right) = \frac{1}{p * q}\sum_{0}^{p-1}\sum_{0}^{q-1} r\left(F_N, F_M\right)_{(i,j)} \quad with \quad S_{prob}\left(F_N, F_M\right) \in [0..1] \quad (4\text{-}54)$$

which are compared against a derived key frame pair similarity threshold Th_{PS_HY},

$$\begin{cases} S\left(F_N, F_M\right) = 1 & if \quad S_{prob} \geq Th_{\mathrm{PS\_HY}} \\ S\left(F_N, F_M\right) = 0 & if \quad S_{prob} < Th_{\mathrm{PS\_HY}} \end{cases} \quad (4\text{-}55)$$

Finally, two individual shots with a key frame pair similarity of $S(F_N,F_M)=1$ are indexed as sufficiently similar and, hence, linked (clustered) together, as explained in more detail later.

ScavFT (scale variant feature transform) based key frame pair analysis

While HSV and HY based key-frame similarity analysis allow measuring the similarity on minimal block resolution, in this section we are interested in a local similarity using distinctive landmark points. The latter are also called feature points *FP or salient points*, which are robust, distinctive, but also scale variant. We further reference the method as *scale variant feature transformation* 'ScaFT' method. The distinctive nature of feature points enables a reliable way to localize and track them across e.g. key frame pairs (e.g. F_N to F_M). Feature points were primarily used thus far to estimate camera-motion in compressed video, as described by Kuhn in [127], or uncompressed video, as explained by Matsuyama in [128]. Feature Points were also applied in image retrieval applications, as described by Sebe in [129]. The feature point selection procedure may take several forms. One of the widely applied methods was the Harris corner detector [130]. Modern techniques claim to improve upon the Harris detector by selecting points other than corners. Sebe for example used wavelets in [129] to select feature points. We chosed for ScaFT a feature point detection method proposed by Shi and Tomasi described in [131], which was based on the *minimum eigenvalue* of a 2-by-2 gradient matrix in combination with Newton-Raphson method for tracking. In our work the goal was to apply feature point selection and tracking from a video viewpoint. This was akin to the image retrieval applications of [129], since a video sequence is merely a sequence of images. Hence, for key frame pair similarity analysis for parallel shot detection, first of all, feature points were selected and, thereafter, were tracked from frame to frame through a video sequence, as shown in Figure 84 left.

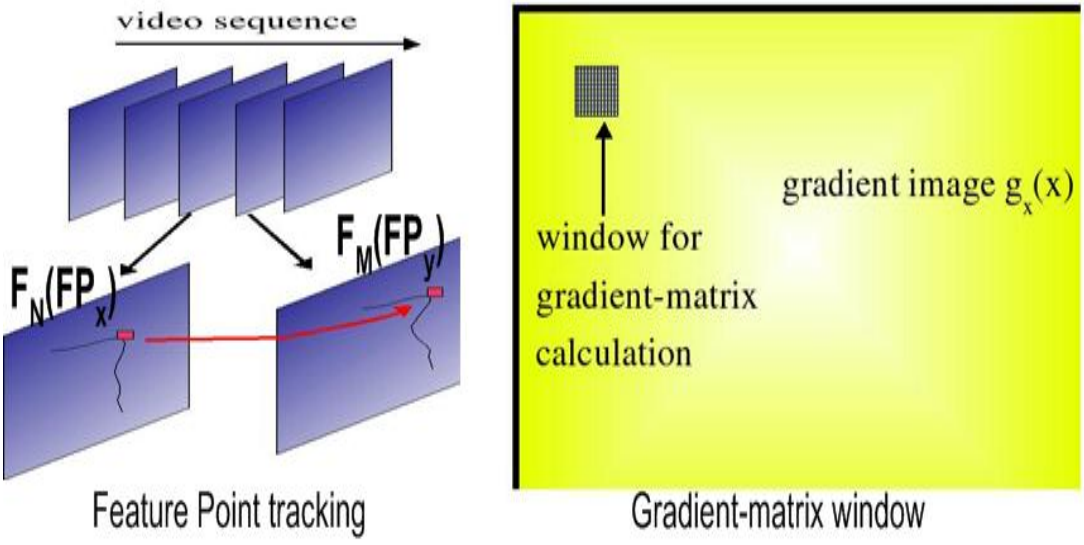


Figure 84. Feature point tracking and gradient-matrix of gradient image.

Scale-variant feature point selection

Hence, for the selection of feature points in selected frames, e.g. F_N of a video sequence, we applied the methods of Shi-Thomasi. In a first pre-calculation step individual RGB or YUV frames (see Figure 85 upper left) were reduced to their accordant luminance (Y) plane, as shown in Figure 85 (upper right, described in Annex 1). Subsequently, we calculate the minimum eigenvalue of a *gradient matrix*, as in [131], which we derive using a window around individual pixels in the Y plane of the frame, i.e. Y_{FN} (Figure 84 right). To compute the gradient matrix we first calculate the horizontal and vertical gradients of Y_{FN}, derived by convolving Y_{FN} with a high-pass filter. We used here a derivative of a Gaussian filter. The horizontal gradients are obtained by convolving the high-pass impulse response row-wise on the pixels of Y_{FN} and the vertical gradients by convolving it column-wise, respectively. The continuous zero-mean, unity-variance Gaussian is given by

$$f(x) = e^{-\frac{x^2}{2}} \qquad \text{and its derivative} \qquad \frac{d}{dx} f(x) = -x * e^{-\frac{x^2}{2}} \tag{4-56}.$$

which forms the basis for the discrete time function,

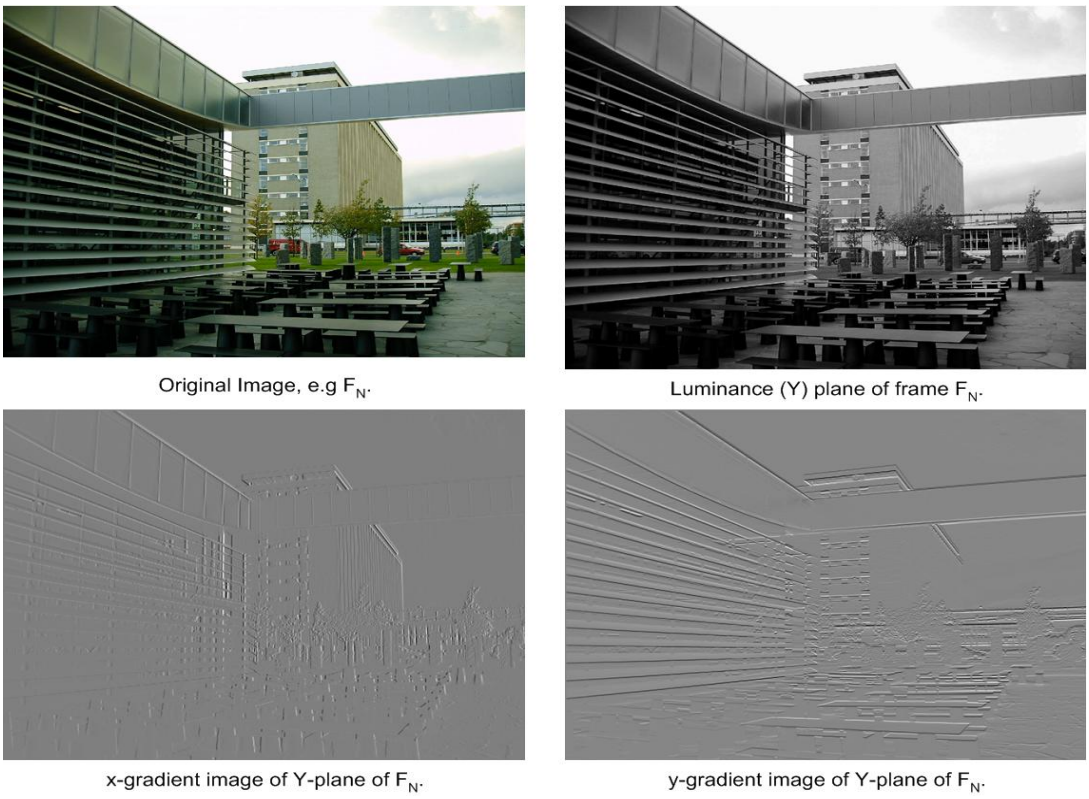


Figure 85. Gradient image generation for feature point analysis[1].

$$f(n) = -n * e^{-\frac{n^2}{2}} \qquad (4\text{-}57),$$

and which is applied as kernel for the convolution. As a result of this operation, two gradient images, $g_x(x,y)$ and $g_y(x,y)$, (shown in Figure 85 lower left and Figure 85 lower right) are obtained, which are used to calculate the gradient matrix *G*,

$$G = \begin{bmatrix} \sum_{window} g_x(x,y)^2 & \sum_{window} g_x(x,y)\,g_y(x,y) \\ \sum_{window} g_y(x,y)\,g_x(x,y) & \sum_{window} g_y(x,y)^2 \end{bmatrix} = \begin{bmatrix} g_{xx} & g_{xy} \\ g_{yx} & g_{yy} \end{bmatrix} \qquad (4\text{-}58).$$

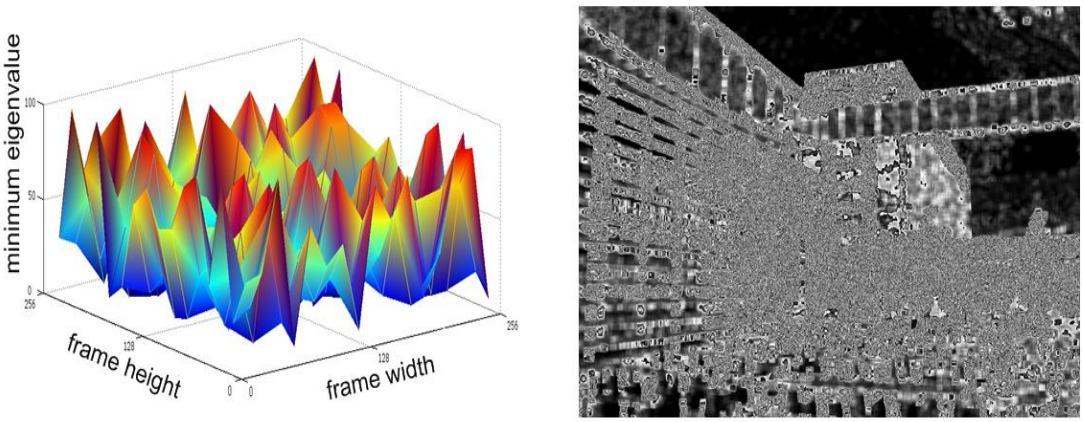
For each individual pixel the matrix G is calculated with a chosen window area *WA* of e.g. 7*7[10] around the pixel, as shown in Figure 84 right. Thereafter, the strength of each individual pixel, i.e. its suitability to form a robust feature point, is measured by means of the minimum eigenvalue of the gradient matrix. The eigenvalues are derived using

$$G * \underline{x} = \lambda * \underline{x}, \quad \begin{bmatrix} g_{xx} & g_{xy} \\ g_{yx} & g_{yy} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \Rightarrow \begin{pmatrix} \lambda_+ \\ \lambda_- \end{pmatrix} \Rightarrow \min\ eigenvalue \qquad (4\text{-}59),$$

with $\underline{x}$ representing the eigenvector, which finally result in the eigenvalues λ_+ and λ_-

$$\lambda_\pm = \frac{1}{2}\left[(g_{xx} + g_{yy}) \pm \sqrt{4 * g_{xy} * g_{xy} + (g_{xx} + g_{yy})^2} \right] \qquad (4\text{-}60).$$

All individual pixels are represented by their location, e.g. $F_N(x,y)$, and their minimum eigenvalue (z-axis in a three-dimensional plot), as sketched in Figure 86 left.



Minimum eigenvalue of Y-plane of F_N.

Minimum eigenvalue (black-to-white with increasing eigenvalue) in 2D.

Figure 86. Minimum eigenvalue results for feature point analysis[1].

[10] Any window size was applicable.

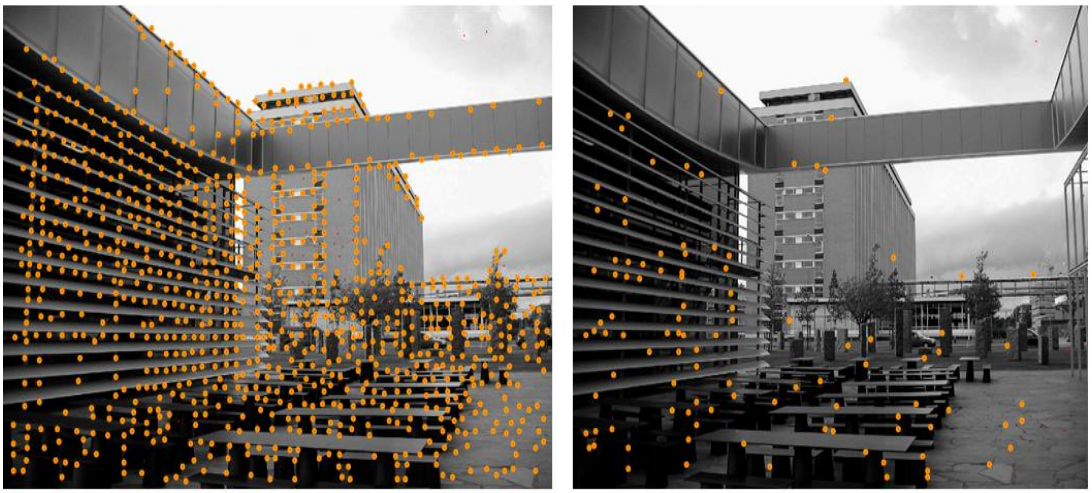
The calculated minimum eigenvalues of example image F_N (of Figure 85) result in the two dimensional representation of Figure 86 (right image).

Finally, pixels with the local highest minimum eigenvalue are selected – from best to worst - into a feature list, ensuring that new additions to the list are at least 10 pixels away in all four directions from all other pixels, which are already selected and added into the list. Pixels that do not meet this criterion are simply discarded. This results in a set of FP_{total_FN}, e.g. FP_{total_FN}=100, feature points containing only well-spaced trackable feature points, as shown in Figure 87 (left image).

Feature point tracking

After we identified feature points, we applied feature point tracking to be able to measure the frame-pair similarity in the sense that, if for a given set of feature points in frame F_N (i.e. FP_{total_FN}) a sufficiently high percentage of them was successfully tracked into frame F_M, then the pair of frames, i.e. (F_N, F_M), was indexed as similar. To track feature points we applied the method proposed in [131].

Firstly, gradient images g_x and g_y were calculated for both frames, F_N and F_M, which resulted in g_{x_FN}, g_{y_FN}, g_{x_FM} and g_{y_FM}. Subsequently, the procedure[11] '*tracking iteration*' was performed for all FP_{total_FN} feature points of F_N with a fixed number of iterations, updating the estimated location of the feature point in F_M after each iteration.



Selected feature points (small dots) of Y-plane of $F_N(x,y)$.

Tracked feature points (small dots) into Y-plane of $F_M(x,y)$.

Figure 87. Feature point selection and tracking for key frame pair similarity analysis[1].

[11] The procedure is applicable as long as the displacement vector of related feature points of F_N and F_M is limited.

Tracking iteration {start}

At the beginning of each iteration the horizontal and vertical gradient sums, s_x and s_y, were calculated within a window (we apply here a 7x7 window[12]) around the current feature point by summing the corresponding gradient values of F_N and F_M,

$$s_x(x,y) = g_{x\_F_N}(x,y) + g_{x\_F_M}(x,y)$$
$$s_y(x,y) = g_{y\_F_N}(x,y) + g_{y\_F_M}(x,y) \quad (4\text{-}61).$$

Hereafter, the gradient sum matrix S is calculated,

$$S = \begin{bmatrix} \sum_{window} s_x(x,y)^2 & \sum_{window} s_x(x,y)s_y(x,y) \\ \sum_{window} s_y(x,y)s_x(x,y) & \sum_{window} s_y(x,y)^2 \end{bmatrix} = \begin{bmatrix} s_{xx} & s_{xy} \\ s_{yx} & s_{yy} \end{bmatrix} \quad (4\text{-}62).$$

To derive the displacement vector $\underline{d}$ the following steps were taken. Firstly, the 2-by-1 error vector $\underline{e}$ in the equation $S*\underline{d}=\underline{e}$ was minimized,

$$\begin{bmatrix} s_{xx} & s_{xy} \\ s_{yx} & s_{yy} \end{bmatrix} \begin{bmatrix} d_x \\ d_y \end{bmatrix} = \begin{bmatrix} e_x \\ e_y \end{bmatrix} \quad (4\text{-}63),$$

as described in [131], to arrive to an estimate for the position of the feature point in F_M, with $\underline{d}$ representing the displacement vector.

By applying Cramer's rule, the equation was solved with[13]

$$d_x = \frac{s_{yy}e_x - s_{yx}e_y}{s_{xx}s_{yy} - s_{yx}s_{xy}}$$
$$d_y = \frac{-s_{xy}e_x + s_{xx}e_y}{s_{xx}s_{yy} - s_{yx}s_{xy}} \quad (4\text{-}64),$$

which resulted in the displacement vector $\underline{d}$. At the end of the iteration $\underline{d}$ was added to the estimated feature point location in F_M. Nevertheless, shortcomings of this gradient-based motion estimation method, i.e. only small displacements were trackable, led incidentally to remaining displacement errors. Hence, for large displacements other methods were applicable such as multiscale schemes proposed by Anandan in [132], or regression-based methods mentioned by Patras in [133], but especially Lowe's SIFT method using difference descriptors to measure the goodness of a match as described in [134].

Tracking iteration {end}

[12] Any window size was applicable, but the author believes that improvements would be probably feasible with a displacement and motion dependent window area WA.

[13] With the assumption that **S** is non-singular, i.e. the denominator of the above equation is unequal to zero.

Table 28. Evaluation of optimal number of '*tracking iterations*'.

# iterations	1	2	3	4	5
Displacement of FPs	2.51	2.05	0.52	0.14	0.08

We derived the optimal number of '*tracking iterations*' by calculating the average displacement value across an entire set of video items, i.e. news, soaps, movies. The feature point displacement decreases with increasing iterations. The optimal number of 'tracking iterations' is, hence, chosen with five, as derived from Table 28. After five iterations the determinant of the matrix decreases so much that further calculations are not feasible anymore. Hence, five iterations prove to be sufficient for tracking the feature point to its final location, as shown in Figure 88.

After the iteration process has been stopped the correctness of the tracking of FP_x is verified calculating the normalized sum of the displacement frame intensity differences D,

$$D(FP_x) = \frac{1}{|WA|} \sum_{window\ FP_x} \left| Y_{F_N}(x, y) - Y_{F_M}(x + dx, y + dy) \right|, \quad D \in [0 \ldots 255] \tag{4-65}$$

in the selected window between the original feature point window in F_N and its tracked counterpart in F_M, respectively. The tracking is labeled as successful if D do not exceed a heuristically chosen threshold of $Th_{window}=20$[14],

$$\begin{cases} FP_x\_tracked = 1 & if \quad D(FP_x) < Th_{window} \\ FP_x\_tracked = 0 & if \quad D(FP_x) \geq Th_{window} \end{cases} \tag{4-66}$$

and, finally, the percentage of successfully tracked feature points S_{prob},

$$S_{prob}(F_{N,} F_M) = \frac{1}{FP_{total\_FN}} \sum_{x=1}^{FP_{total\_FN}} FP_x\_tracked, \quad S_{prob}(F_{N,} F_M) \in [0 \ldots 1] \tag{4-67}$$

is compared against a derived key frame pair similarity threshold Th_{PS_ScaFT},

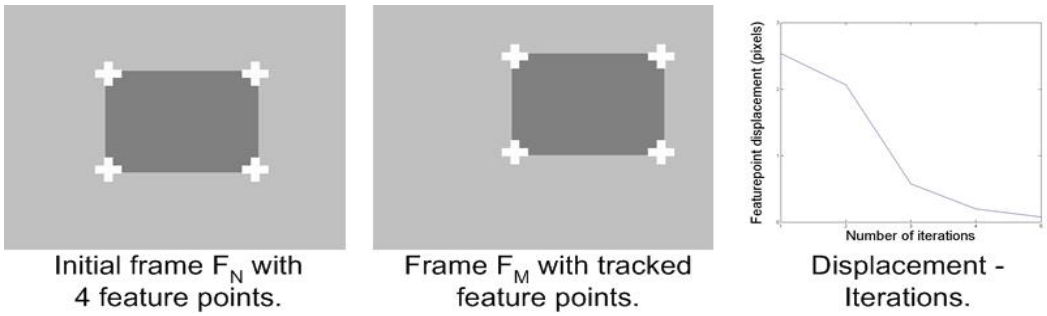


Figure 88. Number of tracking (displacement) iterations.

[14] A trade-off between insufficient correct and too many false trackings.

$$\begin{cases} S(F_N, F_M) = 1 & if \quad S_{prob} \geq Th_{\text{PS\_ScaFT}} \\ S(F_N, F_M) = 0 & if \quad S_{prob} < Th_{\text{PS\_ScaFT}} \end{cases} \quad (4\text{-}68),$$

Finally, two individual shots with a key frame pair similarity of $S(F_N,F_M)=1$ are indexed as sufficiently similar and, hence, linked (clustered) together, as explained in more detail later. For our analysis we chose Th_{PS_ScaFT}=10% (see Table 30).

SIFT (scale invariant feature transform) based key frame pair analysis

The shortcoming of ScaFT is its sensitivity to scaling and its unreliable, i.e. coarse, tracking using only intensity difference *D* as check. We, therefore, choose as 4th method for key frame pair similarity analysis the SIFT method, i.e. *scale-invariant feature transform* SIFT, published by Loewe in [135]. The latter used not only differential scale-space representations, i.e. $D_{FN}(x,y,\sigma)$, of the luminance plane of individual frames e.g. F_N, i.e. Y_{FN}, and selected extrema in this scale-space, i.e. $D_{FN}(x,y,\sigma)>\alpha$, as feature points, but also gradient magnitude $m_{FN}(x,y)$ and gradient orientation $\Theta_{FN}(x,y)$ sets within a certain window around the selected feature point as signature to improve the matching robustness between SIFT feature points between e.g. two frames F_N and F_M, as described in more detail in Annex 4.

Hence, applying Lowe's method, we identify all SIFT feature points per frame, as described in Annex 4. Straight after, we index all SIFT feature points between selected key frames pairs (F_N,F_M) as matching ($SIFT\_FP_x\_tracked=1$), if the normalized Euclidian distance $D(SIFT\_FP_x)$ falls short compared to a defined threshold $Th_{SIFTPoint}$, similarly to equation (4-68), with

$$\begin{cases} SIFT\_FP_x\_tracked = 1 & if \quad D(SIFT\_FP_x) < TH_{SIFTPo\,\text{int}} \\ SIFT\_FP_x\_tracked = 0 & if \quad D(SIFT\_FP_x) \geq Th_{SIFPo\,\text{int}} \end{cases} \quad (4\text{-}69).$$

An example of matching SIFT feature points is shown in Figure 89.

Figure 89. F_N and F_M Y plane with tracked SIFT feature points superimposed[1].

Next, we define the total amount of SIFT feature points in frame F_N by $SIFT\_FP_{total\_FN}$ and calculate the percentage of successfully tracked feature points $S_{prob}(F_N,F_M)$ with

$$S_{prob}(F_N,F_M)=\frac{1}{SIFT\_FP_{total\_FN}}\sum_{x=1}^{SIFT\_FP_{total\_FN}} SIFT\_FP_x\_tracked,\quad S_{prob}(F_N,F_M)\in[0\ldots1] \qquad (4\text{-}70).$$

Here after, we compare the probability $S_{prob}(F_N,F_M)$ against a derived key frame pair similarity threshold Th_{PS_SIFT} and, if the threshold is exceeded the key frame pair (F_N,F_M) is indexed as matching ($S_{prob}(F_N,F_M)=1$), with

$$S(F_N,F_M)=\begin{cases}1 & if \quad S_{prob}\geq Th_{PS\_SIFT}\\ 0 & if \quad S_{prob}<Th_{PS\_SIFT}\end{cases} \qquad (4\text{-}71),$$

Finally, we link (cluster) all individual shots containing matching key frame pairs ($S_{prob}(F_N,F_M)=1$) together. For the threshold value we chose Th_{PS_SIFT}=15% (Table 30).

Comparison and evaluation of parallel shot detection results

To compare and evaluate the reliability of the four key frame similarity analysis methods for parallel shot detection we use the manual annotated parallel shot links as ground truth. That means that we calculate recall and precision of detected parallel shot links (key frame links) and use them as measure for comparison. For all four methods we define two variable parameters, i.e. window size $W_{sh}\epsilon[2..15]$ measured in shots and the absolute key frame pair similarity threshold $Th_{PS}\epsilon[0...1]$.

Hence, parallel shots are established using subsequently all four key frame pair analysis methods varying the two threshold W_{sh} and Th_{PS}. This means that within a certain window range of $+/-W_{sh}$ shots the key frame of the current shot e.g. first key frame of shot x (sh_x) is compared with the last key frame of each predecessor shot within this window, as depicted in Figure 90. The similarity value is then compared to threshold Th_{PS}.

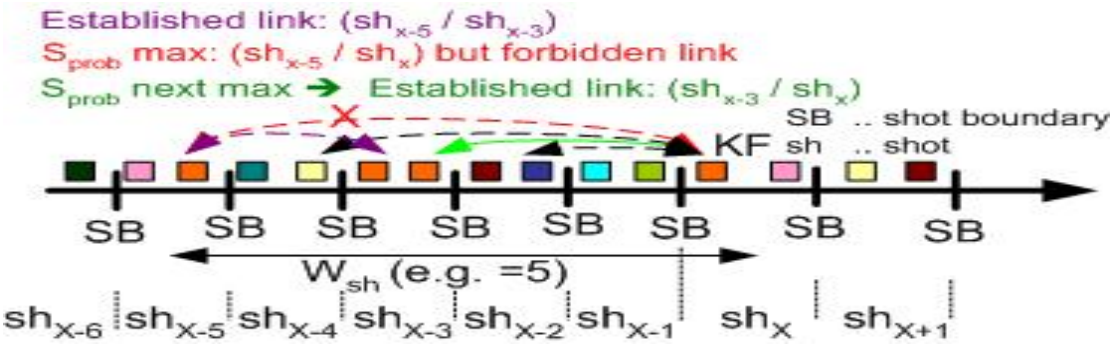


Figure 90. Key frame pair analysis for parallel shot detection with W_{sh} and Th_{PS}.

Next, the backward-based key frame pair similarity with the maximal probability S_{Prob} is selected, but shots to which already a link has been established are excluded from this process, as shown for the case 'forbidden link' in Figure 90. The two shots with the maximal probability are linked together if S_{Prob} does not fall short Th_{PS}. Hereafter, sequence with interleaved shots, i.e. interleaved connections of linked shots, are identified and their start and end shot indexed as boundaries of the parallel shot sequence, as shown in Figure 91. In the optimal case the boundaries correlate exactly with the ground truth of parallel shots (interleaved narrative events). But, before starting the robustness evaluation of the four methods, we define benchmarking criteria. Originally, we started with the traditional way of calculating recall and precision based on the total shots clustered correctly into parallel shot sequence, as sketched in Figure 92. But the fuzziness of the traditional benchmark approach, i.e. limited feedback acquired about the real linking, which we find more objective and relevant, justifies to define two other benchmark criteria, i.e. recall and precision based on links only and, in addition, for link through, as sketched in Figure 93.

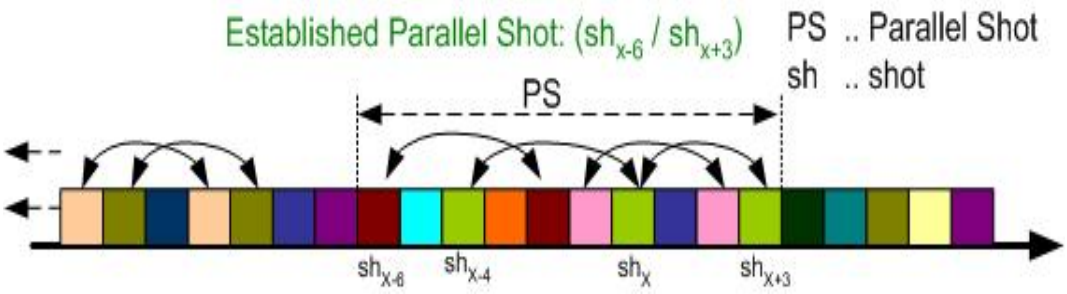


Figure 91. Established parallel shot.

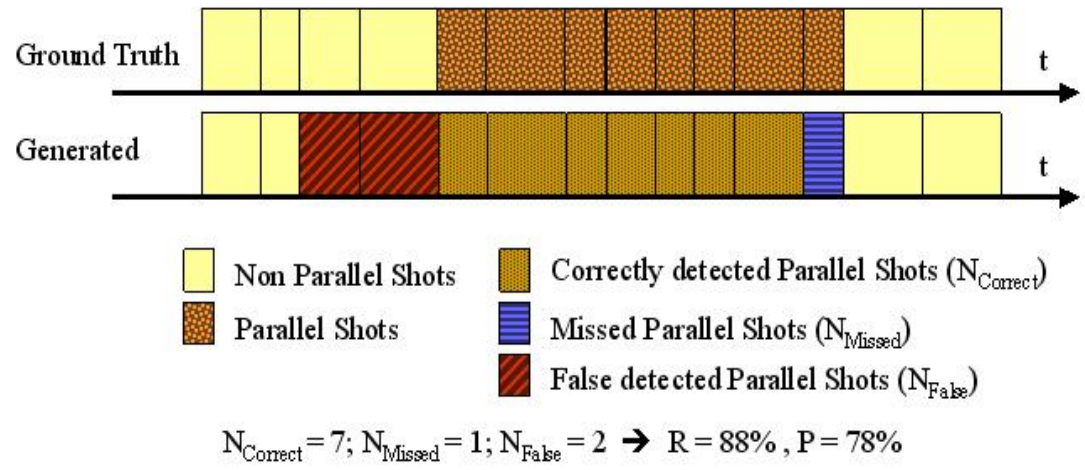


Figure 92. Traditional way of calculating recall and precision for parallel shot evaluation.

We apply these two benchmark criteria in the subsequent analysis for robustness of four parallel shot detector methods, i.e. those based on HSV, HY, ScaFT and SIFT key frame pair similarity.

For the HY based parallel shot analysis method we specify three additional parameters before starting with the key frame pair analysis, i.e. Th_{HY_hue}, Th_{HY_lum} and resolution (p,q), as described in 4.5.2. Due to the visual sensitivity in hue a very small Th_{HY_hue}, differential value threshold, is expected to perform well, in contrary to luminance, which is expected to lie in the range of about 10% of the total luminance range, i.e. Th_{HY_lum}. Furthermore, the theoretically optimal resolution is expected to be a trade-off between recall and precision, because with increasing resolution the recall should drop (i.e. due to increasing variation sensibility), but on contrary precision should increase (i.e. less false link detection due to stricter comparison). The expected theoretical behaviour is verified through iteratively and sub-sequentially (one-by-one) varying the parameter and monitoring the robustness changes of the parallel shot linker. The criteria for the proper settings are to maintain high precision and, hence, to minimize the chance to cross scene boundaries. With a given window length $W_{sh}=3$ and threshold $Th_{PS}=30$ the three parameter are tuned, i.e. $Th_{HY\_hue}\epsilon[3\ldots disabled]$, $Th_{HY\_lum}\epsilon[3...disabled]$ and the resolution $p*q\epsilon[8*8\ldots30*30]$, and the results are summarized in Table 29.
The analysis unveils that the hue thresholds has to be chosen very restrictive, i.e. $Th_{HY\_hue}=7$, and in contrary the luminance difference threshold performs well in the expected range, i.e. $Th_{HY\_lum}=20$. The resolution behaves as predicted, i.e. with increasing resolution recall decreases and precision increases. Hence, a trade-off value is selected with reasonable recall and a low number of scene boundaries crossed, i.e. no scene boundary should be crossed by a parallel shot sequence.

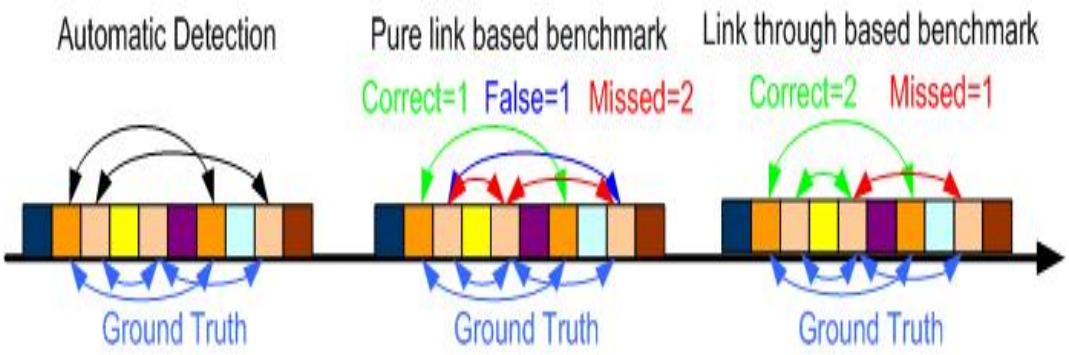


Figure 93. Parallel shot detection benchmark definitions – link and link through.

Table 29. Parameter evaluation for the HY based parallel shot detector method.

	W_{sh} [shot]	Th_{PS} [%]	Th_{HY_hue}	Th_{HY_lum}	p*q	Correct Links	False Links	Missed Links	Crossed ScB	Re [%]	Pr [%]
Th_{HY_hue}	3	30	**3**	10	16*16	1600	161	1450	2	52.5	90.9
	3	30	**7**	10	16*16	2091	356	959	8	68.6	85.5
	3	30	10	10	16*16	2261	505	789	15	74.1	81.7
	3	30	Disabled	10	16*16	3026	2764	624	226	82.9	52.3
Th_{HY_lum}	3	30	5	**10**	16*16	1697	181	1353	4	55.6	90.4
	3	30	5	**20**	16*16	2052	334	998	7	67.3	86.0
	3	30	5	25	16*16	2147	404	903	10	70.4	84.2
	3	30	5	Disabled	16*16	2291	981	664	80	77.5	70.0
Res p.q	3	30	5	10	8*8	2105	423	945	13	69.0	83.3
	3	30	5	10	10*10	2083	338	697	7	68.3	85.6
	3	30	5	10	**12*12**	2051	348	999	7	67.3	85.8
	3	30	5	10	16*16	1912	263	1138	6	62.7	87.9
	3	30	5	10	20*20	1919	263	1131	6	62.6	88.0
	3	30	5	10	23*25	1793	209	1257	4	58.8	89.6
	3	30	5	10	30*30	1865	242	1185	4	58.2	89.8

The resolution *p*qϵ[12*12]* is, therefore, selected, because only scene boundaries at gradual transitions are crossed. The cause of the latter is the motivation to include, subsequently, gradual transitions as well. Hence, the final three values for the HY based method are $Th_{HY\_hue}=7$, $Th_{HY\_lum}=20$ and a resolution of $p=12$ / $q=12$.

Now we can start with the benchmark of the four methods. The results of the analysis are summarized in Figure 94, where the first row covers the results for series and the second one those for movies. The detection results for *shot reversed shots* SRS and *cross-cuttings* CC, respectively were summarized in the 1st and 2nd column of Figure 94.

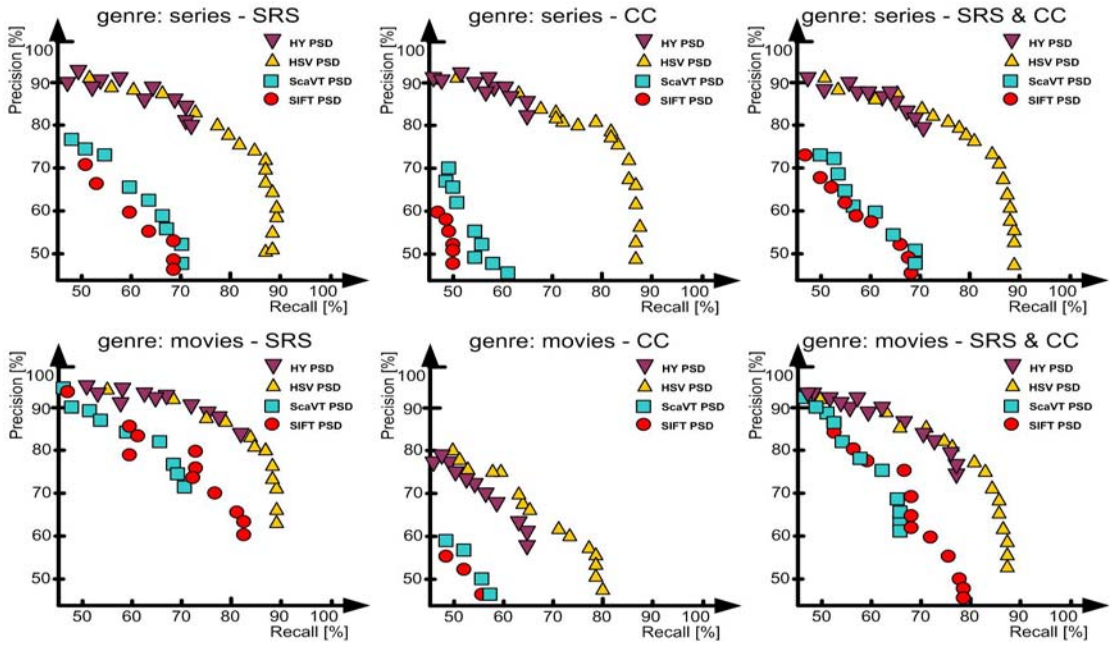


Figure 94. Shot-reveres-shot and cross-cutting benchmark based on shot links.

The analysis unveils that the detection robustness of dialogue sequences, i.e. shot reverse shot sequences, exceed that of cross-cuttings, based on the fact that dialogues appear during less vivid content sequences compared to motion loaded cross-cuttings. The motion-loaded nature of cross-cuttings results in frequent foreground and background changes especially in genres like movies, as reflected in low recall and precision in the graphs 'movies – CC' in Figure 94. The recall and precision of shot reverse shot link detection for both, movies and series, outperforms, therefore, those for cross-cutting link detection. Interesting enough is the fact that both landmark point based methods, i.e. ScaFT and SIFT, perform at least by 10% better for shot reverse shot link detection in movies than in series. The evaluation shows that dialogues in series have a strong focus on the individuals involved in the dialogue (e.g. close up shots) leaving little chances to identify reliable feature points in rigid areas, whereas the individuals in movie dialogues are more distant (e.g. medium long shots) and, hence, more rigid background areas / objects are in focus. Because of the final aim of scene boundary detection, the ambition is to cluster shots together with parallel shot detection, but to avoid that parallel shots sequences cross scene boundaries. The analysis of the results of Figure 94 shows that ~18 of the 240 AV corpus scene boundaries are crossed. This happens mainly during gradual transitions, which we originally excluded in the first benchmark analysis.

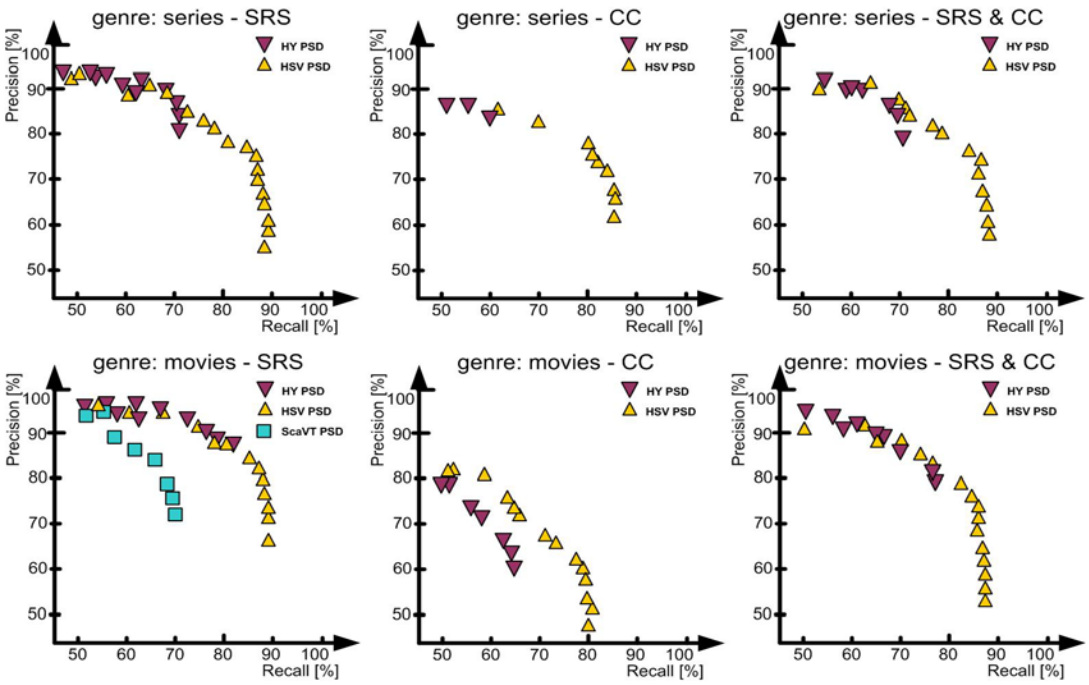


Figure 95. Shot reverse shot and cross-cutting benchmark based on shot links with exact gradual transition boundaries.

Table 30. Evaluation of parallel shot detection methods.

PSD method	W_{sh} [shots]	Th_{PS} [0..1]	Re [%]	Pr [%]
HSV	5	75	83.9	83.6
HY	5	30	69.6	87.0
ScaFT	5	10	53.5	75.1
SIFT	5	15	30.9	84.5

Table 31. Evaluation of parallel shot detection aiming for optimal link trough results.

PSD method	W_{sh} [shots]	Th_{PS} [%]	Correct	False	Missed	Missed ScB	Re [%]	Pr [%]
HSV	5	75	2429	375	775	3	75.8	86.6
HY	5	35	2285	333	919	7	71.3	87.3

Hence in the second iteration, we apply both shot boundary types, i.e. cuts and gradual transitions, to re-evaluate the robustness of the parallel shot detector with the two best performing methods, i.e. HSV and HY. These results of the parallel shot linker are summarized in Figure 95. Even so the ground truth analysis unveils that editors insert up to six shots between correlating shots, i.e. $W_{sh\_max}=7$, a window size of $W_{sh}=5$ results in a good recall/precision trade-off, as depicted in Table 30. These results show that the focus on mostly non-rigid areas, i.e. subjects in dialogues, in the AV content are the major reason for the low recall of the two landmark based detection methods, i.e. ScaFT and SIFT (see Table 30), which are very much dependend on the presents of rigid structures within the focus. On contrary, both color based methods, i.e. HSV and HY, reached comparable high precision (*Pr~90%* at *Re~70%*). But, because the aim is to cluster linked shots into parallel shot sequences the pure link method, as depicted in Figure 97, is less suited for our purpose. The method is too strict compared to the link through method, because the latter allows missing individual links as long as one of the next links is correctly detected, which is more relevant for our purpose. The results of the link through are summarized in Table 31, and reach *Pr~86%* at *Re~76%.*

Analysis of parallel shot detector results

In order to be able to understand the individual strengths and weaknesses of the four methods, we do a detailed analysis of the missed and false link detections. We apply the individual chosen settings for the four methods as given in Table 30.

The evaluation of the HSV and HY based methods unveils that the main reasons for missed link detection are (a) non-captured camera zoom-in or zoom-out between related shots but without panning or tilting further referenced as ‘virtual zooms’, (b) non-

captured object motion, e.g. same subject or object in a different position or pose in mainly medium to close shots, further referenced as 'virtual object motion', (c) non-visible camera motion between correlated shots, e.g. same object/person in a different setting or background, further called 'virtual camera motion' (d) or missed links due to illumination/color changes, e.g. the object of interest passed through settings with different illumination, such as an tunnel or a weather front. We call the non-captured camera and object transitions as 'virtual', because the director has the intention to create the impression of e.g. a camera zoom-in, but do not captures this activity. But also systematic reason lead to miss detections such as links bigger than $W_{sh}=5$.

We divide false detections for the HSV and HY based methods mainly into two groups, i.e. *systematic* and *semantic* errors. The semantic errors occur mainly due to the strict ground truth annotation, which are based on the rule that key frame pairs have to exhibit major visual similarities. The systematic false detections are split into false detection due to (a) similar color ranges, i.e. non-related key frame pairs have similar color compositions, and (b) dark sequences, i.e. that little information is present due to very dark settings. Some false detection has no obvious correlation between the linked key frame pairs, hence, and these are clustered into the cluster 'no obvious similarity'.

The false and missed detection analysis of the ScaFT and SIFT based methods is rather difficult due to the complex nature of these methods and, hence, the explanation is given using visual examples.

Evaluation of HSV based parallel shot detector

The analysis of the in total 795 missed links using the HSV based method unveils that links are missed to 30% because of virtual zooms, i.e. non-captured zooms between connected shots, to 30% due to prominent foreground objects being abruptly displaced between connected shots, to 10% based on non-captured camera motion and to 10% due to significant illumination or colour changes, summarized in Table 32.

The analysis of the in total 736 falsely identified links using the HSV method shows, as summarized in Table 33, that about 17%+18%+10%+9%=51% (i.e. similar color ranges, dark sequences and no obvious similarity) of the false detections are based on systematic shortcomings of the method. On contrary, 11%+15%+3%+6%+14% =49% of falsely identified semantic error links are due to the very strict ground truth annotation rules. The parallel appearance of missed on false detections based on semantic links, i.e. for example the non-captured abrupt zoom between semantically connected shots, is the main reason that we re-define the annotation rules for the semantic ground truth.

Table 32. Missed links of HSV based parallel shot detection in series and movies.

Series	'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total
Virtual zoom	3	17	9	25	14	68
Virtual object motion	2	5	4	25	16	52
Virtual camera motion	1	0	2	10	3	16
Illumination / color change	0	1	1	0	9	11
Link size >5	9	6	2	4	5	26
False ground truth	0	0	2	1	1	4
Virtual zoom & object motio	4	3	6	5	2	20
Total	19	32	26	70	50	197
Recall [%]	91.6	84.9	85.1	85.9	89.6	87.6
Movies	'nl'	'ge1'	'ge2'	'us_ana'	'us_dig'	Total
Virtual zoom	35	16	13	51	47	162
Virtual object motion	61	36	10	28	37	172
Virtual camera motion	13	10	9	26	21	79
Illumination / color change	15	17	3	20	40	95
Link size >5	12	9	3	5	23	52
False ground truth	1	2	0	2	5	10
Virtual zoom & object motio	3	4	2	8	11	28
Total	140	94	40	140	184	598
Recall [%]	84.3	70.1	97.0	88.1	84.8	87.9

Table 33. False links of HSV based parallel shot detection in series and movies.

Series		'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total
Systematic Errors	Similar color range	4	4	3	13	1	25
	Dark image sequence	0	1	0	0	0	1
	No obvious similarity	3	3	0	2	0	8
	Bridge link across virtual zoom lir	0	0	2	3	4	9
Semantic Errors (GT)	No visible change	5	1	1	7	5	19
	Virtual zoom	5	5	3	19	0	32
	Virtual camera motion	1	1	1	2	3	8
	Virtual camera motion and zoom	1	2	0	1	0	4
	Virtual object motion	3	4	3	4	12	26
Total		22	21	13	51	25	132
Precision [%]		90.4	89.6	92.0	89.3	94.5	91.4
Movies		'nl'	'ge1'	'ge2'	'us_ana'	'us_dig'	Total
Systematic Errors	Similar color range	24	25	0	36	16	101
	Dark image sequence	53	20	0	43	18	134
	No obvious similarity	19	13	0	26	5	63
	Bridge link across virtual zoom lir	4	7	2	6	15	34
Semantic Errors (GT)	No visible change	12	15	2	19	12	60
	Virtual zoom	16	11	1	34	19	81
	Virtual camera motion	4	2	0	7	2	15
	Virtual camera motion and zoom	6	3	1	24	7	41
	Virtual object motion	16	22	4	22	11	75
Total		154	118	10	217	105	604
Precision [%]		83.0	65.1	99.2	82.7	90.7	87.8

Evaluation of HY based parallel shot detector

The analysis of the in total 665 missed links of the HY based method unveils that the robustness is comparable to the HSV method.

Table 34. Missed links of HY based parallel shot detection in series and movies.

Series	'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total
Virtual zoom	5	11	9	20	14	59
Virtual object motion	2	2	5	21	18	48
Virtual camera motion	0	0	2	7	3	12
Illumination / color change	0	0	0	0	10	10
Link size >5	9	6	1	4	5	25
False ground truth	0	0	1	1	1	3
Virtual zoom & camera motion	2	2	3	9	0	16
Total	18	21	21	62	51	173
Recall [%]	92.1	90.1	88.0	87.5	89.4	89.1
Movies	'nl'	'ge1'	'ge2'	'us_ana'	'us_dig'	Total
Virtual zoom	27	13	14	39	28	121
Virtual object motion	47	34	15	16	26	138
Virtual camera motion	11	14	8	21	17	71
Virtual zoom & camera motion	4	4	2	7	7	24
Illumination / color change	12	19	0	14	24	69
Link size >5	14	8	3	12	23	60
False ground truth	1	2	0	1	5	9
Total	116	94	42	110	130	492
Recall [%]	87.0	70.1	96.9	90.7	89.2	90.0

Table 35. False links of HY based parallel shot detection in movies.

Series		'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total
Systematic Errors	Similar color range	4	6	4	14	6	34
	Dark image sequence	0	0	0	0	0	0
	No obvious similarity	4	3	0	3	0	10
	Bridge link across virtual zoom lir	0	1	1	3	3	8
Semantic Errors (GT)	No visible change	5	1	1	7	5	19
	Virtual zoom	5	5	4	20	0	34
	Virtual camera motion	1	0	1	2	3	7
	Virtual camera motion and zoom	1	2	0	2	0	5
	Virtual object motion	2	4	3	4	12	25
Total		22	22	14	55	29	142
Precision [%]		90.5	89.7	91.7	88.7	93.7	90.9
Movies		'nl'	'ge1'	'ge2'	'us_ana'	'us_dig'	Total
Systematic Errors	Similar color range	30	28	1	53	36	148
	Dark image sequence	66	20	0	47	14	147
	No obvious similarity	16	13	1	27	7	64
	Bridge link across virtual zoom lir	6	5	2	7	10	30
Semantic Errors (GT)	No visible change	12	18	2	22	13	67
	Virtual zoom	19	13	0	42	42	116
	Virtual camera motion	5	4	0	11	2	22
	Virtual camera motion and zoom	7	3	1	26	15	52
	Virtual object motion	17	22	4	28	22	93
Total		178	126	11	263	161	739
Precision [%]		81.3	63.6	99.2	80.2	87.0	85.8

The results are comparable mainly because of the similar nature of the two methods. Links are missed to 30% because of virtual zooms, to 30% due to prominent foreground objects being abruptly displaced between connected shots, to 10% based on non-captured camera motion and to 5% due to significant illumination or colour changes, as summarized in Table 34.
The analysis of 881 falsely identified links shows, summarized in Table 35, that about 21%+17%+8%+4%=50% of the false detections are based on systematic shortcomings of the HY method and about 10%+17%+3%+6%+14%=50% of the falsely identified links on the strict visual similarity based ground truth annotation rule.

The evaluation of the ScaFT- and SIFT-based parallel shot detector is of less relevance for the remainder of this work, nevertheless, the results could be of interest for future works. Hence, we have placed the results in Annex 5.

Conclusion on key frame similarity analysis methods for PSD

Our evaluation of the four key frame pair similarities methods, i.e. the HSV, HY, ScaFT and Sift based method, for parallel shot detection shows that the simplified color based methods, i.e. HSV and HY, robustness-wise outperform advanced feature point based methods mainly due to the abstract similarity between semantically connected shots, e.g. zooms, and the content's frequent non-rigid nature, e.g. faces in the foreground. Nevertheless, the feature point based methods, i.e. ScaFT and SIFT, perform very well in the most often rigid background areas. To enhance the feature point based methods our next step would be to create a background mosaic per shot and to enhance the SIFT method with spatial constellation information, texture dependent thresholds and colour. We believe that with the SIFT method the robustness of the best performing HSV method, could be enhanced or even outperformed. But we decided to exclude the ScaFT- and SIFT-based methods due to their performance from the subsequent steps in this work. Nevertheless, we published the feature point based method for parallel shot detection PSD in one of our patents [136].

Semantic re-annotation of parallel shot detection links and PSD results

The strict, but still fuzzy, rule for the first manual annotation of the ground truth, i.e. that key frame pairs derived from two shots have to exhibit full visual similarity across the image, lead to the situation that semantically linked shots with non-visualized changes are not always part of the ground truth annotation set. The causes for the latter are (a) virtual zooms, (b) virtual object motions, (c) virtual panning and (d) combination of the

previous three. Hence, the benchmark of e.g. the HSV method contains such instances as misses (see Table 32), as well as false detections (see Table 33). Therefore, we re-annotate the ground truth, called here 2nd ground truth, including these semantically linked shots. Hence, the re-annotation rule for the 2nd ground truth includes (a) virtual zooms, (b) virtual object motions, (c) virtual panning and (d) combination of the previous three. The new ground truth statistics for the data set are summarized in Table 36 (an update of Table 26). We restrict the final benchmark on the new ground truth to the HSV and HY method. Recall and precision based on the semantic ground truth, surprisingly increase considerably for series and slightly for movies, as sketched in Figure 96, which shows that the HSV method is also robust enough to cope with slight variations within the content, e.g. zooming and panning. Finally, we also calculate recall and precision for the more relevant *Link Through* LT case benchmarked against the second ground truth (*2nd GT*), summarized in Figure 97.

Table 36. Ground truth numbers of SRSs and CCs of series /movies AV corpus.

Content		# of SRSs	# of shots in SRSs	# of GT SRS links	# of CCs	# of shots in CCs	# of GT CC links	Total # of shots in PS	total # of shots	PS Coverage [%]
Series	'nl1'	9	125	67	12	71	26	196	227	86.3
	'nl2'	11	140	95	6	57	31	197	212	92.9
	'ge1'	12	164	124	0	0	0	164	175	93.7
	'ge2'	21	357	273	2	6	2	363	495	73.3
	'gb'	31	341	226	9	58	28	399	482	82.8
	Total	84	1127	785	29	192	87	1319	1591	82.9
Movies	'ge1'	36	502	351	35	209	103	711	890	79.9
	'ge2'	17	180	137	9	91	60	271	314	86.3
	'nl'	26	353	267	84	604	311	957	1352	70.8
	'us_dig'	46	950	738	13	110	46	1060	1208	87.8
	'us_ana'	31	443	322	43	579	289	1022	1176	86.9
	Total	156	2428	1815	184	1593	809	4021	4940	81.4

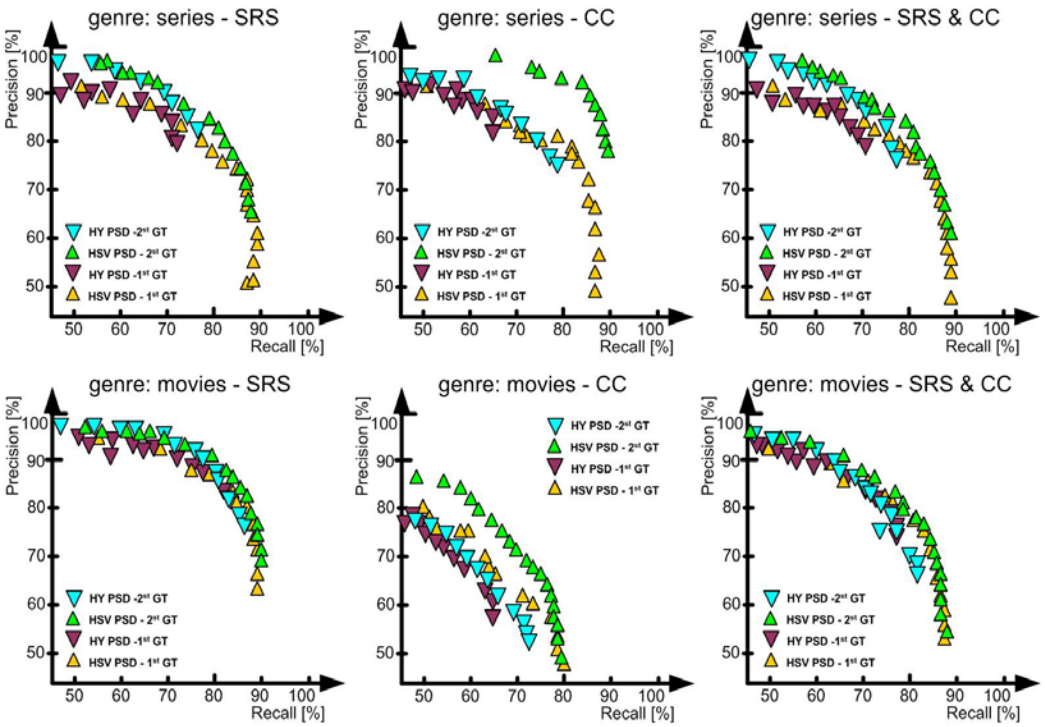


Figure 96. Shot reverse shot and cross-cutting benchmark based on shot links with semantic parallel shot link ground truth (2nd GT).

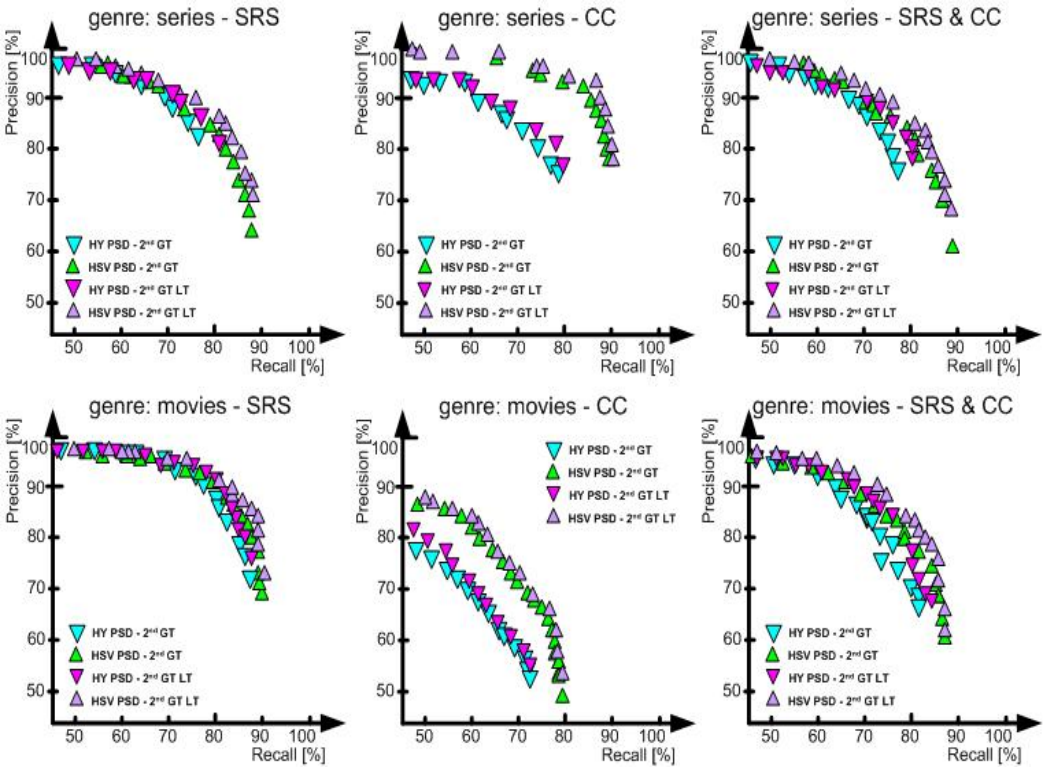


Figure 97. Shot reverse shot and cross-cutting link trough benchmark based on shot links with semantic parallel shot link ground truth (2nd GT).

HSV method evaluation

Because of the comparable results of the HSV- and HY-based methods we make now a selection and choose the processing-wise more efficient one of the two methods, namely the HSV based method, for the remainder of this work.

For the detailed analysis of the HSV key frame similarity analysis for parallel shot detection we choose a parameter setting with a reasonable high link trough precision, derived from the analysis summarized in Figure 97, i.e. W_{sh}=*7* and TH_{PS_HSV}=*75%*. The results with the latter setting are summarized in Table 37. For a detailed analysis of the HSV method, which as described in 4.5.2 is composed of a set of areas, i.e. *foreground* FG, *background* BG and *global area* GA analysis. The influence of the three areas for correct and false links is summarized in Table 38, which sub-divides correct and false decisions into cases were FG, BG or GA was the winning, and hence decisive, class. Correct cross-cutting links are based to about one quarter on the background, mainly due to film grammar based shot distance rules, i.e. many long and medium distance shots are part of those cross-cuttings. Nevertheless, the high contribution of the *global area* reflects the shortcoming of the spatial wise fixed and rough foreground / background segmentation. Objects of interest, i.e. *region of interest* ROI, are not often centred in the middle of the frame, e.g. as specified by film grammar rules during dialogues the protagonists are placed in such a way that spatial rules are satisfied, i.e. faces are presented either in the left or right side of the image.

Table 37. Link trough results of HSV parallel shot detector (W_{sh}=*7*, TH_{PS_HSV}=*75%)*.

Genre		Re [%]	Pr [%]	Correct links	Missed links	False links
Series	SRS	74.0	90.1	574	202	63
	CC	86.2	90.4	75	12	8
	Total	75.2	89.6	649	214	75
Movies	SRS	79.0	91.1	1433	380	140
	CC	62.4	81.0	504	304	118
	Total	73.9	88.2	1937	684	259

Table 38. Detailed HSV area analysis (W_{sh}=*7*, TH_{PS_HSV}=*75%)*.

	Series			Movies		
	SRS Correct	CC Correct	False	SRS Correct	CC Correct	False
FG	12 %	5 %	17 %	17 %	18 %	18 %
BG	15 %	25 %	23 %	18 %	23 %	29 %
GA	73 %	70 %	60 %	65 %	59 %	53 %

Parallel shot detection system

The schematically integration of the service unit HSV key frame pair similarity based parallel shot detector, i.e. SU PSD, into the framework is sketched in Figure 98. SU PSD requires as input the data of SU Shot Boundary Detector, including cut and gradual transition detection, SU Commercial Block Detection, SU Subtitle Detection and a YUV-to-HSV converter. The SU Parallel Shot Detector hosts various units such key frame pair similarity analysis, shot linker and parallel shot decision. Furthermore, two optional extension service units are sketched in the figure, i.e. SU Logo Detection and SU Face Detection, which are scheduled for future extensions.

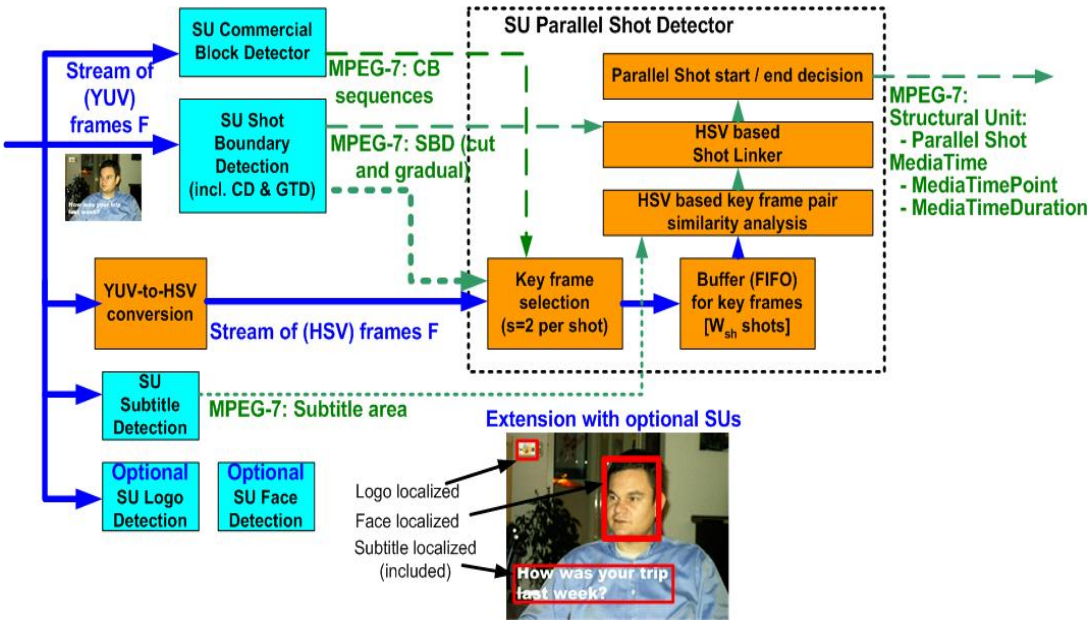


Figure 98. System integration of parallel shot detector[1].

Parallel Shot (general case)

```
<VideoSegment id="RootVS">
 <StructuralUnit href="urn:example:parallel-shots">
  <Name>Parallel Shot</Name>
 </StructuralUnit>
 <MediaTime>
  <MediaTimePoint>T00:00:00</MediaTimePoint>
  <MediaDuration>PT1M30S</MediaDuration>
 </MediaTime>
</VideoSegment>
```

Parallel Shot: Cross-cutting

```
<VideoSegment id="RootVS">
 <StructuralUnit href="urn:example:parallel-shots">
  <Name>Cross-cutting</Name>
  <Definition>Interleaved narrative events</Definition>
 </StructuralUnit>
 <MediaTime>
  <MediaTimePoint>T00:00:00</MediaTimePoint>
  <MediaDuration>PT1M30S</MediaDuration>
 </MediaTime>
</VideoSegment>
```

Parallel Shot: Shot Reverse Shot

```
<VideoSegment id="RootVS">
 <StructuralUnit href="urn:example:parallel-shots">
  <Name>Shot Reverse Shot</Name>
  <Definition>Dialogue sequence</Definition>
 </StructuralUnit>
 <MediaTime>
  <MediaTimePoint>T00:00:00</MediaTimePoint>
  <MediaDuration>PT1M30S</MediaDuration>
 </MediaTime>
</VideoSegment>
```

Figure 99. MPEG-7 description of parallel shots (cross-cutting and shot reverse shot).

As XML-based MPEG-7 like output we suggest to use the *StucturalUnit* element of the *Video Segment DS* and to add three new values, i.e. *Parallel Shot*, *Cross-cutting* and *Shot Reverse Shot*, because nothing applicable is available to specify these elements in a MPEG-7 compliant way. The *StucturalUnit* element seems to be suited, because it is applied in the MPEG-7 specification ([151]: part 5, page 131) as well for descriptions such as *Shot, Scene* and *Story*. The proposed MPEG-7 conform output for the three cases is summarized in Figure 99, i.e. *Parallel Shot:* general case*, Parallel Shot: Cross-cutting* and *Parallel Shot: Shot Reverse Shot*.

Finally, in Figure 100 we visualize the result of the service unit parallel shot detection SU PSD for one selected content item, here movie-ge2, including parallel shot clusters. The black lines in Figure 100 indicate the beginning of a parallel shot sequence and the yellow area indicates the duration of each individual parallel shot sequence. The high density of parallel shot sequences is representative for movies.

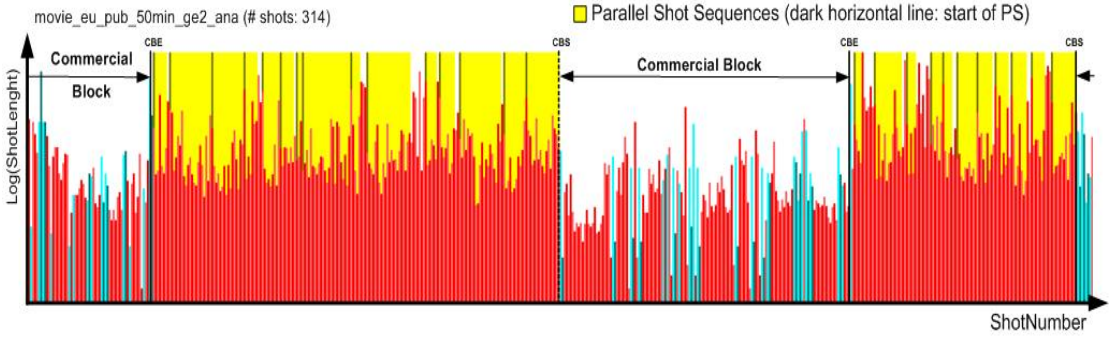


Figure 100. Output of service unit parallel shot detector for content movie_ge2.

4.5.3 Parallel shot categorization

We also exploited the evidence of shot links inside parallel shot sequences to distinguish between

- Cross-Cuttings, i.e. two or more interleaved narrative events, depicting e.g. events unfolding simultaneously at arbitrary locations, and
- Shot / Reverse Shots, i.e. dialogue sequences with two or more individuals shown in alternating fashion.

As presented in

Table 27, the statistics of cross-cuttings and shot reverse shots, i.e. link distance and frequency of appearance, proved to be strong features for content genre classification. Series, for example, contain mainly dialogues and, hence, statistically 70% of all shots of series are Shot/Reverse Shots, as can be seen in Figure 101. On contrary, movies consist not only of dialogues, but also of many parallel events as can be seen in Figure 102. We published the concepts of genre classification in our patent application 'Parallel Shot Detection' [136]. The limitation to only two classes justified developing only a Shot/Reverse Shot, i.e. dialogue, detector and to index the remaining parallel shot clusters as Cross-Cuttings. Hence, we further elaborated the genre classification using face features for Shot/Reverse Shot detection, which we published in [137].

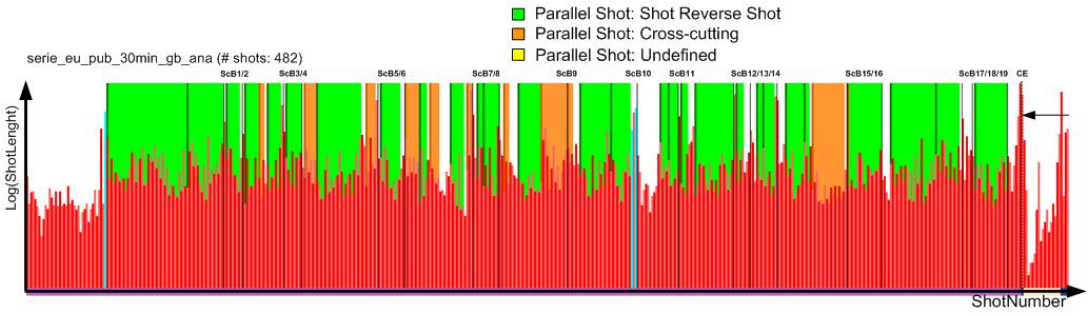


Figure 101. Results after parallel shot classification for content series_gb.

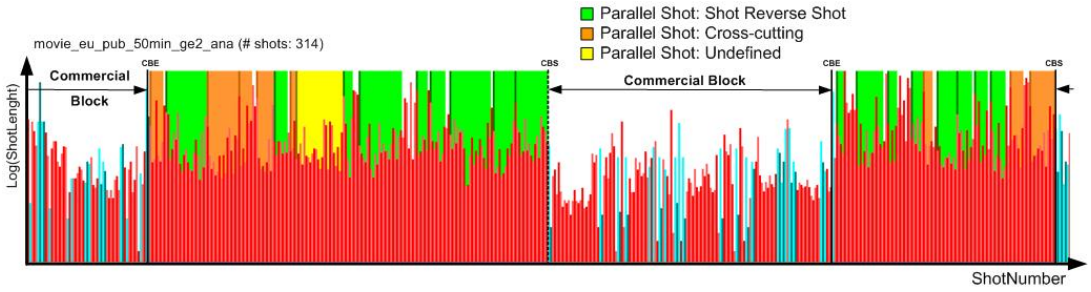


Figure 102. Results after parallel shot classification for content movie_ge2.

The categorization knowledge is very useful information for scene boundary detection, which can be exploited statistically, because the scene boundary attributes around Shot/Reveres Shots are different to those of Cross-Cuttings. In this work we do not exploit this knowledge any further and leave it for future research.

4.5.4 Conclusions concerning service unit parallel shot detection

In this section 4.5 we introduced our own service unit 'Parallel Shot Detection' (SU PSD) exploiting film grammar rules applied in production environments. In particular we exploit the nature of interleaved narrative events and dialogues. We benchmarked four key frame pair similarity analysis methods, i.e. HSV, HY, ScaFT and SIFT, against each other, using shot dependent interval instead of the in prior art time-intervals, which are often applied. We decided, on basis of the benchmark results, to select the best performing HSV based method for parallel shot detection, which reaches in the link trough modus recall / precision of about 83% / 83% or 70% / 90%. Given the fact that about 80% of all shots are members of a parallel shot cluster, i.e. the case for ground truth, this detector reduces the potential scene boundary instances drastically and, hence, is a useful pre-processing step for scene boundary detection. Furthermore, with the parallel shot class knowledge, i.e. the knowledge about the appearance frequency of shot reverse shot or cross-cutting blocks and their internal link structure, a genre classifier has sufficient data to distinguish between soaps, movies and magazines, as we claim in our patent [136].

Nevertheless, our proposed HSV method seems to reach its limits in terms of robustness. We believe that further robustness improvements are achievable, e.g. through the combination of landmark point related techniques, e.g. SIFT, and foreground / background segmentation. These techniques could be applied to create background only mosaics of shots or sub-shots and those mosaics with their landmark points could be used to evaluate the correlation between shots. In addition, mid-level features such as similar face detection could be applied to enhance the robustness even further, e.g. during shot reverse shot sequences.

4.6 Audiovisual segmentation of filtered content

In the final section of this work we describe our research work on retrieving semantic meaningful audiovisual scene boundaries. This section is organized as follows: in section 4.6.1 we define once more scenes and summarize some scene boundary ground truth statistics. In 4.6.2 we present our HSV based dissimilarity scene boundary analysis and the results here of. Subsequently in 4.6.3 and 4.6.4 two post-processing steps using orthogonal features, i.e. based on audio discontinuities and shot length, are presented and the conclusion on scene boundary detection are given in 4.6.5.

4.6.1 Re-definition of scenes

Humans established during their evolution in a tedious process language grammar rules to communicate transparently between each other. Even so, talking about abstract objects such as feelings and interpretations misunderstandings due to improper objective rules are common and frequent. The latter is mainly based on the personal interpretation based on individual past experiences, i.e. individual data sets of situation experiences acquired over during the individual's life time. For the visual domain mankind is even one step further back if compared with languages. In the visual domain artists have started to elaborate some visual grammar, i.e. film grammar, but still a long way has to be passed before well-established objective rules will satisfy transparent interpretation of visual information. On an abstract level it is possible to establish an analogy between film and language entities, i.e. video shots can be set equal to individual spoken words, parallel shots to subordinate clause, visual scenes to full sentences, video acts to book chapters, full video items to books and, finally, series of content items to book series. In course of this work we elaborated so far definitions and technical solutions to identify individual shots, individual parallel shot sequences and to eliminate non-content related inserts. The next step is to use available knowledge about scene rules to reach the final aim of this work, i.e. to identify semantic audiovisual scene boundaries. In the film grammar section we mentioned that narrative content is usually split into three acts, and subsequently each act is split further into semantic scenes. Scenes contain usually several parallel shots, i.e. interleaved narrative elements. By definition these interleaved narrative elements form a semantic entity. Hence, they can never contain a semantic scene boundary. In general, a semantic scene conveys a special message or meaning to the audience to understand the flow of the story (fuzzy definition). Hence, for an objective evaluation of scene detection algorithms we require well-defined *objective* rules for scenes incorporating as much as possible film grammar rules.

Table 39. Ground truth scene boundaries.

Series	'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total	Average Scene Duration [min]
GT ScB	7	16	11	17	19	70	2.1
Movies	'nl'	'ge1'	'ge2'	'us_ana'	'us_dig'	Total	
GT ScB	35	25	30	53	26	169	2.6

Based on the analysis of film grammar and studies of content of various genres we come to the conclusion that the following set of definitions and rules fits best to define semantic scenes:

- Scenes consist of one or more shots conveying one single and consistent underlying semantic;
- Scenes may incorporate one or more interleaved narrative events, i.e. cross-cuttings or dialogues (shot reverse shots). But by definition these interleaved narrative sequences form a consistent semantic entity and, hence, scene boundaries may not appear within them. Nevertheless, interleaved narrative sequences may be bordered by scene boundaries;

Scenes contain usually one or more narrative elements, which often are surrounded by introduction and conclusion elements, i.e. one or more introduction and conclusion shots, as shown in Figure 77 and Figure 80. We have given above listed scene boundary rules to five persons as annotation basis and asked them to select scene boundaries using our AV (series / movies) data corpus. The latter consist of five movies and five series. Applying a majority vote approach we selected the test group's resulting scene boundaries and present them in Table 39 and in more detail in Annex 6. The analysis unveils that semantic scenes have in average durations of 2.1 to 2.6 minutes.

4.6.2 HSV based ScB detection

As we stated in the previous section (4.5.1) mis-en-scene rules and cinematographic rules unveil that shots of interleaved narrative elements clustered together in parallel shots sequences contain visual correspondence, which facilitate the viewer to discriminate intuitively between individual semantic sequences. Hence, parallel shot sequences constitute elements of audiovisual scenes and, therefore, scene boundaries cannot be met within such parallel shot sequences. The detection of parallel shot sequences, which we described in the previous section, can be considered as a pre-filtering for subsequent scene boundary detection. In this section we are interested in the detection of scene boundaries within the remaining content sequences, i.e. after pre-filtering.

From the visual set of features, e.g. color, motion, texture, audio, speech, we choose to apply color first. Several color spaces were to our disposal, i.e. YUV, RGB, HSV, LUV, based either on global or spatial information, e.g. color auto-correlogram or color coherence vectors, and color histograms, i.e. uniform and non-uniform ones. Analysis showed that the strongest discriminative power is reached with uniform distributed HSV color histogram, i.e. 16 discrete bins for hue, 4 for saturation and 4 for value, as described in 4.5.2 and Figure 83.

Here after, we exploit the knowledge that cinematographic rules force directors to secure graphic consistencies, i.e. settings of shots of one scene should exhibit similar color compositions. Hence, if sets of key frames of two shots show strong color dissimilarities it is most likely that they belong to different scenes. We choose the sets of key frames of two shots by regular temporal sub-sampling of the video sequences. Here fore we apply the usual MPEG-2 GOP size of 6, i.e. frame distance of 6, as the temporal sub-sampling rate. We witnessed by tests that the robustness loss compared to lower sub-sampling rates is negligible. Hence, we proceed with the chosen distance.

HSV based scene boundary dissimilarity analysis

In 4.5.2 we introduced the *Histogram Intersection Distance* HID, published by Jeong in [126], on four key frame elements as shown in Figure 103, i.e. foreground FG, background BG and downscaling in two directions, called global zoom GZ/ZG. The latter is used to increase the robustness of the method by being able to detect and link non-captured zoom-in and zoom-out sequences, as shown in Figure 103 (right). Directors apply this global zoom often during the setting describing (re-)establishing and conclusion shots using simply more distant shots, i.e. medium or long shots, with a

similar color setting. For our method we downscale the image horizontally and vertically by a fixed factor of 2/3.

Hence, four histogram intersections were defined of key frame pairs F_N/F_M, i.e.

(a) $$HID_{FG}\left(F_N\left(FG\right),F_M\left(FG\right)\right)=\frac{\sum_{k=o}^{bin_{\max}-1}\min\left(Hist_{FG}\left(k,F_N\right),Hist_{FG}\left(k,F_M\right)\right)}{\min\left(\left|Hist_{FG}\left(F_N\right)\right|,\left|Hist_{FG}\left(F_M\right)\right|\right)} \quad (4\text{-}72),$$

(b) $$HID_{BG}\left(F_N\left(BG\right),F_M\left(BG\right)\right)=\frac{\sum_{k=o}^{bin_{\max}-1}\min\left(Hist_{BG}\left(k,F_N\right),Hist_{BG}\left(k,F_M\right)\right)}{\min\left(\left|Hist_{BG}\left(F_N\right)\right|,\left|Hist_{BG}\left(F_M\right)\right|\right)} \quad (4\text{-}73),$$

(c) $$HID_{GZ}\left(F_N\left(GA\right),F_M\left(GZ\right)\right)=\frac{\sum_{k=o}^{bin_{\max}-1}\min\left(Hist_{GA}\left(k,F_N\right),Hist_{GZ}\left(k,F_M\right)\right)}{\min\left(\left|Hist_{GA}\left(F_N\right)\right|,\left|Hist_{GZ}\left(F_M\right)\right|\right)} \quad (4\text{-}74),$$

(d) $$HID_{ZG}\left(F_N\left(GZ\right),F_M\left(GA\right)\right)=\frac{\sum_{k=o}^{bin_{\max}-1}\min\left(Hist_{GZ}\left(k,F_N\right),Hist_{GA}\left(k,F_M\right)\right)}{\min\left(\left|Hist_{GZ}\left(F_N\right)\right|,\left|Hist_{GA}\left(F_M\right)\right|\right)} \quad (4\text{-}75),$$

Figure 103. Four histogram intersection distances applied for F_N/F_M analysis[1].

as also visualized in Figure 103, which are intended to identify the optimal frame similarity. Hence, the highest one of the four histogram intersection distances is selected to represent the F_N/F_M similarity with

$$HID_{\max}(F_N, F_M) = \max\begin{pmatrix} HID_{FG}(F_N(FG), F_M(FG)), HID_{BG}(F_N(BG), F_M(BG)), \\ (HID_{GZ}(F_N(GA), F_M(GZ)), HID_{ZG}(F_N(GZ), F_M(GA))) \end{pmatrix} \quad (4\text{-}76).$$

For the analysis of scene boundaries we consider the maximal similarity between shot pairs sh_N/sh_M. It is derived by calculating $HID_{max}(F_N,F_M)$ for the entire key frame sets of two shots sh_N/sh_M and selecting the one with the maximal histogram intersection distance with

$$Sim_{\max}(sh_N, sh_M) = \max_{F_i \in sh_N, F_j \in sh_M} \{HID_{\max}(F_i, F_j)\} \quad (4\text{-}77).$$

The minimal dissimilarity between two shots $Dissim_{min}(sh_N,sh_M)$, hence, is simply derived by subtraction the resulting similarity $Sim_{max}(sh_N,sh_M)$ from a value of one with

$$Dissim_{\min}(sh_N, sh_M) = 1 - Sim_{\max}(sh_N, sh_M) \quad with \quad Dissim_{\min} \in [0...1] \quad (4\text{-}78).$$

$Dissim_{min}(sh_N, sh_M)$ represents therefore the absolute minimum key frame dissimilarity of all frames of two shots sh_N /sh_M, which is exactly the parameter required to identify color based cinematographic discontinuities in a video stream.

To build an efficient decision scheme we introduce now the minimal dissimilarity $P(SB_l)$ within a 'window' of W_{sh} shots where the minimum of all dissimilarities $Dissim_{min}(sh_N, sh_M)$ of all shot pair links crossing a certain shot boundary SB_l is chosen, as shown in Figure 104.

The min dissimilarity $P(SB_l)$ is calculated with

$$P(SB_l) = \min_{sh_N, sh_M \in W_{sh}, sh_M < sh_l, sh_N \geq sh_l} \{Dissim_{\min}(sh_N, sh_M)\} \quad with \quad P(SB_l) \in [0...1] \quad (4\text{-}79).$$

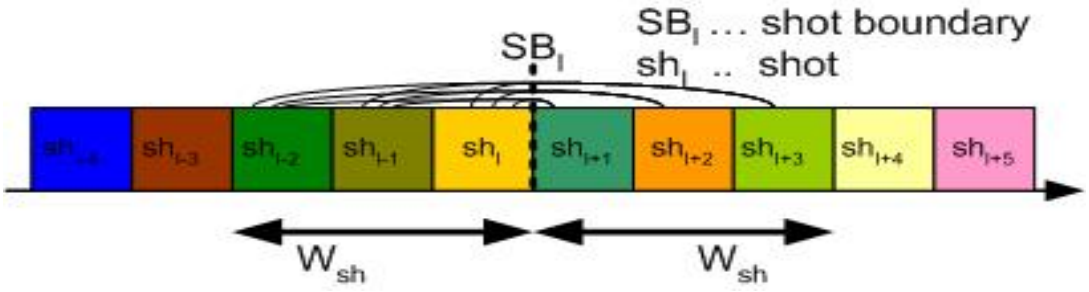


Figure 104. Shot pair dissimilarity analysis within window W_{sh}.

All shots considered for the shot boundary SB_l, i.e. candidates to a scene boundary, have to be within the distance of $W_{sh}\in[3..30]$ shots. Subsequently, we introduce the mean dissimilarity measure μ_{SBl} normalizing within the chosen window W_{sh} across all min dissimilarities $P(SB_l)$ with

$$\mu_{SB_i} = \frac{1}{W_{sh}} \sum_{l=i-\frac{w_{sh}}{2}}^{i+\frac{W_{sh}}{2}-1} P(SB_l) \qquad (4\text{-}80).$$

Finally, we introduce the maxmin dissimilarity position *MaxMinPos*. We need this measure to identify the maximum dissimilarity position for each shot boundary SB_l with

$$MaxMinPos = \arg\max_{l-3\le q\le l+3}\{P(SB_q)\} \qquad (4\text{-}81).,$$

We use a window size of three to circumvent multiple successive detections around local maxima surrounded by slightly lower probability peaks. Finally, a shot boundary is indexed as scene boundary if the local min dissimilarity $P(SB_l)$ not only exceeds the product of mean dissimilarity μ_{SB} and threshold $Th\in[1.2 \dots 3]$, but also represents the local maximum and exceeds a minimum threshold of $P_{min}=0.04$. Thus the decision rule on a scene boundary is formulated as:

$$ScB(SB_i) = \begin{cases} 1 & if \quad P(SB_i) > Th * \mu_{SB_i} \quad \cap \quad i = MaxPos \quad \cap \quad P(SB_i) > P_{\min} \\ 0 & else \end{cases} \qquad (4\text{-}82).$$

Results of HSV based scene boundary dissimilarity analysis

We describe, next, the assessment scheme for scene boundary detection, which is based on HSV dissimilarity. For the analysis of the scene boundary detector the parameters $W_{sh}\in[3..30]$, $Th\in[1.2 \dots 3]$ and $jitter\in[0\dots3]$ are applied. The latter is a tolerance interval. We introduce it to cope with establishing and conclusion shots, which have durations of one to even three shots, as can be seen in Figure 77. These shots visualize the entire setting in medium or long shots and, therefore, capture more of the setting often from a different position, e.g. outside shot. Color based shot dissimilarities appear, therefore, within a certain distance from the exact scene boundary instance. A robust establishing and conclusion shot identification algorithm, which is an interesting research topic in itself, is not available at the moment of this work. Hence, we solve this problem by using a jitter window. With the latter the detection of a boundary is considered as correct, if the instance of the *ground truth* GT falls inside the jitter window *j*. Hence, with the set of *Automatically Detected Scene Boundaries* ADSB, the number of real scene boundaries GTScB, the total number of shot boundaries TotalSB, the

number of correct, missed and false detections are calculated. The set of ADSB is the subset of SB, which is scene boundaries with

$$ADSB = \sum_{i=1}^{TotalSB} (ScB(SB_i)) \quad (4\text{-}83).$$

The numbers of correct and false detections are calculated across all ADSB (*ds* represents the variable) with

$$Correct = \sum_{ds=1}^{ADSB} CorrectScB(ds) \quad and \quad False = \sum_{ds=1}^{ADSB} FalseScB(ds) \quad (4\text{-}84).$$

Then, additionally two problems are taken into account, i.e. (a) that in the case of a correct detection two overlapping windows may appear impropriate increasing recall, and (b) that in the case of a false detection overlapping windows can influence precision. Hence, detection instances appearing in overlapping windows are counted only once. This means, for situations with two scene boundaries, i.e. o^{th} *ScB(o)* and p^{th} *ScB(p)*, which are in close neighbourhood to the s^{th} scene boundary ground truth *GTScB(s)*, only the first instance is taken into consideration for the correct detection value, with

$$CorrectScB(ds) = \begin{cases} 1 \quad if & GTScB(s) \in [ScB(o) - j, ScB(o) + j] \wedge \neg \exists\, ScB(p): \\ & GTScB(s) \in [ScB(p) - j, ScB(p) + j] \wedge\ p < o \\ 0 \quad otherwise & \end{cases} \quad (4\text{-}85).$$

The same approach is taken with false detection, i.e. if two false detections appear within window *j* only the first false detection is added to the total false detection value with

$$FalseScB(ds) = \begin{cases} 1 \quad if & GTScB(s) \notin [ScB(o) - j, ScB(o) + j] \wedge \neg \exists \\ & ScB(p) \in [ScB(o) - j, ScB(o) + j]: ScB(p) \notin FalseScB(ds) \wedge\ p < o \\ 0 \quad otherwise & \vdots \end{cases} \quad (4\text{-}86).$$

Missed detections are calculated by subtracting correct detections from the total number of ground truth scene boundaries. Here we consider the GT presented in Table 39. Subsequently recall and precision are calculated as stated in equation (3-16) and (3-17). The results of the benchmark are visualized in Figure 105 and as shown the filtering of scene boundaries occurring within parallel shots, excluding those within a distance of *j* from a parallel shot boundary, increases as expected precision by several percents. Concrete results for two chosen settings are presented in Table 40. As we know from previous sections 80% of all shots of series are member of parallel shots. Hence, the detection performs better on series compared to movies, because more content is pre-filtered into parallel shot sequences, and, therefore, the detection error (precision) decreases if more content is pre-filtered.

Table 40. Benchmark results of scene boundary detector after parallel shot detection.

	W_{sh}	Th	Jitter	Correct	False	Missed	# potential shot boundaries (outside PS)	Re [%]	Pr [%]
S&M	6	1.4	3	**204**	**206**	**35**	**1191**	**85.2**	**49.4**
Series				58	38	12	272	82.9	59.6
Movies				146	168	23	919	86.3	46.3
S&M	10	2.4	1	146	99	93	1191	60.6	59.1
Series				50	15	20	272	70.6	76.2
Movies				96	84	73	919	56.5	53.1

Furthermore, scene transitions in series are very distinct, i.e. abrupt, applying very few establishing and conclusion shots, which is not the case in movies. Hence, recall-wise the method's potential is promising, but especially for movies the HSV scene boundary detection combined with parallel shot detection requires further features to reach satisfying precision results.

Conclusion for HSV based scene boundary dissimilarity analysis

Our scene boundary detector, based on HSV based dissimilarity analysis of shots, proves to be reasonably robust and processing efficient component to identify scene boundaries. In our experiments it detects with a lose setting, i.e. $W_{sh}=6$, $Th=1.4$ and $j=3$, 204 of the 239 scene boundaries, i.e. *Re=85.2%*. For further clarification, our solution does not take appropriate measurements to deal with scene boundaries, which occur close to content item boundaries. The problem occurs because the applied buffer for our calculation is not filled sufficiently at content item boundaries. Another limitation is, that scene boundaries are discarded, if separated by only one shot, i.e. scene boundaries occurring close to each other. These are the constraints for any detector requiring computation in a temporal buffer. Hence, out of the 35 missed scene boundaries in total 5 are not detectable, because they are placed close to the boundaries of the content, and 13 are not detectable due to their close distance to other scene boundaries, as visualized in Annex 6.

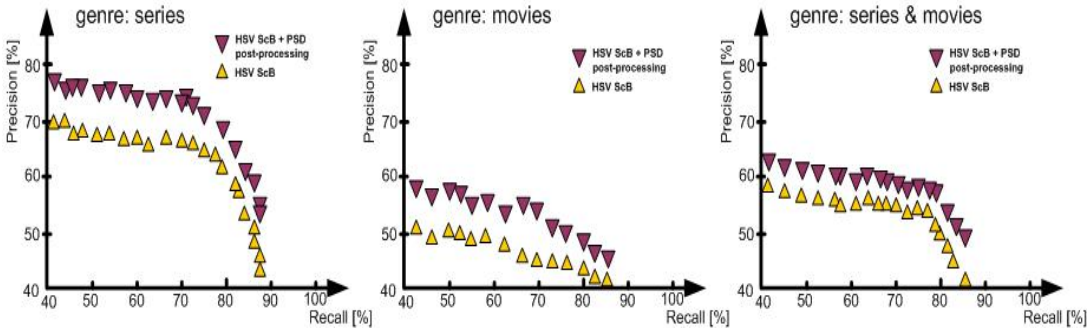


Figure 105. HSV based scene boundary detection with parallel shot post-processing.

Such scene boundaries are questionable, but, because it was a common decision of several manual annotators, we decided to keep these scene boundaries in the ground truth set. In Figure 106 an example is given including ground truth scene boundaries *ScB* (bold black vertical lines), scene boundary min dissimilarity $P(SB_i)$ (blue graph at bottom) and detected scene boundaries (green lines at bottom) applying a specific setting.

Scene boundaries are bordered very often by establishing and conclusion shots as we presented in 4.5. An example for this remaining problem of medium or long distant establishing and conclusion shots, which cause slight offset detections, a problem solved for the moment by us by applying a jitter *j*, is shown in Figure 107.

Hence, in this section we presented methods for scene boundary detection based on visual information only. Nevertheless, AV content contains semantically rich audio as well. In the next section we consider, therefore, combined audio-visual methods to enhance our current scene boundary detection approach.

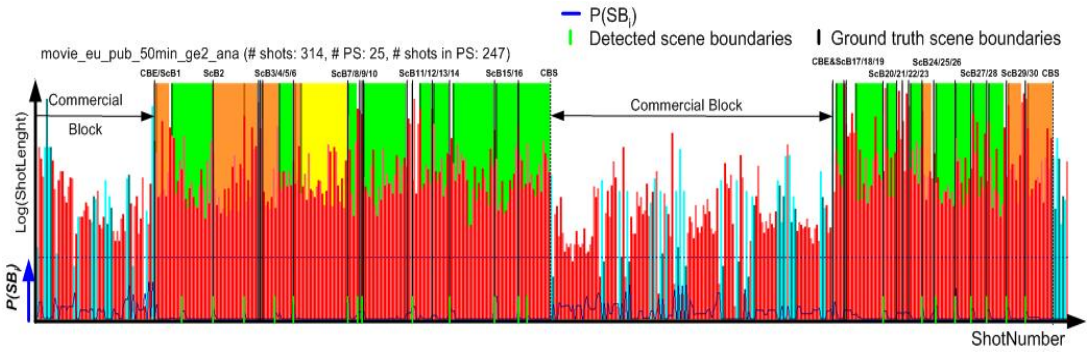


Figure 106. Content *movie-ge2* with ground truth scene boundaries, scene boundary dissimilarity measure and detected scene boundaries (W_{sh}=*10*, *Th=2.4* and *j=1).*

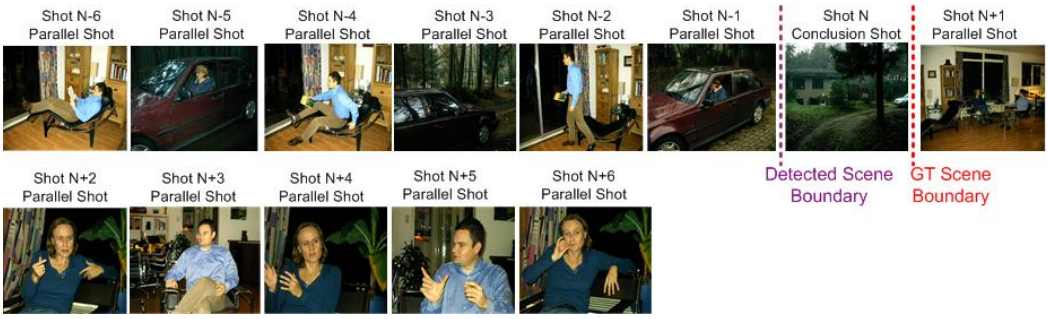


Figure 107. Example for detection within a jitter caused by establishing / conclusion shots[1].

4.6.3 Audio-visual discontinuities as a model of a scene border

Audiovisual scene boundaries have the attribute that the audio signal and the visual signal experience independently intermissions at these distinct scene boundary instances. In the audio domain these can be silences, audio class transitions or energy discontinuities, and in video domain these are shot boundary instances. The assumption is that the intermissions of those independent information sources correlate time-wise according to the scene boundary model. But, as stated by film grammar and production rules in section 4.5, audio and video transitions exhibit, often by intention, certain misalignments, i.e. that an audio changes slightly before or after a video change. For example, often directors intentionally, i.e. semantically, glue independent narrative story elements through audio together, as shown in Figure 108 (left), or during dubbing, i.e. an usual (post-) production technique for language adaptations, these misalignments are introduced. Moreover, in some cases slight offsets between audio and video are added deliberately to avoid unpleasant harsh audiovisual transitions called mixing, as shown in Figure 108 (right).

Audio silence and video cut correlation analysis

Gaussian Distribution Model for AV jitter

In a first attempt we try with one colleague[15] to research the time-wise correlation between audio silence instances and video cut transitions for scene boundary detection.

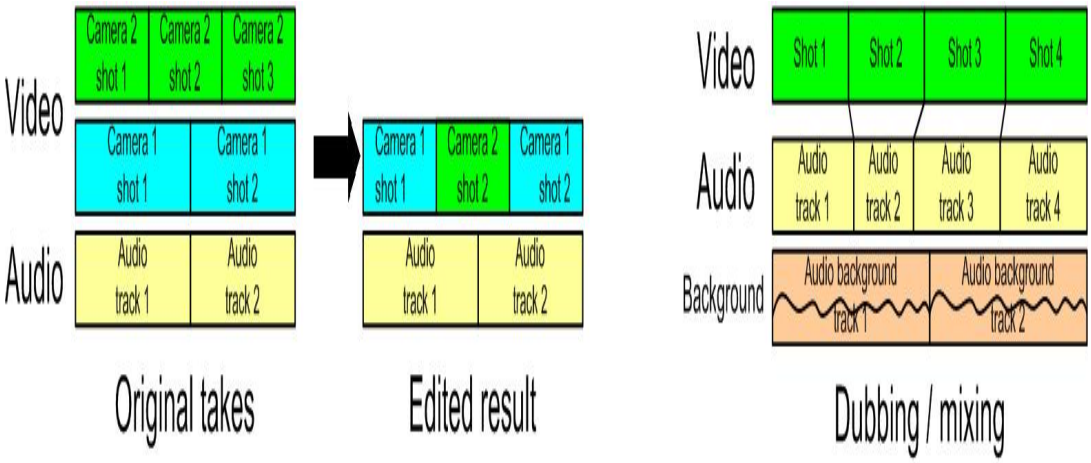


Figure 108. Audiovisual editing and dubbing.

[15] Co-supervised PhD student Nicolas Louis

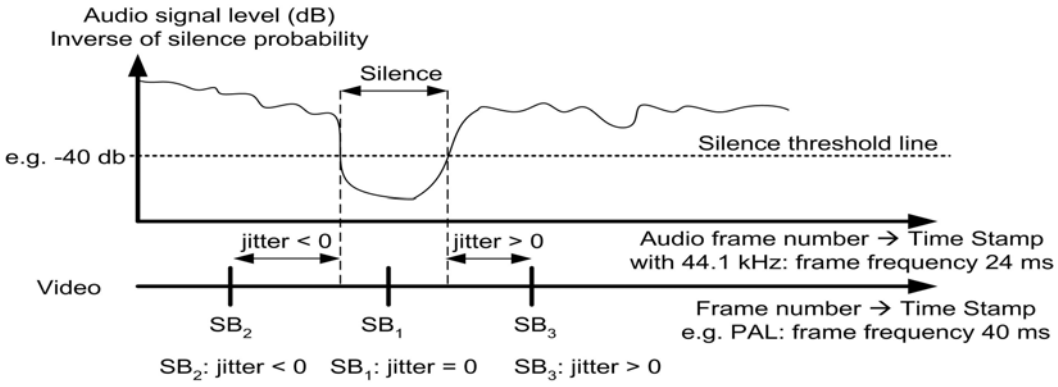


Figure 109. Scheme for AV jitter measurement.

Being aware of the film and production rules, which led to slightly misaligned audio and video transitions a tolerance value, called “audio-visual jitter” *J*, is introduced to cope with this time-wise fuzzy correlation. Here for a combination of a Gaussian distribution model, the Bayesian theorem and a maximum likelihood ratio is applied and evaluated, which we disseminated in our paper [117] and which is fully published by Louis in [37]. In the actual PhD thesis we present only the concept and the results.

The introduced jitter represents the time-wise distance between the current video shot boundary *SB*, i.e. a video cut represented by the timestamp *SB(i)* corresponding to the *i*-th cut instance with $j=[1, N_{SB}]$ wherein N_{SB} represents the total number of indexed shot boundaries, i.e. video cuts, and the closest audio silence instance, as sketched in Figure 109, using a simple silence detector, see [117] and [37]. Each silence instance *s(s)* is represented by the beginning timestamp *sb(s)* and end timestamp *se(s)*, here e.g. the *s*-th instance with $s=[1,S]$ wherein *S* represents the total number of indexed silences.

The jitter *J(i)*, i.e. the jitter at cut instance *i*, is, hence, represented by

$$J(i) = \begin{cases} 0 \cdot if \cdot sb(s) \le SB(i) \le se(s) \\ otherwise \cdot \min_s \left(\min \left(\left| SB(i) - sb(s) \right| \left| SB(i) - se(s) \right| \right) \right) * sign \left(\arg\min \left(\left| SB(i) - sb(s) \right|, \left| SB(i) - se(s) \right| \right) \right) \end{cases} \quad (4\text{-}87)$$

wherein only the closest silences *s(s)* around *SB(i)* are taken into consideration. A statistical Bayesian decision model is applied to specify conditional and unconditional probability density functions and to calculate the maximum likelihood function with correct and false scene boundary detection within a certain jitter distance using a training set. The jitter is assumed to follow a Gaussian distribution. The statistical analysis is applied to derive heuristically an optimal jitter threshold value Th_J [number of video frames]. Finally, Th_J is used to identify potential scene boundary instances, as presented in detail [117] and [37]. The approach and equation applied are further described in [117] and [37]. The evaluation is done with

$$\frac{L_{ScBs}}{L_{NScB}} = \frac{\frac{1}{\sqrt{2\pi\sigma^{2}_{ScBs}}} e\left(-\frac{(j-\mu_{ScB})^{2}}{2\sigma^{2}_{ScBs}}\right)}{\frac{1}{\sqrt{2\pi\sigma^{2}_{NScB}}} e\left(-\frac{(j-\mu_{NScB})^{2}}{2\sigma^{2}_{NScB}}\right)} * \frac{P(H_{ScB})}{1-P(H_{ScB})} \begin{matrix} > \\ < \end{matrix} Th_{J} \Rightarrow \begin{matrix} H_{ScB} \\ H_{NScB} \end{matrix} \quad (4\text{-}88)$$

with following parameters applied: mean μ and standard deviation σ jitter value of positive (*ScB*) and negative (*NScB*) examples and the probabilities of the hypothesis that a jitter value corresponds to a scene boundary with a silence $P(H_{ScB})$ or does not correspond $P(H_{NScB})$.

Training of stochastic AV jitter model

As described in [117] and [37], a sub-set of the AV corpus, see Table 41, is used to derive a Gaussian-distribution-based parameter for the training and evaluation process, i.e. four series and four movies. The columns in Table 41 represented the number of shot boundaries *SB* before parallel-shot-based clustering *PS*, number of shots clustered into a number of *PSs* (as described in section 4.5.2 using the first ground truth) and number of shot boundaries after parallel-shot-based clustering *PS*, i.e. the shot boundaries left outside parallel shots, which are potential scene boundaries *ScB*. The last column identifies all other potential *ScB*, i.e. potential *ScBs* at boundaries of content item boundaries or adds.

Table 41. AV sub-corpus for AV segmentation analysis (series and movies).

AV sub-corpus		Total # SB	# Shots inside PS (1st GT)	/ # PS	# of potential ScB	# ScB
Series	'nl1'	231	151	15	95	7
	'ge1'	181	128	9	62	11
	'ge2'	490	307	23	206	17
	'gb'	481	360	21	142	19
	Total	1383	946	68	505	54
Movies	'ge2'	442	242	21	221	30
	'nl'	1400	523	33	910	35
	'us_dig'	1190	1041	44	193	26
	'us_ana'	1260	736	31	555	53
	Total	4292	2542	129	1879	144

Successively, each content item is split into two parts, whereof the first part, further called training set, is used for training and the second, further called test set, for testing the performance of the solution. Successively, each shot boundary instance *SB*, i.e. video cut, of the training and test set is manually labelled either as scene boundary with a correlating silence SB_a (*±½ s ~ Th_J=±15* video frames) with the index *a=ScB* for positive (indeed scene boundaries) and index *a=NScB* for negative examples. For each

of the two data sets $X_a=\{x_{a1},..x_{aNa}\}$, wherein N_a represented the size of each of the two datasets,

$$\mu_a = \frac{1}{N_a}\sum_{z=1}^{N_a} x_{az} \quad , \quad \sigma_a{}^2 = \frac{1}{N_a}\sum_{z=1}^{N_a}\left(x_{az} - \mu_a\right)^2 \qquad (4\text{-}89)$$

and the class probabilities $P(H_{ScB})=N_{ScB}/N_{SB}$ and $P(H_{NScB})=1-P(H_{ScB})$ are calculated.

Results with audio-visual jitter model

The results of the evaluation with a statistically derived threshold Th_J,=(0,+2) from 4.6.3 on the trainings set and an empirical threshold $Th_J = \pm 3$, described in [117] and [37], are summarized in Table 42. The evaluation is executed on the test set and parallel shot post-processing is applied to eliminate detections within parallel shots. As can be seen in Table 42, the precision of parallel shot detection increased from 5% to more than 16%. The analysis resulted in a reasonable recall; nevertheless, the precision is still too low for most applications.

Table 42. Results with AV jitter before and after parallel shot detection.

	Th_J	Total # ScB	Corrrect ScB_S	Missed ScB	False ScB	Re [%]	Pr [%]
S&M before PSD	$Th_J = \pm 3$	64	52	12	965	81,2	5,1
	$Th_J = (0,+2)$	64	47	17	853	73,4	5,2
Series after PSD	$Th_J = \pm 3$	16	11	5	25	75.7	31.0
Movies after PSD	$Th_J = \pm 3$	48	41	7	238	87.1	14.5
S&M after PSD	$Th_J = \pm 3$	64	52	12	263	81.2	16.5

Conclusion of AV Jitter

Our manual evaluation reveals, that about 60% of all scene boundaries in the ground truth contain shot boundary and silence instance correlations. The remaining 40% exhibit other audio class transitions in the close neighbourhood to the remaining scene boundaries, as summarized in Table 43.

Table 43. Remaining 40%: audio class transitions in close neighbourhood of ScB.

Audio class transition	Percentage
Speech to noise	14,28 %
Noise to noise	12,24 %
Noise to speech	12,24 %
Music to speech	12,24 %
Speech to music	10,21 %
Music to noise	8,16 %
Music to music	8,16 %
Speech to speech	6,13 %
No transition	6,13 %
Music to noise plus music	4,09 %
Noise to speech plus music	2,04 %
Noise plus speech to noise plus speech	2,04 %
Noise to music	2,04 %

Unfortunately, the analysis with the empirical value $Th_J = \pm 3$ unveils that recall is high, i.e. 84%, but precision is far too low, i.e. <17%. Very often this is the case due to speech sequences outside parallel shots. Hence, we decide to evaluate alternative audio segmentation methods.

Audio scene segmentation for audio scene boundary detection

As specified by film grammar, scenes contain in general one or more interleaved narrative sequences encapsulated by establishing and conclusion shots introducing or concluding a story element. These scene transitions have to be audio-visually identifiable by the viewer to secure the understandability. But as stated before, audio and visual transitions not necessarily have to be time wise aligned. Even more, directors slightly dislocate them from each other to create abstract connections. Moreover, not only scene boundaries have to be recognizable, but also individual events inside a scene. We assume that at these instances audio will exhibit class changes. Therefore, we manually identified together with five colleagues audio scene boundaries *AScB*, i.e. instances at which either (a) the continuous ambient sound that surrounds the scenery audibly changes or (b) audio classes exhibit transitions. The results of the manual annotations are summarized in Table 44 and Annex 6. The manual evaluation unveils that the average ratio, i.e. series and movies, between audiovisual scene boundaries *AV ScB* and audio scene boundaries *AScB* is about 1:1.3, i.e. about 30% more *AScBs* compared to *AV ScBs*. The average duration of an audio scene is about ~1.7 minutes and only 12 of the 239 ground truth scene boundaries miss a correlating audio scene boundary. One could question those 12 scene boundaries, but due to a majority vote we left these scene boundaries in the ground truth.

Table 44. Audio scene boundary *AScB* ground truth.

Series	'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total AScB	Total AV ScB	Average Audio Scene Duration [min]	Average AV Scene Duration [min]
GT Audio ScB	6	19	10	16	34	85	70	1.7	2.1
Movies	'nl'	'ge1'	'ge2'	'us_ana'	'us_dig'				
GT Audio ScB	57	34	41	75	32	239	169	1.8	2.6

In general six audio classes are taken into consideration by us, i.e. music *M*, speech *Sp*, noise *N*, crowd noise *Cr*, silence *Si* and unknown *U*. In Annex 6 we specify for each audio scene boundary *AScB* not only the audio class transition type, e.g. noise to music (N→M), but also the misalignment in shots between audio scene boundary and the audiovisual scene boundary, see annex 6 last column. The misalignment evaluation shows that at (a) 113 out of 239 scene boundary instances an exact alignment with an

audio scene boundary occurs, (b) 68 out of 239 instances of the audio transition occur during the subsequent shot, at (c) 33 out of 239 instances the audio transition appears during the predecessor shot, and (d) 13 out of 239 instances have a higher misalignment than one shot. From a human perceptual point of view audio should either be aligned with video or time-wise follow video, because humans are used to see first and then to receive the related audio information. We expect that the misalignment information alone could contain some semantic information, which could be a potential field for future research.

Audio scene boundary detection with audio classifier

Our goal in this section is to detect audio scene boundaries, which we understand as transitions between audio classes, as mentioned before. We decided for this purpose to apply an audio and music classifier described in [74] for the detection of audio class transitions. The classifier provides independent class probabilities, i.e. class independent probability values between zero and one, for six audio classes, i.e. speech, music, noise, crowd, silence and unknown, as shown in Figure 110 for various classes in various colors for a sequence of movie_ge1. The classifier is trained on a set of low level signal properties, mel-frequency cepstral coefficients *MFCC* and psychoacoustic features, of pure audio samples, as further described by McKinney in [74].

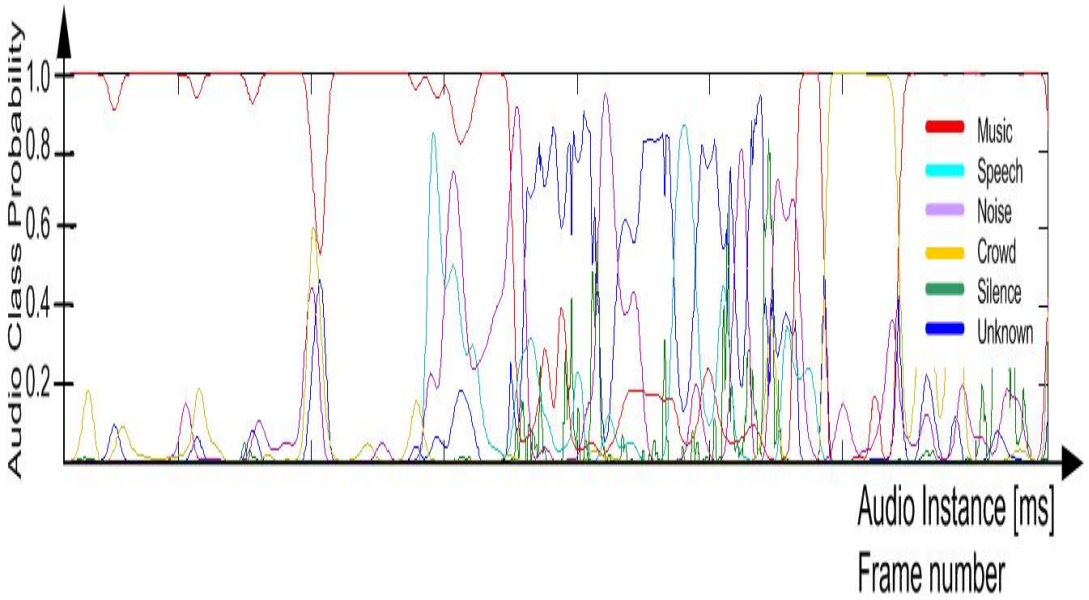


Figure 110. Audio class probability results obtained with audio classifier.

For the detection of audio class changes we apply first sub-sampling of the audio classifier output, i.e. reducing the frequency of the six class vectors to the PAL video frame rate. Subsequently several methods are tried to identify audio scene boundaries. One applied method is to cluster the feature vectors within a sliding window using k-mean clustering. Feature vectors, here after, are substituted by the index of the cluster they belong to. Subsequently, the dominant indices are calculated within a certain predecessor and successor window *W* at a video frame instance. These dominant indices are applied to identify the dissimilarity measure P_{Dis} between the predecessor and successor window, an equivalent of computing the relative entropy, i.e. Kullback-Leibler distance. Finally local maxima are identified by means of an adaptive threshold. Unfortunately the output of the audio classifier do not meet the robustness requirements for this purpose mainly due to the fact that the classifier was developed for audio only signals, i.e. radio stations [74]. At some instance the generated class transitions correlate with the manually annotated ground truth, as pictured in Figure 111 (left), but in many cases audio class probabilities mask audio scene boundaries present in ground truth, as shown in Figure 111 (left).

Subsequently, we try another approach using the classifier output for detecting audio scene boundaries applying a Mamdani-type Fuzzy Interference System (FIS), as presented in [138], and an Adaptive Neuro Fuzzy System (ANFIS), described in [139]. But, because in both cases the results confirm that the classifier require re-training we discontinue the approach using audio class transition detection for audio scene boundary detection. Nevertheless, we believe that audio scene boundary detection by means of audio class transition detection contain opportunities.

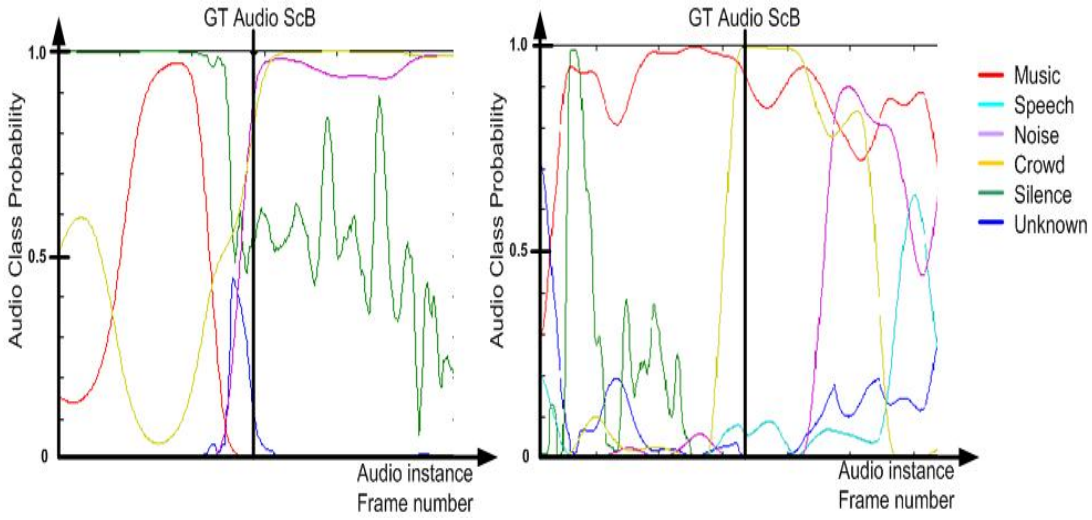


Figure 111. Positive and negative class transition examples for ground truth AScB.

Audio scene boundary detection

Another attempt to include audio for scene boundary detection is based on the experience that the audio power level exhibits a level change around scene boundaries. Hence, we select the low-level feature available within the audio classifier, i.e. audio power, as described in [74], and calculate the video shot mean audio power level μ_{AP}, i.e. the mean power level calculated across an entire shot. Here after, we apply a first order Gaussian derivative filter with

$$Gaussian'(x) = \frac{-x}{\sqrt{2\pi}\sigma^3} e^{\frac{-X^2}{2\sigma^2}}, \quad \sigma = 2 \tag{4-90}$$

on μ_{AP} through a convolution with

$$AC(i) = \left(\sum_{j=-6}^{6} Gaussian'(j) * \mu_{AP}(i-j) \right)^2 \tag{4-91}$$

to obtain an audio level change function *AC*. *σ=2* is chosen as appropriate filter parameter, because meaningful scenes contain about four of more shots. The audio change curve represents the audio change intensities and with its first derivative audio power level change instances *APC* are identified,

$$APC(i) = \begin{cases} 1 & if \quad AC'(i-1) \geq 0 \wedge AC'(i) < 0 \\ 0 & else \end{cases} \tag{4-92}$$

where *i* represents the shot number, but also the shot boundary instance. The analysis of the detection results after applying parallel shot post-processing and the misalignment jitter $j_{APC}=[0,1,2]$ confirm as well for this method the difficulties with audio, i.e. a high over segmentation resulting in an insufficient by low precision and, hence, no robustness improvements, as shown in Figure 112.

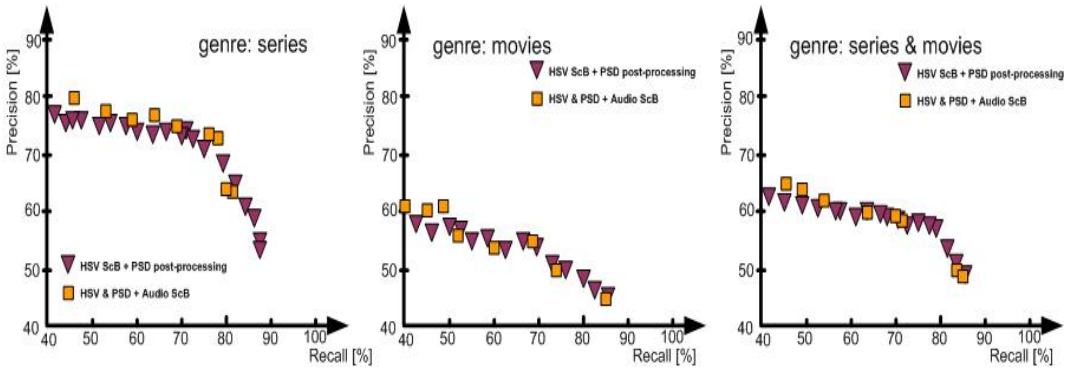


Figure 112. Results of HSV scene boundary detection with audio power change analysis.

Conclusions about audio based scene boundary detection

In this section 4.6.3, we applied various audiovisual methods, i.e. correlation of cut transitions with silences, audio class transition analysis and audio power transition analysis, to obtain independent features for our audiovisual scene boundary detection solution. Unfortunately the researched methods and applied audio classification tools are not sufficiently reliable, i.e. over detection or false detections, and, hence, we decide to exclude audio segmentation from the further analysis in this work.

Nevertheless, as the manually annotated ground truth of audio scene boundaries unveils, as shown in Annex 6, audio exhibits changes close to scene boundaries and, hence, it is necessary to further research audio segmentation algorithms to improve their robustness for the future and to combine them with visual scene boundary detection methods to multi-modal scene boundary detection solutions.

4.6.4 Shot-length-based scene boundary detection

Another independent parameter we investigate is the shot duration, a.k.a. *shot length* SL, which is reverse proportional to the shot boundary frequency or cut instances per hour. Shots of narrative content, e.g. series and movies, are concatenated during the post-production process following a film grammar like structure, as described in 4.5.1. Directors, cutters or producers use this attribute as artistic tool to create e.g. certain emotions. Action scenes, for example, are created by deliberately decreasing the average shot length, as stated by Faulstich in [140] and [141]. On the contrary, artistic 'master shots', i.e. shots with a high semantic meaning, are by purpose very long to convey a specific story message. Shots of suspense scenes are also rather long and include specific shot types, i.e. close or medium shots, as explained in 4.5.1, which leads to a kind of claustrophobic impression. Furthermore, shot durations are also genre dependent. In this section we intend to use the shot length to identify lengthy shots bordering scene boundaries.

Using the corpus, we evaluate an indicative overview of shot boundary frequencies, summarized in Table 45, showing that especially channel and commercial advertisements have a deliberately high transition frequency often used by commercial block detectors, as described in 4.4. This can be also deduced from the shot duration histograms – clustered in 50-, 20- and 5-frames-length bins - shown in Figure 113 and Figure 114 for various genres.

The narrative nature of AV contents, conveying a complex message to the viewer, requests for short shots inside individual scenes following rhythmic relations, i.e. the shot length or in other words shot cut tempo have to be constant within scenes.

Table 45. Average number of shot boundaries per hour for various genres.

Genre	MPEG-7 genre	Cut Instances / hour	Average shot length [sec]	Gradual Transitions / hour
Channel adds	Information\\Information/tabloid	2000	2	300
Commercial adds	Information\\Leisure	1900	2	300
Talk shows	Entertainment\\Talk Show	900 – 1300	3 – 4	30
Action movies	Movies\\Action	700 – 1300	3 – 5	0
Series	Drama\\Popular Drama	600 – 1200	3 – 6	30
Magazines	Information\\General Non-fiction topics	750	5	80
News	Information\\General Non-fiction topics	700 – 1100	3 – 5	80
Cartoons	Drama\\Animated	700 – 900	3 – 5	200
Quiz shows	Entertainment\\Quiz	700	5	30
Romantic movies	Movies\\Romance	700	5	0
Sport items	Information\Sport events	600 – 900	4 – 6	50
Documentaries	Information\\Documentary	500	7	100

On the contrary, at the beginning and the end of scenes the director uses so-called establishing (or introduction) and conclusion shots, as described in section 4.5. These two shot classes convey a lot of scenery information and complexity and, hence, they are in general medium or long shots. The shot duration is, therefore, distinctively longer than the average shot duration within a scene, which is elaborated further in this section by us.

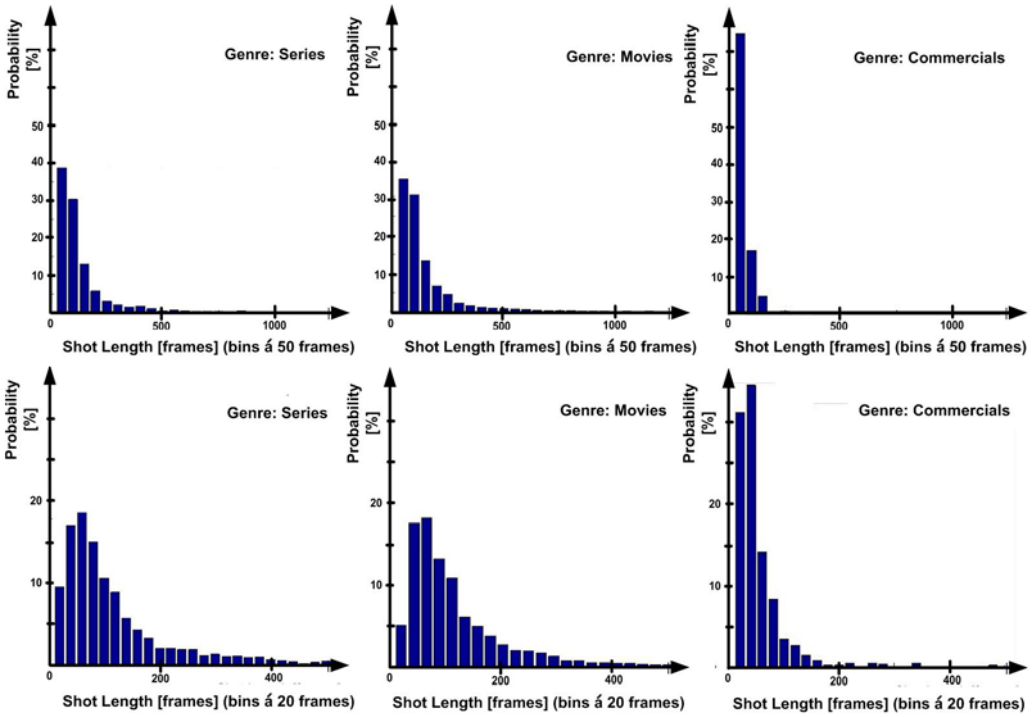


Figure 113. Shot length distribution for three genres (with 50- and 20-frame bins).

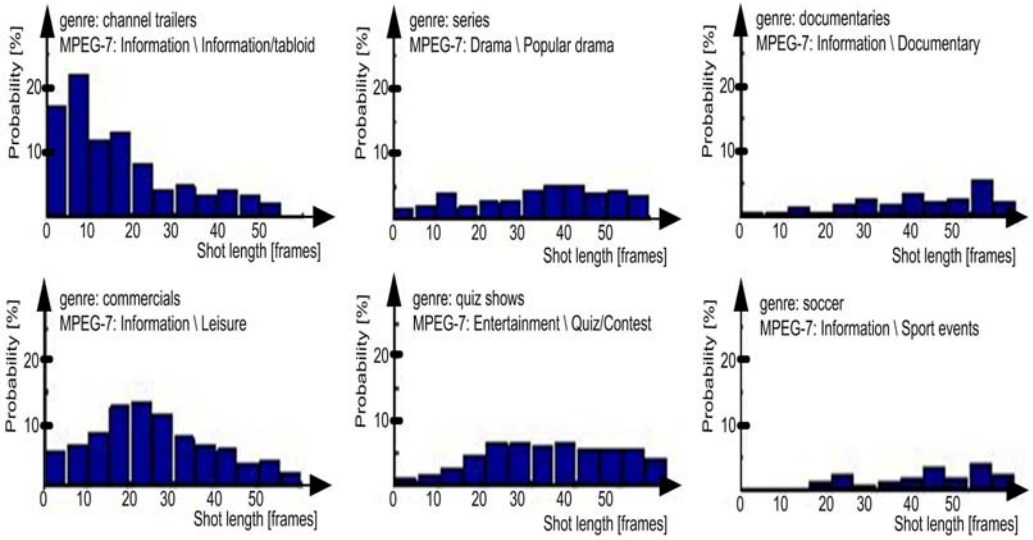


Figure 114. Shot length distribution (extreme zoom, 5-frame bins).

Our goal is to improve the robustness of the scene boundary detector of 4.6.2 in terms of precision being willing to scarifying recall to some extend, because one of the intentional applications is to identify correct scene boundaries with a high precision likelihood to insert at these instances automatically other content. Especially commercial audiovisual Internet video portals are interested to insert advertisements to foster a new business model.

Statistical analysis for establishing and conclusion shot duration

Our analysis unveils that directors deviate from the strict rule to use for each scene simultaneously an establishing shot, i.e. the first shot of a scene, and a conclusion shot, i.e. is the last shot of a scene. As a first step, we make a statistical evaluation of the shot length of both (a) establishing shots and (b) conclusion shots. Here fore we perform a manual shots length evaluation, i.e. visualizing the shot length before and after a shot boundary for all non-scene shot boundaries *SB* (red crosses in Figure 115, including shots of parallel shots) and all shot boundaries being a scene boundary *ScB* (black triangles in Figure 115).

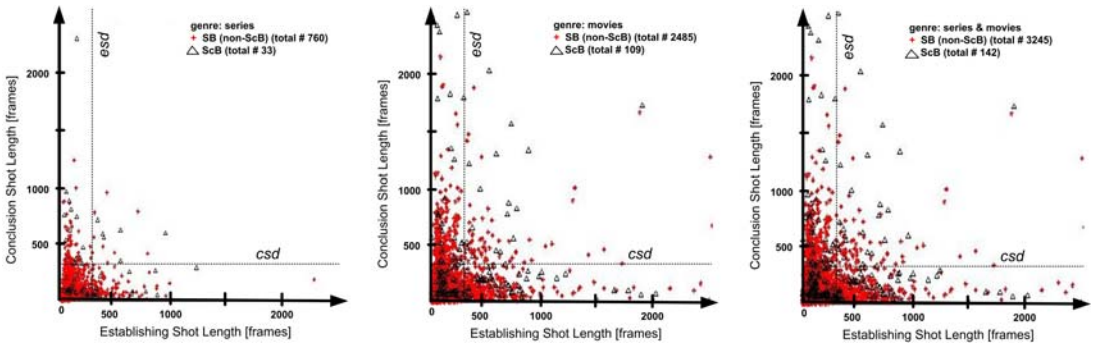


Figure 115. Establishing and conclusion shot length analysis.

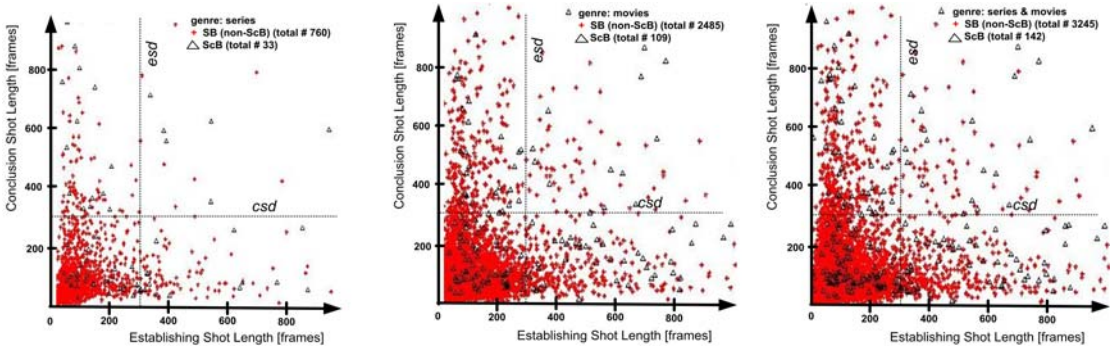


Figure 116. Establishing and conclusion shot length analysis (zoomed).

These points are visualized in Figure 115 and in its zoomed out version in Figure 116 including two threshold lines, i.e. establishing shot duration *esd=300 [frames]* and conclusion shot duration *csd=300 [frames]*. The figures unveil that a significant number of scene boundary instances exhibited long duration shots in their close neighbourhood, but little of them are simultaneously bordered by long duration establishing and conclusion shots.

Hence, we identify potential scene boundaries through averaging the shot length of predecessor and successor shot of each shot boundary $\mu_{SL}=½*(sh_N+sh_{N+1})$ and to statistically derive an average shot length threshold Th_{SL} identifying scene boundaries. Here fore, we create two clusters, i.e. shot boundaries being a scene boundary ScB and those no being one NScB, for the statistical analysis.

For the statistical analysis we apply a single sided *probability density function* pdf, i.e. pdf with Weibull distribution [142] using the average shot length $j=\mu_{SL}$, a shape parameter k and a scale parameter λ, which results in

$$pdf(j;k,\lambda)=\frac{k}{\lambda}*\left(\frac{j}{\lambda}\right)^{(k-1)}e^{-\left(\frac{j}{\lambda}\right)^{k}} \quad (4\text{-}93),$$

For the specific case of this section we choose the special case of a Weibull distribution with $pdf(j;k=2,\ \lambda=\sqrt{2}*\sigma)$ resulting in a pdf with Rayleigh distribution,

$$pdf(j\mid H)=\frac{j}{\sigma^2}*e^{\left(-\frac{j^2}{2\sigma^2}\right)} \quad (4\text{-}94),$$

wherein the maximum likelihood estimation of σ is represented by

$$\sigma\approx\sqrt{\frac{1}{2N}\sum_{i=0}^{N}j_i^2} \quad (4\text{-}95).$$

Using the pdf in the maximum likelihood ratio, as applied in 0 and presented in [117] and [37], we obtain

$$\frac{L_{ScB}}{L_{NScB}}=\frac{\frac{j}{\sigma_{ScB}^2}e\left(-\frac{j^2}{2\sigma_{ScB}^2}\right)}{\frac{j}{\sigma_{NScB}^2}e\left(-\frac{j^2}{2\sigma_{NScB}^2}\right)}*\frac{P(H_{ScB})}{1-P(H_{ScB})}\begin{matrix}>\\<\end{matrix}Th_{SL}\begin{matrix}\Rightarrow H_{ScB}\\\Rightarrow H_{NScB}\end{matrix} \quad (4\text{-}96),$$

using the same notations as in 4.6.3. With the thresholds derived from annex 8, we are able to calculate statistically the required threshold Th_{SL}. For the statistical analysis we take the same approach as in 4.6.3, i.e. the first half of each content item of the corpus sub-set is used for training (training set) and the second half for testing (test set). The with the training set statistically derived optimal threshold setting results in $Th_{SL}=258$ [frames] for series and movies, but due to their genre specific narrative structure and film grammar individual evaluations result in $Th_{SL}=205$ for series and $Th_{SL}=289$ for

movies. Using the statistically derived threshold Th_{SL} with our shot length based scene boundary detector in combination with our parallel shot post-processing, hence, not using our HSV scene boundary detector here, we witness that ~50% of ground truth scene boundaries are detected appropriately (column 'Correct ScB' in Table 46). Because many non-ScB instances exhibit long shot durations (see Figure 115, Figure 116 and column 'False ScB' in Table 46), the shot length based method leads to high over-detection and performs worse than the HSV based method (see Table 40). Nevertheless, the shot length based method is valuable for our approach, because our aim is to increase the precision of our HSV based method developed in 4.6.3.

Table 46. Results of shot length ScBD after parallel shot detection with $Th_{SL}=258$.

	# GT ScB	Correct ScB	Missed ScB	False ScB	Re [%]	Pr [%]
Series	69	32	37	83	46.4	27.9
Movies	170	97	73	382	57,1	20.3
Series & Movies	239	129	110	465	54.0	21.7

Conclusions of shot length based scene boundary detection

The assumption derived from film grammar rules that scenes, and hence scene boundaries, are bordered by setting exposing medium and long distance shots, which by definition are of long duration, was proven by us to be correct for more than half of the ground truth scene boundary set, as can be seen in Table 46. The seldom-isochroous appearance of establishing and conclusion shots at individual scene boundaries justifies our approach to combine the shot length into one threshold parameter Th_{SL}. Nevertheless, because of the high over-detection but acceptable recall we decide to combine this shot-length-based feature with the independent scene boundary detection features, i.e. HSV based scene boundary detector of 4.6.3, to increase the robustness of the overall scene boundary detector. We, nevertheless, believe that future research on establishing and conclusion shot detection, a.k.a. objective shot detection, by means of not only shot length, but also non-presence of faces, camera motion and texture, will be valuable to further improve the scene boundary detector.

4.6.5 Results of combined scene boundary detection system

Finally, we combine our parallel shot detector of section 4.6.2, our HSV scene boundary detector of section 4.6.3, which is based on the knowledge of cinematographic non-uniformity between subsequent scene, and our independent feature of shot length of section 4.6.4. The system, therefore, identifies first with our HSV scene boundary detector all potential scene boundary instances. Here after, our parallel shot detector filters the latter and only these located outside parallel shot sequences are provided to the shot length based analysis. These instances, which fulfil the criteria of the latter (section 4.6.4), are indexed as scene boundaries.

The analysis is done using a broad range of settings, i.e. window length $W_{sh} \in [3..30]$, $Th \in [1.2 \ldots 3]$, $jitter \in [0 \ldots 3]$ and $Th_{SL} \in [50 \ldots 300]$, and unveils that the best performances are achieved with low thresholds during the HSV scene boundary detection phase, i.e. high jitter *j*, low window length W_{sh} and a low passing threshold *Th*, followed by a rigid post processing, i.e. high shot length threshold Th_{SL}. In this way almost all correct scene boundary passes the first step together with a reasonable number of false detection, which, here after, are filtered out by the rigid post processing.

An example subset of HSV scene boundary settings is presented in Table 47 with their achieved detection results after parallel shot post-processing. Some representative results after performing shot length post processing (section 4.6.4) are shown in Table 48. As presented Table 48 and in Figure 117 the post-processing increases the overall robustness significantly towards higher precision at reasonable losses of recall, which confirms that directors and producers follow the film grammar rule to begin and/or end a scene with long lasting shots. The latter are expected to be long or medium establishing or conclusion shots and a shot length is only a first primitive way to identify and use them.

Table 47. Example subset of HSV scene boundary settings with detection results (after parallel shot processing, i.e. methods of section 4.6.2 and 4.6.3).

	jitter	W_{sh}	Th	Correct ScB	False ScB	Missed ScB	Re [%]	Pr [%]
Series & Movies	1	10	2.4	143	99	96	60.6	59.1
	2	20	3	128	78	111	53.0	61.6
	3	10	2	173	119	66	72.0	58.8
	3	6	1.4	204	206	35	85.2	49.4
Series	3	10	2	55	22	15	77.9	70.7
	3	6	1.4	58	38	12	82.4	59.6
Movies	3	10	2	118	97	51	69.6	54.7

Table 48. Results of combined scene boundary detector
(i.e. methods of section 4.6.2, 4.6.3 and 4.6.4).

	jitter	W_{sh}	Th	Th_{SL}	Correct ScB	False ScB	Missed ScB	Re [%]	Pr [%]
Series & Movies	3	6	1.4	100	179	99	60	74.6	64
	3	6	1.4	150	140	60	99	58.1	69.5
	3	10	2	100	148	60	91	61.4	70.7
	3	10	1.6	100	159	71	79	**66.5**	**68.9**
	3	10	1.6	150	129	42	110	53.4	75.0
Series	3	6	1.4	50	57	17	13	80.9	76.4
	3	6	1.4	75	55	14	15	**79.1**	**78.9**
	3	8	1.4	100	49	10	21	69.1	82.5
	3	10	2	100	46	8	24	64.7	84.6
	3	10	2	200	23	2	47	30.9	91.3
Movies	3	6	1.4	100	132	88	37	77.4	59.6
	3	8	1.4	150	110	56	59	**66.1**	**65.8**
	3	10	1.6	150	90	36	79	53.0	71.2
	3	10	2	150	81	30	88	47.5	72.2
	3	10	2	200	63	18	106	36.3	77.2

Furthermore, as already seen in previous analysis sections series adhere much stricter to the rules than movies, hence, recall- and precision-wise the scene boundary detection system performs, as expected, much better in series than in movies.

Moreover, shots encapsulating scene boundaries in series are shorter than movies as implicitly included in Table 48, i.e. shorter threshold Th_{SL} for optimal results, which is conform with the findings of 4.6.4, where we came to the same conclusion. Unfortunately, we are short of time to further research establishing and conclusion shots to render more precisely scene boundary instance, which in the course of this work are still allowed to be slightly dislocated, i.e. jitter *j*.

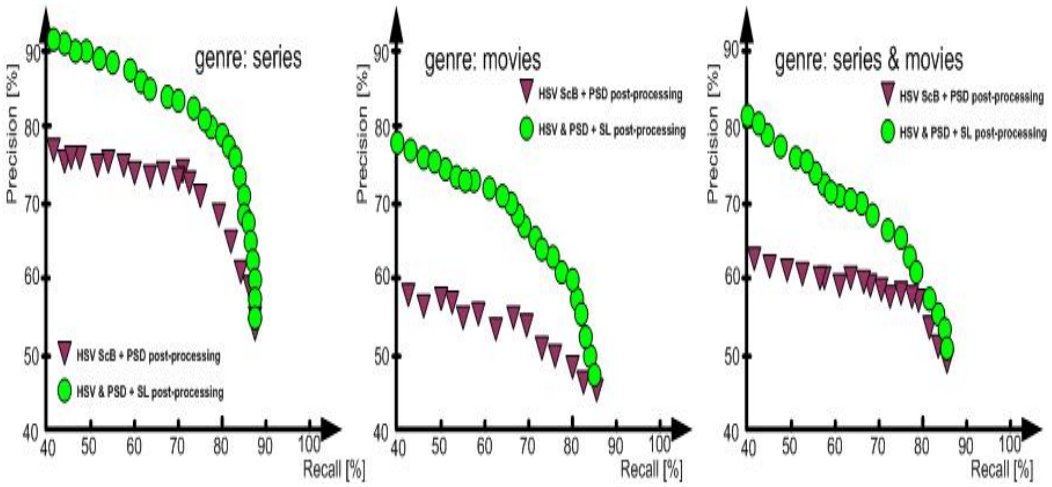


Figure 117. Results of combined scene boundary detector.

To emphasize once more the problem of long lasting establishing and conclusion shots at scene boundaries resulting in miss detections if using e.g. only a color based segmentation algorithms, we summarized the results with a low scene boundary jitter *j=0* and *j=1*, which result in low recall and precision justifying the use of higher jitter values in the course of this work, i.e. *j=3*. The results of the HSV-based scene boundary detector with the settings W_{sh}*=10*, *Th=2.4* and jitter=0 are summarized in Table 49 (for series) and Table 50 (for movies) and for jitter=1 in Table 51 (for series) and Table 52 (for movies).

Table 49. HSV-based scene boundary detector with jitter=0 in series.

series , jitter=0	'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total
# ScB GT	7	16	11	17	19	70
Correct	1	11	11	3	15	40
Missed	6	5	0	14	4	29
False	7	3	1	7	17	35
Re[%]	14.3	66.7	100	17. 7	79.0	57.1
Pr[%]	12.5	76.9	91. 7	30	46.9	53.3

Table 50. HSV-based based scene boundary detector with jitter=0 in movies.

movies , jitter=0	'nl'	'ge1'	'ge2'	'us_ana'	'us_dig'	Total
# ScB GT	35	25	30	53	26	169
Correct	18	11	14	11	14	68
Missed	17	14	16	42	12	101
False	44	37	6	22	25	134
Re[%]	50.0	44.0	46.7	22.2	53.9	40.2
Pr[%]	27.9	22.9	70.1	35.3	35.9	33.7

Table 51. HSV-based based scene boundary detector with jitter=1 in series.

series , jitter=1	'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total
# ScB GT	7	15	11	17	19	69
Correct	1	13	11	6	18	49
Missed	6	2	0	11	1	20
False	7	0	1	4	13	25
Re[%]	14.3	86.7	100	35.3	94.7	71.0
Pr[%]	12.5	100	91.7	60.0	58.0	66.2

Table 52. HSV-based based scene boundary detector with jitter=1 in movies.

movies , jitter=1	'nl'	'ge1'	'ge2'	'us_ana'	'us_dig'	Total
# ScB GT	34	25	30	54	26	169
Correct	29	13	16	18	16	92
Missed	5	12	14	36	10	77
False	32	35	3	17	23	110
Re[%]	85.3	52.0	53.3	33.3	61.5	54.4
Pr[%]	47.5	27.1	84.2	51.4	41.0	45.6

Furthermore, the results of the HSV-based scene boundary detector in combination with our parallel shot detector post-processing with jitter=0 are summarized in Table 53 (for series) and Table 54 (for movies) and for jitter=1 in Table 55 (for series) and Table 56 (for movies).

Table 53. HSV based scene boundary detector with jitter=0 and PSD filter in series.

series , jitter=0	'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total
# ScB GT	7	15	11	17	19	69
Correct	1	10	11	3	15	40
Missed	6	5	0	14	4	29
False	6	2	1	3	13	25
Re[%]	14.3	66.7	100	17.6	79.0	58.0
Pr[%]	14.3	83.3	91.7	50.0	53.6	61.5

Table 54. HSV based scene boundary detector with jitter=0 and PSD filter in movies.

movies , jitter=0	'nl'	'ge1'	'ge2'	'us_ana'	'us_dig'	Total
# ScB GT	34	25	30	54	26	169
Correct	17	11	14	11[16]	14	67
Missed	17	14	16	43	12	102
False	41	32	3	11	17	104
Re[%]	50.0	44.0	46.7	20.4	53.8	39.7
Pr[%]	29.3	25.6	82.4	50.0	45.2	39.2

Table 55. HSV based scene boundary detector with jitter=1 and PSD filter in series.

series , jitter=1	'nl1'	'nl2'	'ge1'	'ge2'	'gb'	Total
# ScB GT	7	15	11	17	19	69
Correct	1	13	11	6	18	49
Missed	6	2	0	11	1	20
False	6	0	1	1	10	18
Re[%]	14.3	86.7	100	35.3	94.7	**71.0**
Pr[%]	14.3	100	91.7	85.7	64.3	**73.1**

Table 56. HSV based scene boundary detector with jitter=1 and PSD filter in movies.

movies , jitter=1	'nl'	'ge1'	'ge2'	'us_ana'	'us_dig'	Total
# ScB GT	34	25	30	54	26	169
Correct	28	13	16	16[16]	16	89
Missed	6[16]	12	14	38	10	80
False	29	32	1	7	15	84
Re[%]	82.4	52.0	53.3	29.6	61.5	**52.7**
Pr[%]	49.1	28.9	94.1	69.6	51.6	**51.5**

The jitter results in slightly dislocated detection results, which are acceptable for the application in mind, i.e. as service support tool for video portals. In the near future research will be required to apply post-processing algorithm to specify the exact scene boundary instances on top of the current detection solutions.

[16] Narrative scene boundary became part of a semantic parallel shot (second PS ground truth), i.e. two narrative events with interleaved parallel shot sequences.

As derived form the results of Table 55 and Table 56, recall and precision were due to the rigid jitter of *j=0* and *j=1* quite low (series: R/P=71%/73%, movies: R/P=53%/52%, series & movies: R/P=59%/51%). Therefore, we apply a jitter of *j=3*.

Hence, with the here described approach we are able to reach for film grammar conforms behaving series recall and precision values of almost *80%* (Table 48 and in Figure 117). On movies, which often included film grammar deviating artistic elements at scene boundaries, the detection rate is lower reaching recall and precision values of *66%* (Table 48 and in Figure 117), which is approximately as well the level of detection for series and movies together. The results would be slighter higher if, as mentioned in the beginning of 4.6.2, scene boundaries at the boundaries of the content and scene boundaries with one-shot distance are either detected with a slightly adapted approach.
The final results were achieved applying optimal performance, i.e. ground truth, of individual predecessor Service Units enabling an objective development of analysis algorithms. Real analysis results of the cut detector, i.e. 98.3% / 98.3%, had no impact on the scene boundary detector but minor on the Parallel Shot Detector. The detection results of the gradual transition detector, i.e. 40% / 60%, were due to the jitter=3 of the scene boundary detector neglectable, but would have impact at a jitter=0. Applying real detection results of the Parallel Shot Detector, i.e. 83% / 83%, decreased the robustness of the Scene Boundary Detector to 70% and 55% for series and movies, respectively. Finally, we integrate the combined scene boundary detector as *Scene Boundary Detection* service unit into the overall framework, as sketched in Figure 118, with an XML-based MPEG-7 compliant metadata output, according to the example shown in Figure 119.

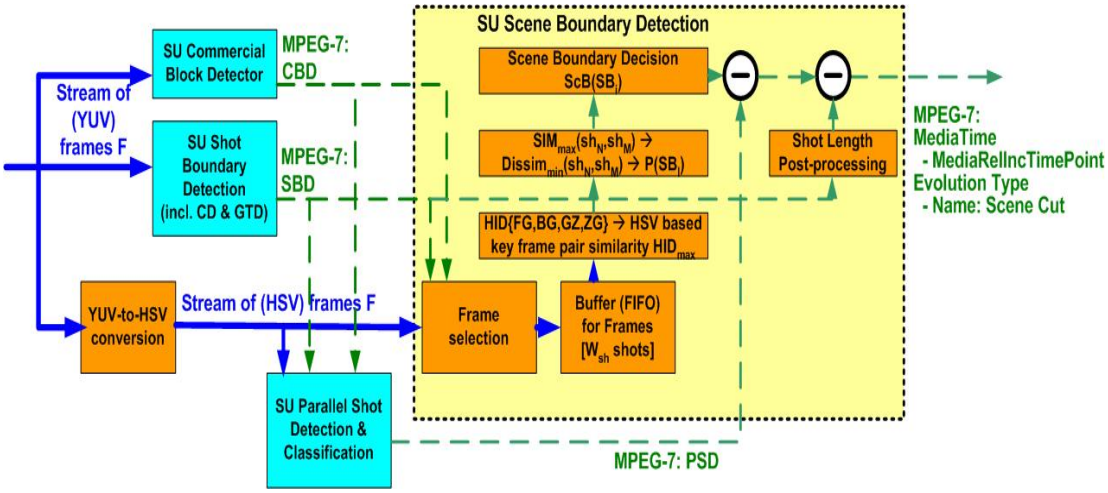


Figure 118. System integration of combined scene boundary detector.

Scene Boundary

```
<Description xsi:type="ContentEntityType">
 <MultimediaContent xsi:type="VideoType">
  <AnalyticEditedVideo xsi:type="EditedVideoType">
   <AnalyticEditingTemporalDecomposition gap="false" overlap="false">
    <GlobalTransition>
     <MediaTime>
      <MediaRelIncrTimePoint mediaTimeUnit="PT1N1000000F">1
      </MediaRelIncrTimePoint>
     </MediaTime>
     <EvolutionType href="urn:mpeg:mpeg7:cs:EvolutionTypeCS:2001">
      <Name xml:lang="en">Scene</Name>
     </EvolutionType>
    </GlobalTransition>
   </AnalyticEditingTemporalDecomposition>
  </AnalyticEditedVideo>
 </MultimediaContent>
</Description>
```

Figure 119. XML-based MPEG-7 description of scene boundary instances.

4.6.6 Conclusions of audiovisual segmentation of filtered content

In section 4.1 we elaborated our shot segmentation method, eliminated with our method of section 4.4 non-content related entities and clustered with our method described in section 4.5 parts of the remaining shots into parallel shots.

In this section we elaborated in 4.6.2 an HSV-based scene boundary detector, which aim to identify discontinuities in the video flow applying HSV-based colour features. In combination with our parallel shot detector we achieved a high recall of ~85% with a low precision of ~50%. To improve the low precision we elaborated several methods enhancing the results. In section 4.6.3 we elaborate methods based on silence detection and audio scene boundary detection, but the results remained behind our expectations. Hence, we decided to use another independent feature derived from the knowledge of film grammar from section 4.5, i.e. the discriminative length of establishing and conclusion shots. We described the method in section 4.6.4 and the results confirmed our theory. We, therefore, combined the method of section 4.6.2, i.e. HSV-based scene boundary detector, with our shot-length based post-processing (section 4.6.4) and parallel shot filtering approach (section 4.5.2). With our approach we achieved a recall and precision of almost 80% for series and ~66% for movies (as summarized in Table 48 and in Figure 117).

4.7 Conclusions of audiovisual segmentation

In this chapter we presented our individual service unit elements required to segment content into its objective, but also subjective semantic audiovisual elements. In 4.1 we presented several of our low-level and mid-level video analysis service units to e.g. segment content into its individual shots. We developed, here fore, various new shot boundary detectors, i.e. a compressed domain macroblock correlation cut detector, a field difference cut detector, a color segmentation based cut detector and benchmarked them against a representative AV corpus. Our filed difference cut detector *FD CD* proved to be the most reliable one with recall and precision reaching 96%, as presented in 0. Hence, we used the FD CD as basis and elaborated several enhancement service units using feature-point-based similarity analysis. In combination with the enhancements the FD CD reached recall and precision values of 98.3%, as presented in 0. Subsequently, we integrated the FD CD with its enhancement service units and our gradual transition detector into our framework as service units.

Here after, we aimed to clean the content item from non-content item related inserts, i.e. commercial blocks. Here fore we developed task oriented low-level and mid-level audio features (section 4.2), which we applied in section 4.4 to build a commercial block detector for detecting and subsequently deleting non-content related inserts.

In 4.5 we then presented content production related know-how, i.e. film grammar, which describe the rules of content production.

We applied the knowledge of film grammar to elaborate our parallel shot detector in section 4.5.2, which based our analysis can be used to cluster up to 70% of all shots into parallel shot entities, i.e. interleaved narrative sequences. In section 4.5.2 we described several of our methods for parallel shot detection, i.e. HSV-based key frame pair analysis, HY-based key frame based analysis, ScaFT-based key frame based analysis and SIFT-based key frame based analysis. Our benchmark, summarized in section 4.5.2, unveiled that our HSV- and HY-based methods outperformed the other two. We decided, therefore, to select the processing-wise more efficient HSV-based method. Our HSV-based parallel shot detector achieved a recall~75% and precision~90%, as summarized in the end of that section.

In the subsequent step, we developed a method to detect content flow dissimilarities, which are representative for scene boundaries. In section 4.6.2 we presented our HSV-based scene boundary detector, which detected color dissimilarity instances in the content flow. With our HSV-based scene boundary detector in combination with our parallel shot detector we reached a high recall=85%, but only a low precision~50%. We,

therefore, aimed to identify independent features to enhance the precision. In section 4.6.3 we elaborated several audio related segmentation methods using audio features, such as audio silence, audio scene segmentation. The audio-based methods did not reach the detection levels expected. Hence, we elaborated anther independent feature derived from film grammar rules, i.e. shot length. In section 4.6.4 we described our shot-length-based scene boundary detector, which proved to be a distinctive feature for scene boundary detection. We, therefore, combined the method of section 4.6.2, i.e. HSV-based scene boundary detector, with our shot-length based post-processing (section 4.6.4) and parallel shot filtering approach (section 4.5.2). Finally, we achieved with our service unit scene boundary detection SU ScBD a recall and precision of almost 80% for series and ~66% for movies (as summarized in Table 48 and in Figure 117).

We have to state that all features we used for the detection of boundaries of semantic scenes can be quantified as ‘low-level’ and ‘mid-level’ from a semantic point of view. By intention, we did not included ‘high-level’ features such as objects and speakers, e.g. as we present in [137], in this work to keep our solution generic and most processing efficient, as required for our target platform. Nevertheless, to achieve better recall and precision, these high-level features are surely useful and have to be taken into consideration, when enhancing the current system.

CHAPTER 5

5 Conclusions and perspectives

Conclusions (in English)

The ambitious aim of this work was providing a technical sound, intuitive and lean-backward-oriented browse- and navigation solution for consumer electronics devices meeting consumer's current need managing his / her audiovisual content archive. Consumer's positive acceptance of chapter makers provided with commercial content on purchased carriers, such as with DVDs, increased consumer's desire having such a chaptering solution as well for recorded broadcast content. Furthermore, our market analysis unveiled a strong consumer desire skipping undesired advertisements, while watching e.g. recorded items on their Personal Video Recorders PVRs. We applied these market insights when selecting our target application, i.e. semantic segmentation of audiovisual content items resulting in semantic audiovisual scenes. Furthermore, we aimed providing the market with a commercial skipping application, which we expected being a low-effort 'by-product' of our work.

In order to provide the market with a suited technology- and consumer solution at the appropriate time, we thoroughly researched today's technology- and consumer trends as summarized in chapter two. The selected technology trends investigated covered those of processing, storage and connectivity. The researched consumer trends were

related or at least triggered by the technology trends and their solutions introduced into the market. The consumer trend analysis unveiled the expressed consumer desire for ubiquitous, ambient, intuitive, content- and context-aware solutions and applications. Consumer's request for intuitiveness and content-awareness endorsed our application choice, i.e. for semantic scene segmentation. Achieving the latter we had to research the underlying content-awareness creation in more detail, which meant creating metadata, i.e. data describing the content, by means of content analysis algorithms. Realizing the semantic nature of our semantic scene segmentation task we identified soon the need but also at the same time the opportunity to exploit more than just one single modality of our audiovisual content. Hence, we were facing the issue of syndicating many single modality content analysis solutions into one prototyping system allowing us to exploit not only low- / mid-level features but also features at a semantic level. The technology trends unveiled that former individual consumer electronics were syndicated today by means of today's In-Home networks sharing their processing and memory transparently. The latter allowed distribution of e.g. content-awareness creation processes across the network. These insights were the basis for our decision applying a *Service Oriented Architecture* SOA based technology for our work resulting in a SOA based prototyping framework. We embedded each individual content-awareness creating single modality solutions into one component, a so-called *Service Unit*. Each of the latter communicated with the system through standardized interfaces and in this way the components became transparent to the system, but at the same time the system became transparent to the content analysis component. This approach allowed the efficient usage of the resources of the network. During our research we identified as well several robustness and maintenance issues when applying SOA based technology for our prototyping system. In the second part of chapter two we describe our research of several components, such as our *Connection Manager* and *Health Keeper*, solving these identified issues. Finally, in the end of chapter two we introduced a set of selected individual content analysis *Service Units*, which, when syndicated together, formed our envisioned semantic content segmentation application solution.

Finally, our SOA based and distributed framework enabled an efficient and seamless collaboration of multiple teams, each of them having expertise in one specific content analysis modalities, i.e. speech, music, image or video. But even more important for us, this framework allowed an efficient research of multi-modality based content analysis solutions and application. Furthermore, our prototyping system capabilities allowed us to simulate the target platform and hence to verify the performance before integration onto e.g. an embedded platform.

In a sub-sequent step we studied thoroughly a selected group of segmentation- and classification state-of-the-art technologies and the here for required feature parameter solutions. We did this not only to streamline our work with available works, but also to be able exploiting, if appropriate, those technologies or their underlying concepts efficiently. For each of these selected technology groups we collected and evaluated the relevant state-of-the-art solutions based on their suitability for our specific application, but also on the technology's maturity and reliability, i.e. robustness. The description of the technologies and our evaluations were summarized in chapter three of this work. Furthermore, special attention in this state-of-the-art-analysis was given to video segmentation-based technologies such *shot*- and *scene* segmentation. The results of this study served as basis for the decision, which technology blocks reached maturity and which had to be researched even further by us in order to meet our requirements. Unfortunately, neither the shot- nor the state-of-the-art scene segmentation technologies met our criteria. Hence, we decided to research them in more depth.

In the state-of-the-art analysis we identified, furthermore, an issue with semantic scene segmentation, i.e. that non-content related inserts such as inserted commercials deteriorated the robustness of any automatic chaptering solution. We retrieved several interesting single- and multi-modality based commercial block detection solutions during our analysis. Unfortunately, neither these solutions were suited for our target platform nor they met our additional requirements for a dedicated commercial skip application. Hence, we researched several video low-level and mid-level parameters suited for the implementation on a dedicated video compressor. These features included features such as a *Monochrome Frame Detector*, *Interlaced-progressive Detector*, *Letterbox Detector* and *Shot Boundary Detector*. The first three features were compressed domain low-level features, which we developed using special parameters provided by our dedicated video compressor. In addition, the mid-level feature *Shot Boundary Detector* was of great importance for our work, because shots were seen as a kind of atomic unit of video, next to individual frames. The robustness of this feature was of utmost importance, because many segmentation and clustering solutions were based on shots. The analysis of the state-of-the-art shot boundary detection solutions - including those which participated in TRECVid - raised two shortcomings.

The first one was that none of the benchmark corpora applied resembled the content group, which was relevant for our target application and target user group, i.e. consisting of a variety of genres and derived from a variety of cultures. Hence, we researched the consumer behaviours of our target audience and established a representative benchmark corpus accordingly including the required ground truth data.

Subsequently, we applied this corpus for testing and benchmarking our researched analysis solutions, such as the *Shot Boundary Detector*.

The second shortcoming, we witnessed, was the reasonable but still improvable robustness of the *Shot Boundary Detectors* retrieved. Due to the importance of this feature we decided to research three new *Shot Boundary Detector*s, i.e. *Macroblock Correlation Cut Detector*, *Field Difference Cut Detector* and *Colour Segmentation Cut Detector*. We benchmarked them, using our benchmark corpus, against each other and against one detector derived from academia. The latter we used as objective reference as it participated in the TRECVid benchmark. Here after we selected the winner of the benchmark, i.e. *Field Difference Cut Detector*, and researched additional post-processing steps, e.g. using *Feature Points* and backwards analysis, boosting the robustness even further. The high detection results, i.e. recall and precision of 98.3%, were comparable to the winning TRECVid solutions. The high robustness in combination with our methods simplicity was the reason, why we integrated this detector as *Service Unit* into our framework.

But our analysis of semantic scenes showed that scene boundaries occurred as well at smooth video transitions, i.e. gradual ones. Hence, we decided to integrate a *Gradual Transition Detector*, which we identified during the state-of-the-art analysis as suited, and improved its robustness by means of some post-processing steps, as described in chapter four.

In order to exploit the multi-modal nature of our content efficiently, we researched, here after, audio related low-level and mid-level features, which we aimed to combine - similar to methods studied in the state-of-the-art - with video features. In particular, we researched a dedicated commercial silence detector for the compressed domain as described in section three of chapter four, which satisfied our requirements. We did so because in the state-of-the-art we retrieved dominantly general purpose silence detectors not suited for our purpose.

In the next part of our work we aimed to research a specialized genre detector, i.e. a commercial block detector, which had to identify non-content related inserts. But we also aimed to apply this solution for a commercial skip application. For this purpose we experimented with various video- and audio feature combinations and identified that the combination of our *Monochrome Frame Detector* with our *Commercial Silence Detector* performed best and outperformed robustness-wise the state-of-the-art commercial block detectors. The detection results, i.e. recall of 91.4 and precision of 99.93%, met the requirements of our commercial skip application. Hence, we integrated this detector as

Service Unit into our framework, which here after eliminated automatically all non-content related inserts, i.e. commercials inserted by the broadcasters. We described and summarized our work on the commercial block detector in the fourth section of chapter four.

Being very much aware of the subjective nature of our semantic scene segmentation task and triggered by some clustering technologies, which we retrieved during our state-of-the-art analysis, we decided studying the art of film production. With the knowledge of the latter we aimed extracting objective film grammar rules, which we witnessed, were commonly applied in the content production business. One of the many film grammar rules was related to interleaved narrative sequences, further referenced as *Parallel Shots*, often applied in narrative contents such as feature films and series. These *Parallel Shots* form semantic sub-entities, i.e. clusters. *Parallel Shots* were divided into two classes *Cross-Cuttings* and *Shot-Reverse-Shots*, where the latter were dialogues. By definition these *Parallel Shot* sequences did not contain any scene boundaries and, there for, we selected them to pre-cluster shots into such *Parallel Shots* prior *Scene Boundary Detection*. Our research showed that up to 70% of narrative content was clustered into such *Parallel Shots*, which we described in the fifth section of chapter four.
For the clustering of shots into *Parallel Shots* we decided to research four similarity based methods using the knowledge derived from the state-of-the-art analysis. For our analysis we developed two simple colour based methods, i.e. one based on HSV and one based on HY, and two *Feature Points* based methods, i.e. Scaft and SIFT. Our benchmark showed the potential of the *Feature Point* methods, but nevertheless in their current form they fall short compared to the simple colour based methods. The two colour based methods showed comparable detection results, hence, because of its processing-wise more efficient nature we selected the HSV based method, with which we reached recall and precision of about 83% for narrative content.

Finally, in the last section of chapter four, we aimed to identify scene boundary instances in the remaining parts of our content, i.e. after non-content related inserts were removed and up to 70% of the shots were clustered into *Parallel Shots*. For this purpose we researched technologies to identify discontinuities in the content flow. Applying a simple HSV based dissimilarity method we achieved detection results of 85% recall and of 50% precision.

Because of the low robustness of this solution we decided investigating independent features to increase the detection results. The selection of these additional features was partially based on the knowledge derived from our film grammar study.
The study of audio based scene boundary detection methods, unfortunately, did not resulted in sufficiently high detection results, mainly due to the limitation of the inherited audio classification component we applied for this purpose.
Fortunately, the specific nature of scenes and scene boundaries, as specified by film grammar, led to an independent shot-length based scene boundary detector, mainly exploiting the presents of extremely long establishing- and conclusion shots at scene boundaries. The combination of our shot-length based and HSV based *Scene Boundary Detector* enabled us boosting the detection rate of the semantic scene boundary detector to almost 80% for series and 66% for movies. Our market analysis showed that this achieved accuracy was sufficient high for our lean-backward oriented *Advanced Content Navigation* application. For the latter our solution identified automatically the boundaries of non-content related inserts, i.e. commercial blocks, and scene boundaries throughout the recorded content, in some cases with a certain offset. With our method described in Annex 10 the user was enabled to relocate slightly misaligned boundaries efficiently and subsequently burn the resulting clean chaptered content onto e.g. a DVD. Hence, finally we integrated our solution as *Service Unit Scene Boundary Detection* into our framework.

Furthermore, we identified several other application domains, which we were able to serve with our scene segmentation solution such as professional semi-automatic content indexing. In the case of the latter our solution might serve as pre-processing step for humans, who wish to manually annotate content for professional content management and service businesses. But the solution might also serve individual consumer device applications, where human involvement is accepted, e.g. correcting manually slightly incorrect detection results. We described this application, i.e. *Content Item Boundary Detection* CIBD and the manual post-annotation, in more detail in Annex 10 and our publications [12], [13] and [14]. The CIBD application clusters coherent chapters, i.e. scenes, together and identifies strong discontinuities, which are indicative for e.g. the start and end of a movie. Hence, applied in consumer devices, this enables consumers to obtain a clean recording of e.g. recorded broadcast content. Here after, we might apply our intuitive shot- and scene boundary based *User Interface* UI for manual post-annotation, with which consumers can easily eliminate shortcomings of the solutions' robustness, i.e. inaccuracies, as described in Annex 10.

Hence, in this work we proposed a low-level and mid-level approach for high-level semantic segmentation of AV content into semantically coherent chapters. The results achieved were satisfactory for a range of applications targeted for the consumer- and, partially, for the professional domain. Nevertheless, the problem was not solved entirely, i.e. the solution might be further enhanced. In addition, we believed that for a fully automatic content analysis solution more apriori knowledge, i.e. film grammar, and related to this, more mid-level and high-level features, have to be applied. The latter was also stated by Leonardi in [102], where he aimed syndicating top-down and bottom-up results. We believe that additional independent mid-level and high-level features, such as object recognition, speaker identification, but also features like focus and depth, contain valuable insights, which in combination with film grammar knowledge allows extracting deeper semantics about the content and its production concept. For example, speaker identification would help increasing the robustness of dialogue related analysis, i.e. Shot-Reverse-Shots. The modular and generic nature of our framework allows seamless, transparent and straightforward integration of new mid-level and high-level *Service Units* and, hence, the opportunity to efficiently develop an enhanced solution. Furthermore, the to some extend subjective nature of semantic scene boundaries justifies at the same time applying machine-learning solutions. But we believe that the latter should be based on film grammar compliant mid-level and high-level features rather than on low-level data.

Conclusion (en Français)

L‘objectif de ce travail de thèse de doctorat est d’explorer une solution intuitive et utilisant des techniques adaptées et orientées à la catégorie *lean-backward* permettant au grand public la navigation intégrée dans les équipements domestiques de stockage des données. De plus, cette solution doit satisfaire l’attente du consommateur en termes de facilité d’utilisation, notamment en ce qui concerne le classement des archives audiovisuelles. La segmentation d’un contenu en chapitres est très appréciée par le consommateur, étant donné que cette dernière est déjà à sa disposition dans certains matériels commerciaux comme par exemple dans les DVD. Ainsi, ceci explique le désir du consommateur de pouvoir réaliser le même travail de classifications sur ses vidéos personnelles. Ensuite, notre analyse du marché montre une demande très accentuée d’une autre application : l’élimination des passages publicitaires. Après quoi, nous avons formulé nos objectifs, à savoir la segmentation sémantique des données audiovisuelles. Avec cette approche, le contenu doit être classifié en scènes sémantiques. Enfin, notre travail doit offrir une solution d’élimination des passages publicitaires, une application qui – à coup sur – serrât un produit supplémentaire de notre recherche.

Apporter des technologies et des solutions souhaitables et actuelles au consommateur s nécessite l’analyse des tendances existantes, les résultats de cette enquête nous les présentons dans le deuxième chapitre. Nous avons étudié les évolutions techniques en termes de capacité de calcul, de capacité d’enregistrement et de largeur des bandes de transmission. L’analyse montre surtout que le consommateur s’attend à des solutions qui sont globales, intuitives et considérant le contenu et le contexte. Cette information nous a amené à nous orienter vers une application de segmentation. Ce choix est suivi par la nécessité d’analyser en détail les diverses technologies décrivant le contenu. Nous avons examiné de plus prés ces technologies qui génèrent des meta-données, permettant ainsi l’analyse des contenus par l’intermédiaire d’algorithmes spéciaux.
Due à la nature spéciale de la segmentation sémantique de notre domaine de recherche nous avons rapidement eu la possibilité, mais aussi le besoin, de considérer plusieurs modalités pour l’analyse (audio et vidéo). Cette pluralité avait pour but de trouver la solution adéquate et correcte pour l’intégration de plusieurs modalités en un

prototype commun afin d'obtenir des résultats surpassant les analyses mono-modales avec de nouveaux paramètres sémantiques.
L'analyse du marché nous a d'ailleurs indiqué que les équipements domestiques, travaillant auparavant en unités individuelles (mono-modale), sont reliés de plus en plus dans les ménages en réseau intégrés, partageant ainsi leurs capacités de calcul et capacités d'enregistrement. Cette réalité nous permet de localiser les solutions d'analyse du contenu au diverses endroits du réseau domestique. Cela nous a guidé pour choisir la méthode du *Service Oriented Architecture* (SOA), nous offrant ainsi un système orientée SOA, qui sera dorénavant notre base de recherche pour les activités futures. Nous avons donc intégré chaque unité individuelle dédiée à une tâche spécifique dans l'analyse du contenu, que nous avons appelée *Service Unit*. Chaque *Service Unit* est reliée avec le reste du système à l'aide d'une interface de communication. Les informations fournies par le système sont donc transparentes et nous donnent la possibilité de disposer efficacement des capacités du système global.
Au cours des recherches suivantes nous avons résolus quelques problèmes en vue d'améliorer la robustesse et les services dans notre système SOA. Dans la seconde partie du chapitre 2, nous décrivons quelques composantes résolvant les problèmes cités ci-dessus, puis nous décrivons le *Connection Manager* et le *Health Keeper*. A la fin du chapitre 2, nous présentons notre solution de segmentation sémantique composée de différentes *Sevice Units*.

Notre approche consistant à s'orienter d'abord vers les SOA individuelles, mais cohérentes, nous permet de former des familles différentes ayant une approche spéciale de l'analyse d'une modalité, comme par exemple la langue, la musique, l'image ou bien encore la vidéo, qui pourraient néanmoins être liées facilement et efficacement en un seul ensemble. De plus, très importante pour notre travail, cette approche de rapports interactifs nous permet d'examiner facilement des solutions analysant le contenant à caractère multi-modal. Notre approche nous offre, en plus, la possibilité de créer des prototypes simples et puissants. Ces dernières nous permettent, ainsi, de tester les résultats et les caractéristiques avant de les appliquer au système définitif.

Dans l'étape suivante, nous avons analysé un groupe de technologies sélectionnées de classification et de segmentation, en examinant de plus les paramètres exigés. Ce travail est nécessaire non seulement pour éviter les redondances, mais aussi pour pouvoir éventuellement les appliquer dans nos solutions à part entière ou au moins en partie dans la conception singulière. Nous avons collecté pour chaque groupe de

technologies toutes les informations existantes et faisons une évaluation en vue de l'application éventuelle dans notre système, critère décisif étant donné leur robustesse et la maturité de la technologie. Les détails de cette recherche ainsi que les évaluations sont donnés dans le chapitre 3. Une approche très approfondie de cette analyse des technologies s'intéresse aussi à la segmentation des vidéos, en *Shots* et en scènes. Ce groupe d'analyse est décisif pour le choix des technologies à appliquer dans le travail à suivre et nous donne les indications suivantes: quelles technologies devraient être examinées plus en détail pour trouver des améliorations nécessaires ayant comme but de les appliquer dans notre solution définitive. Aucune des technologies étudiées, ni les *Shots*, ni même d'autres technologies de segmentation appliquées au marché n'avaient des performances satisfaisantes : des améliorations étaient donc nécessaires.

En analysant les technologies présentes nous avons très vite rencontré un problème assez important affectant la segmentation sémantique et ayant un effet négatif au classement d'un contenu en chapitres: les informations additionnelles, surtout les publicités. Nous avons trouvé quelques détecteurs de publicités mono-modaux et multi-modaux, mais aucun d'entre eux n'avaient des qualités acceptables pour la solution envisagée et ne pouvait garantir une élimination automatique de ces segments publicitaires dans notre application. Par conséquent, nous avons examiné quelques descripteurs vidéo de bas et moyen au niveau qui pourraient être de bons candidats pour un détecteur global de vidéo compressée. Parmi ces descripteurs, nous nous sommes intéressé aux paramètres *Monochrome Frame* detector, au *Interlaced–progressive* detector, au *Letterbox* detector et au *Shot Boundary* detector. Les trois premiers éléments cités ci-dessus sont des paramètres de bas niveau du flux vidéo compressé. Nous avons modifié ces derniers à l'aide de paramètres spéciaux issus de la compression. Pour la suite de notre travail le descripteur de moyen niveau, *Shot Boundary* detector, s'avère de plus en plus important, puisque un *Shot*, de la même manière qu'une image individuelle, peut être classifié comme un élément de base (atome) d'un contenu audiovisuel. La robustesse de ce paramètre est décisive pour notre choix, puisque plusieurs solutions de segmentation et de clustering sont basés sur les *Shots*.

Après avoir analysé les différentes solutions d'aujourd'hui incluant les méthodes présentées au TrecVid, nous avons caractérisé deux aspects de défaillance. D'abord aucune de ces techniques n'appliquent les aspects pertinents et indispensables dans notre application et ils ne comprennent pas de contenus provenant de diverses genres, diverses cultures et pays. Afin de compenser et d'éliminer cette défaillance nous avons

analysé les habitudes du consommateur de notre groupe sélectionné, ensuite nous formulons divers tests nécessaires, ajoutons et compilons les données et les critères pertinents. Ces tests ont servis pour l'évaluation de nos solutions, par exemple pour tester notre propre *Shot Boundary* detector. La deuxième défaillance que nous avons décelée provient du fait que les nouvelles techniques de *Shot Boundary* detector ont une robustesse acceptable, mais du point de vue de nos objectifs de qualité ces derniers nécessitaient une amélioration. Tenant compte de l'importance de ce détecteur nous avons formulé et examiné trois détecteurs : le *Macroblock Correlation Cut* detector, le *Field Difference Cut* detector et le *Colour Segmentation Cut* detector. Nous avons testé et analysé ces trois détecteurs avec nos méthodes de test, mais aussi en appliquant les tests déjà utilisés ailleurs. Ce détecteur qui a participé à la campagne d'évaluation de *TrecVid* assure l'objectivité de notre propre évaluation.

Le meilleur détecteur selon l'évaluation est le *Field Difference Cut* detector. C'est celui-ci que nous avons pris comme base pour nos recherches futures. Nous avons apporté quelques améliorations : l'analyse basée sur les *Feature Points* et les analyses retro, avec comme but déclaré d'atteindre une qualité de robustesse supérieure. Le niveau de détection, rappel et précision, est de 98,3% étant comparables aux résultats de la solution présentée au *TrecVid*. Notre méthode se révèle beaucoup moins compliquée en comparaison de cette dernière nous facilitant, ainsi, la justification de prendre notre solution propre comme *Service Unit* dans notre système. Notre analyse des scènes sémantiques par notre détecteur a montré que la classification tenait compte des transitions nuancées, donc la classification générait un nombre trop important de fausses alarmes. Pour compenser cette sensibilité, nous avons ajouté un paramètre correctif, le *Gradual Transition* detector, ce dernier étant un bon candidat. Nous avons adapté ce détecteur à nos besoins en y ajoutant quelques modifications pour améliorer sa robustesse – voir chapitre 4.

Pour pouvoir appliquer la multi-modalité dans nos recherches nous avons pris en compte des paramètres de bas et moyen niveau en les combinant – comme c'est le cas dans les technologies du marché – avec des paramètres vidéo. Prenant un détecteur de silence de réclame spécialement étudié pour agir au niveau compressé, nous avons analysé et modifié ce détecteur, qui était plus ou moins suffisant pour nos objectifs. Une description de cette recherche est donnée dans le chapitre 4. Les modifications sont nécessaires puisque les détecteurs disponibles sur le marché comprennent seulement des paramètres généraux.

Dans la section suivante, nous avons examiné de plus près un détecteur de publicité spécialisé, lequel permettant de caractériser des inclusions (publicités) ajoutées au flux original. Nous avons testé la méthode apportée par cette solution, essayant d'appliquer ce détecteur comme méthode pour éliminer ces passages de réclames. Nous avons examiné plusieurs variantes en combinant des paramètres audio et vidéo, et enfin nous avons découvert, que la combinaison de notre détecteur du *Monochrome frame* et notre détecteur de silence de réclame donnait des résultats très satisfaisants et, en plus, surpassait considérablement la robustesse des détecteurs connues du marché. En ce qui concerne les performances de détection, nous avons une valeur de rappel de 91,4% et de précision allant jusqu'à 99,93% justifiant de prendre ce détecteur comme notre solution pour l'application d'élimination des passages publicitaires. Nous avons donc considéré ce détecteur comme *Service Unit* dans notre système global. Cette unité élimine dorénavant tous les passages ajoutés par les stations d'émission, surtout les réclames. Les étapes de la recherche à ce sujet sont décrites dans la quatrième section du chapitre 4.

Notre solution d'analyse et de segmentation des scènes sémantiques, ainsi que l'utilisation de quelques technologies faisant appel à la technique de contexte - découvertes au cours de notre analyse du marché - nous a incités à étudier de plus près la procédure commune de production d'un film. Nous avons tenté d'identifier les règles de grammaire objectives appliquées et avons appris que, ces règles sont effectivement existantes et en vigueur. Une de ces règles est la présence de séquences narratives interalliés l'une avec l'autre, appelées dans la suite *Parallel Shots*. Ces *Shots*, très représentés dans les séries ou films, font naître des sous-groupes sémantiques: les clusters.

Les *Parallel Shots* peuvent être sous-divisés en deux classes : *Cross-Cuttings* et *Shot-Reverse-Shots*, cette dernière étant des dialogues. Les *Parallel Shots* n'ont, par définition, aucun changement de scène. Nous avons sélectionné et classifié quelques uns de ces *Parallel Shots* pour être en position d'avoir une base nécessaire pour la recherche des changements de scènes. On peut constater que, presque 70% du contenu narratif peut être classifié dans la catégorie des *Parallel Shots* - comme décrit en détail dans la section 5 du chapitre 4.

Pour l'assemblage des 'Shots' en *Parallel Shots* nous avons choisi – parmi les méthodes présentées dans notre analyse du marché – quatre méthodes d'analogie. D'une part, nous avons examiné deux méthodes d'analyse simple couleur: le détecteur

HSV et le détecteur HY. D'autre part, deux méthodes de type *Feature Points* - le détecteur ScaFT et le détecteur SIFT. Nos test montrent le potentiel des méthodes basées sur le *Feature Point*, mais en réalité l'application directe de ces méthodes basées sur les couleurs simples donnent des résultats meilleurs. Entre les deux méthodes simples basées sur la couleur il n'existe pas de différence significative, c'est pourquoi nous avons choisi la méthode plus efficace en termes de calcul – le détecteur HSV. Avec ce détecteur, nous avons obtenu des résultats de 83% pour le rappel et la précision.

Arrivant à la dernière section de notre quatrième chapitre, nous avons atteint un des buts de notre travail, à savoir examiner la méthode de détection des changements de scène dans le contenu après l'élimination des réclames et passages supplémentaires. Cette partie restante, couvrant à peu prés 70% du contenu initial, nous l'avons classifié en *Parallel Shots*. Pour caractériser un changement en scène nous avons examiné diverses technologies permettant de détecter des discontinuités.

Avec une méthode basée sur la HSV simple, nous avons obtenu des résultats de 85% pour le rappel et 50% pour la précision. Cela n'étant pas suffisant pour notre recherche successive, nous avons entrepris de rechercher d'autres méthodes qui pourraient apporter une amélioration en termes de robustesse. Les informations acquises lors de l'étude des règles de grammaire d'un film nous ont aidés efficacement pour chercher ces méthodes nouvelles. L'application des détecteurs bas-niveau est basée sur la reconnaissance de changement de scène analysant le flux audio n'ont malheureusement pas donnée des résultats acceptables. Une explication pour l'inaptitude de ces détecteurs pourrait être que ces détecteurs étaient construits pour des buts autres que ceux définis dans notre travail. Les attributs spéciaux et différents pour une scène et pour un changement de scène nous ont offert la possibilité de construire un détecteur indépendant de changement de scène *Shot* long. Un changement de scène est très souvent accompagné par des scènes assez longues qui décrivent le commencement et la fin du changement de la scène. La combinaison de notre détecteur *Shot* longue et de notre détecteur de changement de scène HSV nous a apporté un résultat de presque 80% pour les séries et 66% pour les films.

Notre analyse du marché montre, que les résultats obtenus sont suffisants pour notre *lean-backward* méthode d'application *Advanced Content Navigation*, donc apte pour être implémentée comme solution dans les équipements domestiques. En conséquence

de quoi, nous avons intégré la solution basée sur le détecteur de changement de scènes dans notre *Service Unit*.
Ensuite, nous avons trouvé d'autres applications pour notre détecteur de reconnaissance des changements de scènes, par exemple le marché indexant les vidéos professionnelles. Notre méthode peut servir comme analyse primaire donnant aux enregistreurs des données d'archive ou archives professionnelles de vidéo. L'application est aussi possible dans le domaine des consommateurs privés, donnant la possibilité de l'interactivité, où les frontières des scènes est légèrement flottante, permettant une correction manuelle et selon les vœux du consommateur. Une application de cette dernière, appelée *Content Item Boundary* Détection CIBD est décrite dans l'Annexe 10, mais aussi dans [12], [13] et [14].

Dans notre travail, nous avons décrits divers paramètres audio et vidéo de bas et moyen niveau, leur combinaison avec des paramètres sémantiques, menant à la fin à une segmentation cohérente en chapitres. Les résultats obtenus sont suffisants pour diverses applications dans le marché du grand public mais aussi pour le marché professionnel. Néanmoins, les problèmes de reconnaissance du changement des scènes ne sont pas totalement résolus, des recherches futures sont possibles afin d'améliorer encore les résultats. Nous pensons qu'une exploration encore plus profonde des règles de la production d'un film ainsi que la construction et l'inclusion des paramètres de moyen niveau et ceux de niveau sémantique apporterait des résultats encore plus satisfaisants. Parmi les détecteurs spéciaux nous pourrions explorer l'application des paramètres comme par exemple l'identification des objets, identification du narrateur, focus de l'appareil photo, analyse de la profondeur, … Tous ces détecteurs supplémentaires combinés avec la connaissance des règles du film pourraient apporter beaucoup plus des informations du contenu et du concept de la production. Par exemple – avec la reconnaissance du narrateur - on pourrait augmenter drastiquement le niveau de reconnaissance du *Shot-Reverse-Shot*.

Les modules et solutions génériques dans notre concept offrent sans effort et avec une grande efficacité une intégration consécutive par les *Service Units*, permettant de trouver et d'introduire de nouvelles solutions. Notre concept de reconnaissance des changements de scènes peut être appliqué au maniement mécanique. Aussi, en ce cas, notre conseil serait de s'orienter vers les règles de grammaire liée à la production d'un film.

ANNEXES

ANNEX 1: MPEG-2 and Compression Parameters

ANNEX 2: Interlaced / Progressive

ANNEX 3: AV Corpus selection: demographics and statistics of content

ANNEX 4: Scale invariant feature transform SIFT

ANNEX 5: Evaluation of ScaFT and SIFT based parallel shot detector

ANNEX 6: Scene description on AV corpus

ANNEX 7: Formulae for AV Jitter

ANNEX 8: Formulae for Shot Length

ANNEX 9: MPEG-7 descriptors for Service Units

ANNEX 10: Manual post-annotation tool

ANNEX 11: Abbreviations

1. ANNEX: MPEG-2 and Compression Parameters

Moving Picture Experts Group (MPEG) is the name of an international committee responsible for digital audio and video compression (coding) standards, e.g. used for transmission and digital archiving. MPEG is a working group within the *International Standards Organization* (ISO) and the *International Electrotechnical Commission* (ISO) Initially, the MPEG-1 [143] standard, also known as ISO/IEC 11172, has been elaborated in the ’80s and finalized in 1993. It addresses video compression with data rates of up to 1.856 Mbps and three audio compressions standards MPEG-1 Layer-1, -2 and -3. MPEG-1 has been developed to store information on a *Compact Disc – Read Only Memory* (CD-ROM). The advent of the *Digital Versatile Disc* (DVD), an optical disc with ~4 GBs storage capacity, requested for a high-quality video coding standard with data rates between 4-15 Mbps, called MPEG-2 [144], also known as ISO/IEC 13818, established in 1994. The usage of the latter spans from *Digital Video Broadcast* (DVB) for television broadcast, to digital video storage for media archives, to *Video On Demand* (VOD) services. In contrary to MPEG-1, MPEG-2 supports fully interlaced video. In addition, MPEG-1 can cope only with resolutions up to *Standard Interchange Format* (SIF, 352 x 240/288 pixels), but MPEG-2 handles up to *International Radio Consultative Committee* (CCIR)-601 resolution (720 x 480/576 pixels).

Today’s video compression standards, such as MPEG-2, rely on the human eye's inability to resolve high frequency color changes and the fact that there is a lot of redundancy spatial-wise within each frame[17] and time-wise between successive frames. According to this, MPEG-2 defines methods to compress video using the aforementioned characteristics. Figure 120 shows schematically the underlying compression idea, which will be further described in detail.

Details of MPEG-2

MPEG-2 YUV color space

A color model is an abstract mathematical model describing the way colors can be represented. Colors are displayed by means of *Red Green Blue* (RGB), further called the RGB color space. Consequentially, each image pixel consists of 3 color pixels: a red, a green and a blue one (Figure 121, left). In the optimal situation each color pixel is represented by an 8-bit value (resulting in a value range of 0-255) enabling to display up to 1.67 million different colors. The color representation used by e.g. MPEG-2 is YUV, also referenced as YC_bC_r.

[17] one picture out of a motion sequence

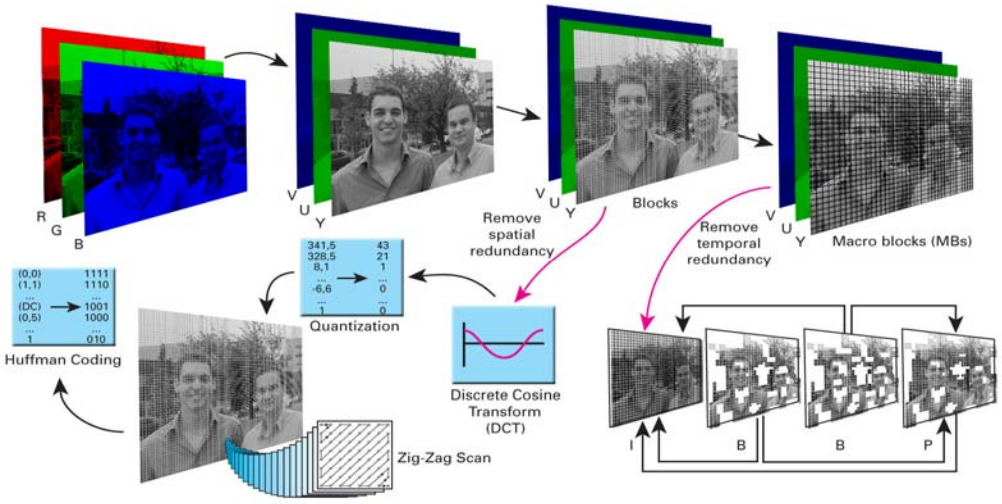


Figure 120: Video compression by using spatial and temporal redundancy[1].

It is composed of an Y-value, which represents the luminance[18], and an U- and V-value, which represent the chrominance[19] (Figure 121, right). The different systems can be easily transformed, e.g. from RGB to YUV, using linear equations (5-1), (5-2) and (5-3). The advantage of YUV is, that it exploits human eye's characteristics, i.e. it perceives brightness changes (Y) more accurate than color changes (U and V). Thus, it is possible to decrease the accuracy of the U- and V-value without loosing much visible information, also called *lossy compression*.

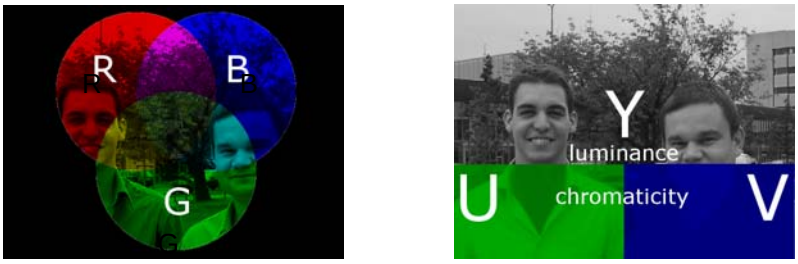


Figure 121: RGB – color space (left), YUV – color space (right) [1].

[18] Also called brightness, which can be derived as output signal from black-and-white cameras.

[19] Also called color difference component.

$$Y = 0{,}299 * R + 0{,}587 * G + 0{,}114 * B \quad (5\text{-}1)$$

$$U = -0{,}146 * R - 0{,}288 * G + 0{,}434 * B = 0{,}493 * (B - Y) \quad (5\text{-}2)$$

$$V = 0{,}617 * R - 0{,}517 * G - 0{,}100 * B = 0{,}877 * (R - Y) \quad (5\text{-}3)$$

MPEG-2 Sub-sampling

Consequently, to exploit the knowledge of the human eye's color resolution deficiency, MPEG-2 uses different down sampling formats, such as *YUV 4:2:0*. In YUV 4:2:0 every individual pixel is represented by an 8 bits Y-value. On contrary, in the U- and V-space only the pixels of odd lines will be sampled (vertical sub-sampling), and furthermore only every second pixel in those odd lines (horizontal sub-sampling). That means odd lines are sampled in 4:2:2 format (4 Y-values, 2 U-values and 2 V-values) and even lines in 4:0:0 (4 Y-values, 0 U-values and 0 V-values). Therefore the representation of a 2 by 2 pixel block (4 pixels) requires 6 bytes (Y: 4 bytes; U: 1 byte; V: 1 byte), which leads to a compression rate of 2:1, as summarized in Figure 122.

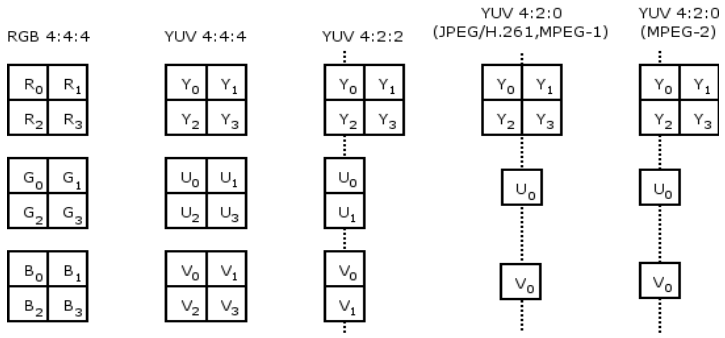


Figure 122: MPEG-2 sub-sampling (U = C_b; V= C_r)

Blocks and MacroBlocks (MB) in MPEG-2

In MPEG-2, as in many other compression formats, the individual frames are split into blocks of 8x8 pixels, further referenced as *Blocks (see* Figure 124*)*. Four neighboring blocks are clustered into one *MacroBlock* (MB), consisting of 16x16 pixels, as visualized in Figure 123. Subsequently, MBs are sub-sampled, as explained above. For example an YUV 4:2:0 (see Figure 122) MB consists of four Y-blocks, 1 U(C_b)-block and one V(C_r)-block (Figure 123).

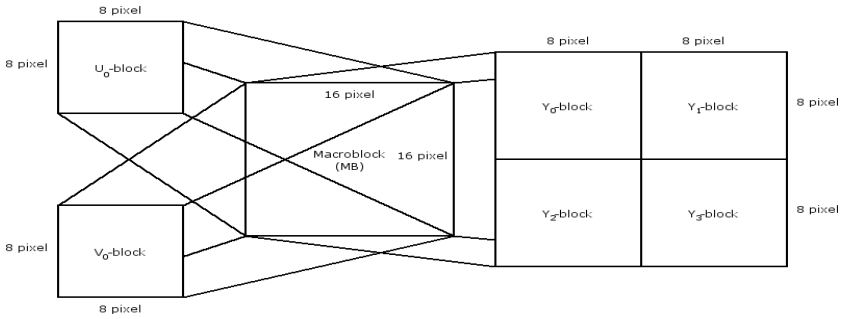


Figure 123: YUV 4:2:0 MacroBlock

MPEG-2 Spatial Redundancy[20] (Intra-frame coding)

Discrete Cosine Transformation

In the next step, *Discrete Cosine Transformation* (DCT, see Figure 124) is used to convert the spatial information within blocks into the frequency domain by means of equation (5-4) and (5-5).

$$F(u,v) = \tfrac{1}{4} \cdot C_u \cdot C_v \cdot \sum_{x=0}^{7} \sum_{y=0}^{7} f(x,y) \cdot \cos\left(\frac{\pi \cdot u(2x+1)}{16}\right) \cdot \cos\left(\frac{\pi \cdot v(2x+1)}{16}\right) \qquad (5\text{-}4)$$

$$C_u, C_v = \tfrac{1}{\sqrt{2}} \quad \text{for} \quad u = 0, v = 0 \quad \text{and} \quad C_u, C_v = 1 \quad \text{otherwise} \qquad (5\text{-}5)$$

with: $F_{(u,v)}$ representing DCT coefficient in row u and column v of the DCT matrix and $f_{(x,y)}$ standing for the intensity of the pixel in row x and column y

After the DCT, the top left value of the block, in Figure 124 sketched as position (00), represents the DC level, the average brightness or chrominance, respectively, of the block. Furthermore, horizontal successive values, e.g. position (01) to (07), represent the strength of horizontal-wise increasing frequency. That means, that the (07)-value represents high frequency horizontal information. Similarly, the (70)-value represents high frequency vertical information. High frequencies appear in regions with a lot of details, texture or edges. Intuitively, the DC value and its close AC neighbors include most information (energy) of the image, resulting in low information in higher frequency values.

Quantization

The next the lossy step is the quantization, as sketched in Figure 124 and Figure 125. The technique was originally developed for *Pulse Code Modulation* (PCM) coding, for

[20] *Spatial redundancy* refers to the correlation between neighbouring pixels in, for example, a flat uniform background.

which it was necessary to map continuous analog values to discrete levels. As mentioned above, the human eye does not have the same sensitivity to all frequencies. Therefore, a coarse quantization of high frequency DCT coefficients is less annoying to the viewer than the same quantization applied to low frequencies. Hence, to obtain a minimal perceptual distortion, each DCT coefficient should be individually weighted. In MPEG-2 this is achieved by the use of a weighting matrix, further referenced as *Quantization Matrix*, which is part of the MPEG-2 data. In addition to the Quantization Matrix, an uniform quantization factor can be applied to the entire image. This quantization factor, further referred to as *Quantization Scale*, will appear to be the key to the rate control algorithm. As a result of this, it is typically the case that many of the higher frequency DCT components are rounded to zero or at least they are clipped to very small positive or negative numbers.

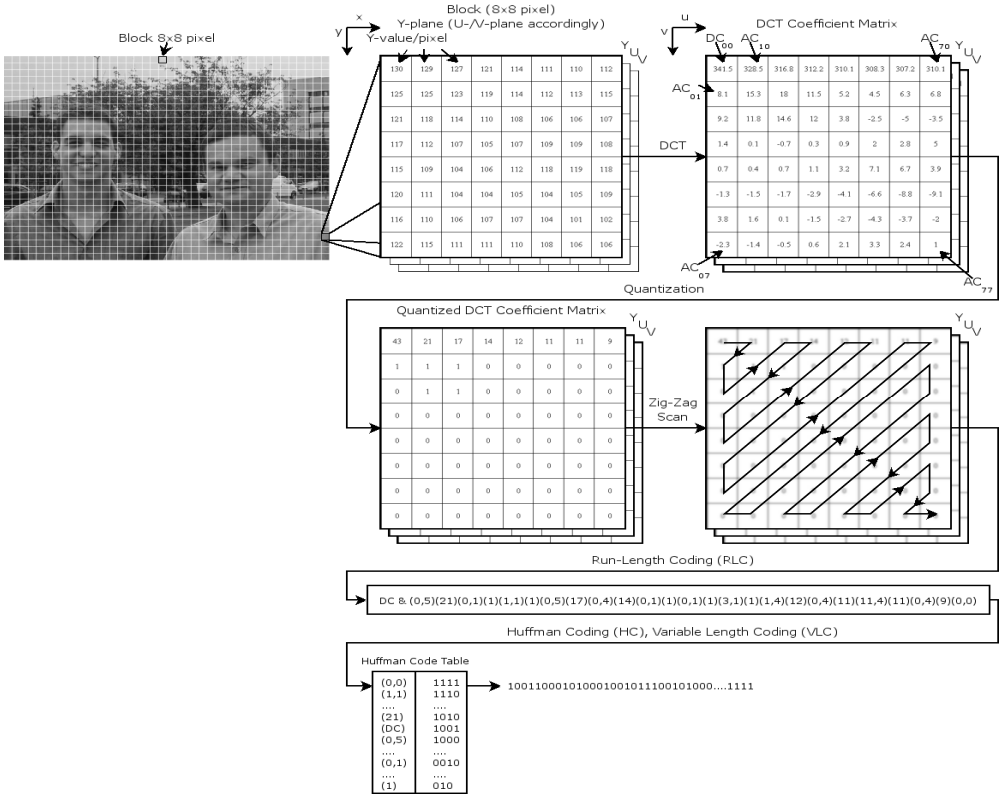


Figure 124: Intra-frame coding (DCT→Quantization →Zig-Zag Scan →RLC → Huffman Coding) [1.]

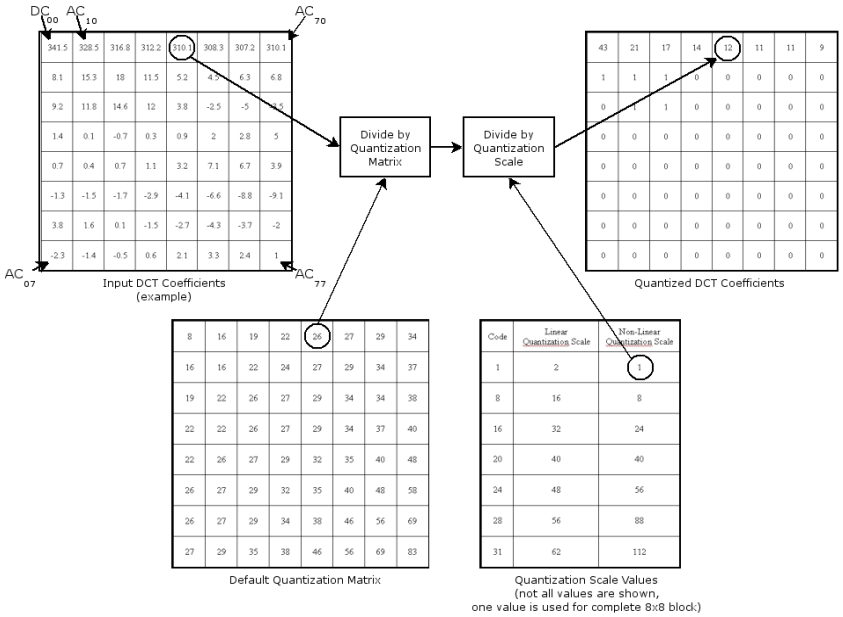


Figure 125: Quantization

Entropy coding

Entropy coding is a form of lossless data compression used in MPEG-2. Basically, it clusters DCT components with similar frequencies together using a *Zig-Zag*[21] Scan, as shown in Figure 126. Successively, *Run-Length Coding* (RLC) and finally *Huffman Coding* are applied, as can be seen in Figure 124 and further described next.

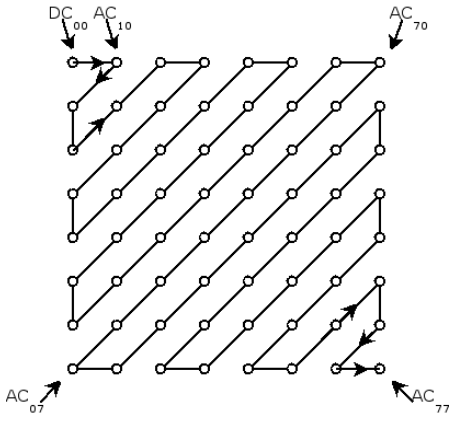


Figure 126: Zig-Zag Scanning

Run Length Coding (RLC)

Run-Length Coding, a simple data compression technique, exploits the redundancy of same consecutive symbols. To avoid saving each symbol individually, the count of symbols will be coded instead, as may be seen in Figure 127.

[21] Clustering of DCT coefficients to encourage runs of 0s.

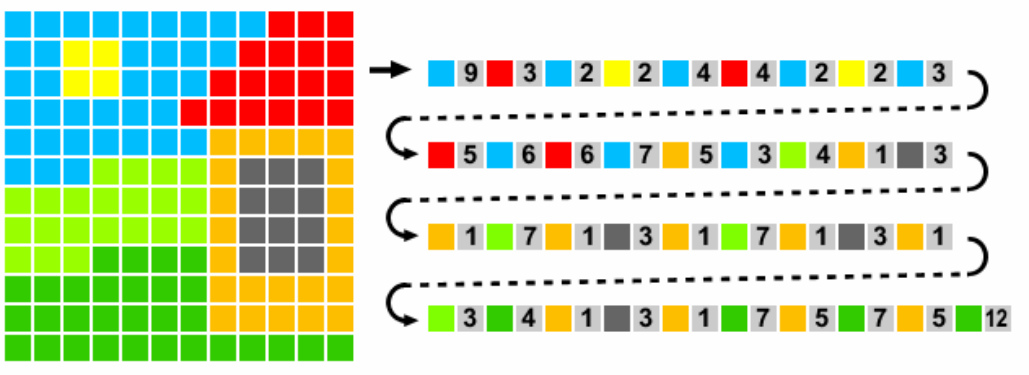


Figure 127: Example for Run-Length Coding (schematically)

Furthermore, MPEG-2 uses the *End-Of-Block* (EOB) symbol, which, in case the last non-zero symbol of a block is reached, will be attached. This helps to further reduce the data, because usually the higher frequencies are quantized to zero.

Huffman Coding

Huffman Coding (HC) is a form of variable-length information encoding, mininizing the number of necessary encoding bits, also called entropy. In an information stream the data for a given variable may be given as *N* bits. HC exploits the statistical nature, meaning that not all 2^N bit combinations are used or at least not with an equal probability. This means, that HC orders all symbols according to their decreasing occurance probability, as seen in Figure 124. Succesively, it assigns the *0* bit to the symbol of highest probability and the *1* bit to what is left. Iterateviley, HC proceeds the same way for the second-highest probability value, which get the 10 code assigned, and it further iterates.

MPEG-2 Temporal Redundancy

I-, P- and B-frames

Naturally, temporal redundancy occurs in time-wise consecutive images e.g. during uninterrupted recording events, as e.g. in broadcast material. The recording event is limited by the start- and stop-recording, respectively, instances, which are represented by shot transitions, also called *Shot Boundaries* (SBs) [103]. To exploits the temporal redundancy, MPEG-2 defines three types of frames, as sketched in Figure 128:

- Intra coded intra frames, also referenced as I-Frames,
- Inter coded predicted frames, also called P-Frames, and

- Inter coded bi-directional frames, usually called B-Frames.

To be more specific, intra coded I-frames contain all information required for decoding, which means that they can be reconstructed without any reference to other frames. P-frames are forward predicted based on the predecessor I- or P-frame, as sketched in Figure 129. Therefore, it is impossible to reconstruct a P-frame without the data of another I- or P frame, unless it is entirely intra coded, which could be the case after a shot transition. Finally, B-frames are both, forward- and/or backward predicted, as shown in Figure 130. In average, B-frames require the data from both, the predecessor and the successor I- or P-frame. This implies, that normally two other frames are necessary to decode B-frames.

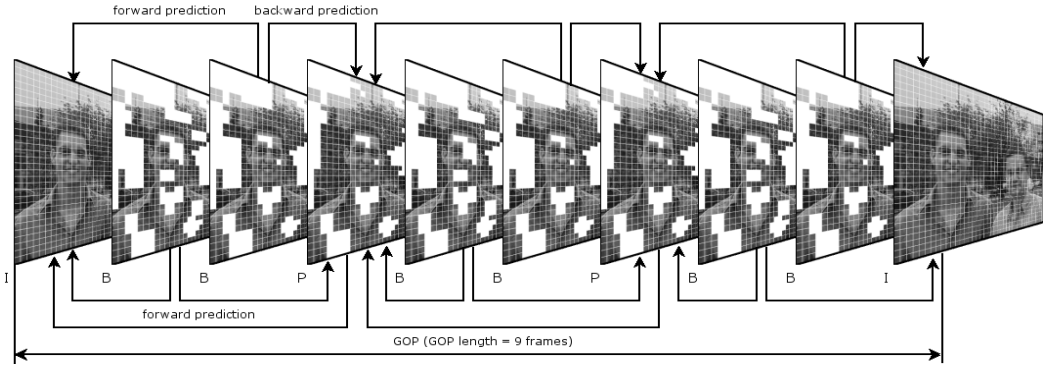


Figure 128: I-, P- and B-frames used in MPEG-2[1].

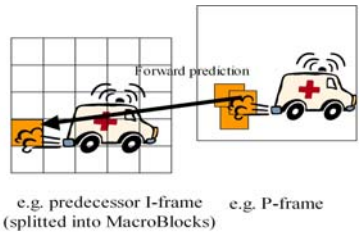


Figure 129. Forward Prediction

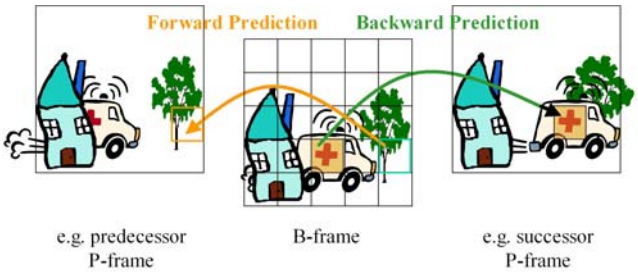


Figure 130. For- and Backward PredictionI

n addition, the time-wise successive frames are clustered into so called *Group Of Pictures* (GOPs) with arbitrary length. GOPs are headed by an I-frame, as shown in Figure 128.

Motion compensation

I-frames are reference frames exclusively compressed by means of present spatial redundancy. As a result of this, I-frames contain all required information to decode them autonomously. On contrary, P- and B-frames are compressed by exploiting not only spatial-, but also temporal redundancies. To identify the presents of the latter, the *Mean*

Absolute Difference (MAD) value is used (labeled with letter 'b' in Figure 131), which is an output of the video compressor *Motion Estimator* (ME) as described in [106].

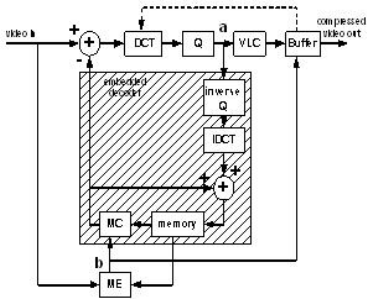


Figure 131. Schematic of an MPEG-2 video compressor (partially)

The MAD is a matching criterion used to quantify the similarity between the target MB and best matching candidate MBs in the reference frame exploiting temporal redundancies. To be more specific, corresponding pixels of two MBs, one from source frame *n* and one from reference frame *m*, are compared and their differences are summed up, as described by equation (5-6).

$$MAD(x,y,dx,dy) = \frac{1}{256}\sum_{i=0}^{15}\sum_{j=0}^{15}\left|V_n(x+i,y+j) - V_m((x+dx)+i,(y+dy)+j)\right| \quad (5\text{-}6)$$

where *MAD(x,y,dx,dy)* represents the MAD between a 16*16 array of pixels (pels) of intensities $V_n(x+i,y+j)$, at MB position (x,y) in the source frame *n*, and a corresponding 16*16 array of pixels of intensity $V_m(x+dx+i,y+dy+j)$, in reference frame *m*, with *dx* and *dy* representing the shift along the x and y coordinates, also called motion vector.

Using the MAD value the compressor can select one of the following three approaches to minimize the bit rate by retrieving correlations between time-wise successive frames.
The first one, the MB at the same position in the successor frame hasn't changed, quantified by an MAD value falling short a certain threshold. This leads to the decision that no coding will be executed on this MB. The information will be transmitted that both MBs are the same.
The second one, the MAD of the MB at the same location exceeds a defined threshold. In this case a search is initiated to search for the best matching MB in a defined search area (normally 48 by 48 pixels) clustered around the current MB position. The horizontal and vertical position change between the best matching- and the current MB represents the motion vector, which will be transmitted as well.
Finally the third one, in this case even the MAD value of the best matching MB exceeds a defined threshold. In this case the complete intra-coding process will be applied to the current MB.

Applied compression parameters

The MPEG-2 compression parameters applied in sub-section 4.1.1 are summarized in Table 57.

Table 57. Compression format settings.

		PAL			NTSC		
		D1	½ D1	SIF	D1	½ D1	SIF
Width	[pixels]	720	352	352	720	352	352
Height	[pixels]	576	576	288	480	480	240
Active MBs / slice		720/16=45	352/16=22	352/16=22	720/16=45	352/16=22	352/16=22
Active slice / frame		576/16=36	576/16=36	288/16=18	480/16=30	480/16=30	240/16=15
Letterbox upper	LB1	1	1	1	1	1	1
	LB2	4	4	2	4	4	2
	LBs	4	4	2	4	4	2
Letterbox lower	LB1	33	33	17	33	33	17
	LB2	36	36	18	36	36	18
	LBs	4	4	2	4	4	2
Upper	[slices]	26	26	13	21	21	10
LowMiddle	[slices]	10	10	5	9	9	5
Centre	[slices]	16	16	8	12	12	6
Subtitle	[sliceNr]	27-32	27-32	14-16	22-26	22-26	11-13

2. ANNEX: Interlaced / Progressive

Video frames of video broadcast are composed of a number of *Red Green Blue* (RGB) pixels, resulting in a full frame as shown in Figure 132 (left). The number of pixels, in vertical and horizontal direction, depends on the broadcast standard and the resolution used for the transmission of the video, as summarized in Table 58. The rendering of the pixels starts at the top left and will successively, pixel by pixel, continue from the left to the right and line-by-line, i.e. row-wise through the frame. This continuous way of capturing or rendering of pixels is called *Progressive Scan*. This scan method is applied in modern studio cameras and modern rendering devices, such as *Liquid Crystal Displays* (LCD), which are capable to render the video frames fast enough in progressive mode. The term progressive (p) is used, because each frame is captured and displayed, respectively, from the first scan line to the last scan line without any discontinuation. On contrary, technical constraints of the *Cathode Ray Tube* (CRT) television sets of the early days of television required the split of each frame into two fields (half-frames), the upper field, see Figure 132 (center), and the lower field, see Figure 132 (right). The first half-frame, the upper field, contains all odd lines of the original frame, i.e. line 1, 3, 5, etcetera, and the second half-frame, the lower field, all even lines of the frame, i.e. line 2, 4, 6, etcetera. This capturing and rendering method is called interlaced (i) mode, applied in low-cost studio cameras and camcorders, but also in legacy rendering devices, such as CRT television sets.

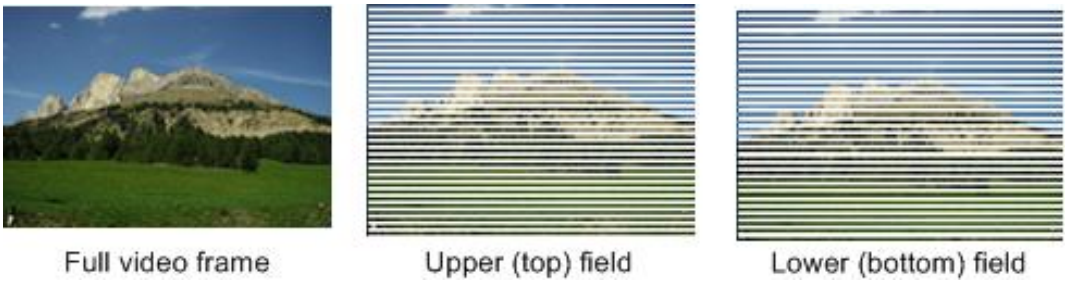


Figure 132. Interlaced video with top and bottom field[1].

Table 58. Video Resolutions [Systeme Electronique Couleur Avec Memoire (SECAM), Standard Interchange Format (SIF), International Radio Consultative Committee (CCIR)].

Resolution	PAL / SECAM		NTSC	
	Number of pixels in horizontal direction	Number of pixels in vertical direction	Number of pixels in horizontal direction	Number of pixels in vertical direction
SIF	352	288	352	240
CCIR 601	720	576	720	480

Another reason for this split of frames was the halved bandwidth required, a bandwidth problem of the early days of television transmission.

The frame capturing and display frequency is dependent of the broadcast standard applied. In Western Europe, for example, *Phase Alternating Line* (PAL) is the standard used with 50 fields per second resulting in frequency of 50Hz and referenced as 50i. With the afterglow of the pixel points in e.g. a CRT, the viewer gets the impression that real 50 'full' frames per second are rendered. The standard frame resolution used for PAL transmission is based on the CCIR 601 standard, see Table 58, with 720-by-576 pixels per frame and 720-by-288 pixels per field, respectively. In other regions such as in Northern America the *National Television System Committee* (NTSC) is applied with 59.94 fields per second (29.97 fps), respectively, resulting in 59.94 Hz (59.94 Hz are round up in literature to 60 Hz and referenced as 60i).

Scan Rate Conversion: telecine

Unfortunately, any film content captured with a progressive-film-format camera, as e.g. used in professional movie studios, is recorded on celluloid with 24 fps (24p, e.g. in US) and 25 fps (25p, e.g. in Europe), which is also referenced as film mode. In order to display film on TV sets, the 24p/25p format has to be converted to 50i/60i format. The process is called telecine or pull-down.

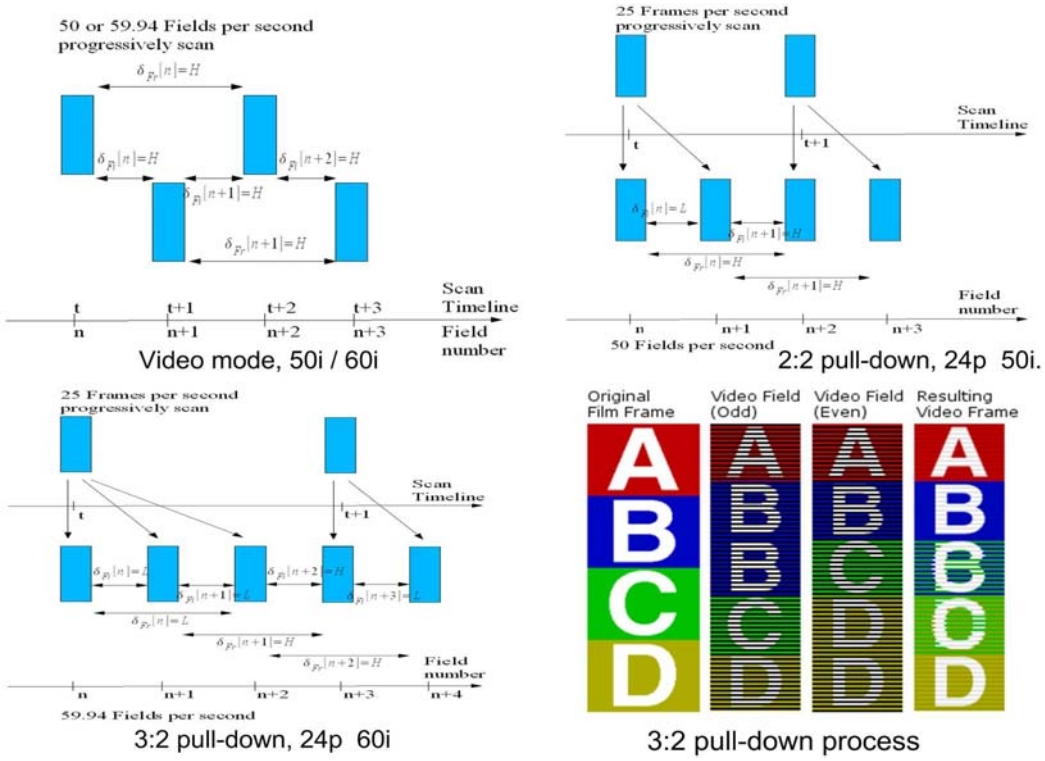


Figure 133. 2:2 and 3:2 pull down mode.

3:2 pull-down for film-to-NTSC (24p-to-60i) conversion

To convert a 24p content to 29,97 fps based NTSC requires more effort to accurately render the film's motion, called *3:2 pull-down*. 3:2 pull-down is accomplished in two steps. Firstly, the film's motion has to be reduced, or pulled-down, by 0.1%, which is unnoticeable to the viewer, and reduces the frequency to 23.976 fps. Secondly, at 23.976 fps, there are 4 frames of film for every 5 frames of NTSC video (23.976/29.97 = 4/5). These four frames are now "stretched" into five frames exploiting the interlaced nature of NTSC video. Each NTSC frame is the result of two fields of the interlaced video, an upper field (odd lines) and a lower field (even lines). Therefore, ten fields are required to assemble 5 NTSC frames. To meet this requirement, alternately one film frame is placed across 3 fields, the next film frame across 2 fields, then again across 3 fields and so on (3:2:3:2…), as can be seen in Figure 133 (right bottom). The 3-to-2 cycle repeats itself completely after four film frames have been exposed[22].

2:2 pull-down for film-to-PAL (25p-to-50i) conversion

The conversion from 24 fps film mode to 25 fps based PAL is simply done by speeding up the 24 fps content by 4% to reach 25p. This causes a noticeable increase in audio pitch by about one semitone, which is corrected by a pitch shifter. In a second step each frame is split into an odd and an even field resulting in 50 fields-per-second (50i) and therefore it's called 2:2 pull-down.

[22] Note that the pattern in this example is 2-3 and 2-3. The name 3:2 pull-down is an archaic reference to the pattern, which was used by older telecine equipment. The modern telecine uses a 2-3 technique.

3. ANNEX: AV Corpus selection: demographics and statistics

Table 59: Demographic data for China 2002 – 2025 – 2050 from [105]

Ages	Men	Women	Ages	Men	Women	Ages	Men	Women
0- 4	50 110 996	45 771 602	0- 4	42 516 389	40 190 790	0- 4	36 040 448	34 040 885
5- 9	54 198 573	49 416 465	5- 9	46 501 006	43 808 040	5- 9	37 041 087	34 998 978
10- 14	62 816 940	57 148 486	10- 14	48 150 313	45 030 010	10- 14	36 799 988	34 788 343
15- 19	51 385 852	47 672 902	15- 19	44 291 013	41 173 750	15- 19	36 657 224	34 741 013
20- 24	49 357 387	46 532 006	20- 24	41 888 507	38 785 016	20- 24	37 859 350	36 074 383
25- 29	61 095 341	57 718 981	25- 29	48 611 194	44 945 031	25- 29	41 438 336	39 688 139
30- 34	64 434 769	60 704 306	30- 34	52 415 563	48 504 489	30- 34	45 126 325	43 198 110
35- 39	52 752 092	50 054 406	35- 39	60 500 300	55 931 796	35- 39	46 491 343	44 286 491
40- 44	42 779 757	39 491 531	40- 44	49 046 277	46 386 217	40- 44	42 529 256	40 352 627
45- 49	43 199 071	40 832 053	45- 49	46 810 564	45 045 132	45- 49	40 085 733	37 888 627
50- 54	31 731 338	29 322 907	50- 54	57 226 778	55 351 648	50- 54	46 189 985	43 646 144
55- 59	23 850 562	21 972 889	55- 59	58 617 247	57 120 078	55- 59	48 722 464	46 458 109
60- 64	21 071 214	19 655 095	60- 64	45 574 142	45 637 400	60- 64	53 948 455	52 315 254
65- 69	17 488 055	17 145 676	65- 69	33 407 460	33 830 467	65- 69	40 738 282	41 668 661
70- 74	12 130 589	12 954 251	70- 74	29 147 072	31 953 987	70- 74	34 595 250	37 822 318
75- 79	7 054 614	8 844 624	75- 79	16 664 374	19 494 948	75- 79	34 829 009	41 402 075
80+	4 317 168	7 461 803	80+	13 570 567	20 319 584	80+	46 364 123	68 803 815

Table 60: Demographic data for US 2002 – 2025 – 2050 from [105]

Ages	Men	Women	Ages	Men	Women	Ages	Men	Women
0- 4	9 831 175	9 386 999	0- 4	12 015 262	11 503 133	0- 4	14 348 291	13 731 791
5- 9	10 488 829	9 994 277	5- 9	11 831 125	11 331 891	5- 9	14 059 510	13 461 160
10- 14	10 560 818	10 047 597	10- 14	11 692 274	11 195 478	10- 14	13 782 484	13 191 510
15- 19	10 412 689	9 837 270	15- 19	11 496 334	10 972 391	15- 19	13 602 067	12 981 326
20- 24	9 821 860	9 363 203	20- 24	11 296 473	10 828 512	20- 24	13 465 834	12 904 593
25- 29	9 785 399	9 531 418	25- 29	10 882 110	10 559 020	25- 29	13 375 482	13 002 541
30- 34	10 372 884	10 214 189	30- 34	11 646 783	11 346 511	30- 34	13 407 428	13 107 356
35- 39	11 304 995	11 343 359	35- 39	11 654 430	11 425 246	35- 39	13 277 400	13 056 279
40- 44	11 179 973	11 355 395	40- 44	11 232 276	11 086 703	40- 44	12 806 833	12 703 154
45- 49	9 959 477	10 271 081	45- 49	10 327 250	10 354 883	45- 49	12 234 140	12 235 962
50- 54	8 706 996	9 083 620	50- 54	9 914 318	10 130 050	50- 54	11 418 281	11 500 158
55- 59	6 553 207	7 005 944	55- 59	9 945 218	10 346 340	55- 59	11 568 562	11 768 619
60- 64	5 165 703	5 699 027	60- 64	10 184 920	10 943 536	60- 64	10 997 678	11 386 511
65- 69	4 402 844	5 131 111	65- 69	9 283 604	10 363 146	65- 69	9 911 357	10 532 466
70- 74	3 904 321	4 945 625	70- 74	7 346 016	8 694 809	70- 74	8 273 629	9 224 985
75- 79	3 051 227	4 374 151	75- 79	5 376 751	6 890 873	75- 79	6 853 651	8 213 190
80+	3 093 305	6 158 663	80+	5 793 098	9 775 435	80+	13 289 512	20 406 847

Table 61: Demographic data: Europe 2002 – 2025 – 2050 from [105].

Ages	Men	Women	Ages	Men	Women	Ages	Men	Women
0- 4	10 542 939	10 007 895	0- 4	9 165 564	8 698 547	0- 4	8 427 193	7 994 322
5- 9	11 124 837	10 568 773	5- 9	9 435 834	8 959 945	5- 9	8 608 629	8 170 232
10- 14	11 602 210	11 032 061	10- 14	9 713 911	9 226 075	10- 14	8 811 603	8 365 403
15- 19	11 958 312	11 390 902	15- 19	10 135 301	9 639 872	15- 19	9 086 621	8 639 665
20- 24	12 694 112	12 164 576	20- 24	10 817 334	10 328 002	20- 24	9 471 820	9 039 278
25- 29	14 237 830	13 677 340	25- 29	11 325 887	10 818 497	25- 29	9 930 066	9 469 727
30- 34	15 650 195	15 076 087	30- 34	12 042 438	11 468 368	30- 34	10 343 481	9 820 621
35- 39	15 699 130	15 234 251	35- 39	12 557 586	11 960 998	35- 39	10 680 311	10 124 746
40- 44	14 190 246	13 966 199	40- 44	12 809 820	12 233 392	40- 44	11 029 466	10 469 461
45- 49	13 102 197	13 095 660	45- 49	13 223 926	12 728 194	45- 49	11 438 803	10 908 202
50- 54	12 716 783	12 796 671	50- 54	14 224 273	13 859 061	50- 54	11 524 561	11 082 898
55- 59	10 729 461	11 000 789	55- 59	14 948 882	14 861 587	55- 59	11 731 847	11 433 125
60- 64	10 134 619	10 836 527	60- 64	14 214 309	14 593 531	60- 64	11 637 084	11 592 265
65- 69	8 711 064	9 944 514	65- 69	11 911 258	12 873 596	65- 69	11 102 560	11 430 231
70- 74	7 274 005	9 340 219	70- 74	9 797 952	11 361 778	70- 74	10 404 591	11 278 037
75- 79	5 283 416	8 278 284	75- 79	7 950 245	10 043 844	75- 79	9 601 648	11 269 157
80+	4 58 2932	9 913 597	80+	9 612 485	16 794 675	80+	16 567 689	26 505 737

Statistics of recorded benchmark AV corpus

The recorded *Benchmark AV Corpus* consists of selected content items summarized in Table 62, which will be further used for the evaluation of developed feature extraction algorithms:

Table 62. AV Corpus data files and AV Corpus subset for parallel shot detection and scene boundary detection.

Filename	Genre	Corpus for 1st SBD	# GT SB	Corpus fro 2nd SBD, PSD, ScBD	AV subset content: Items further referenced as
cartoons_eu_com_15min_nl_dig	Cartoon	X	211		
mag_eu_com_30min_ge_ana	Magazine	X	298		
mag_eu_com_30min_nl_ana	Magazine	X	294		
mag_eu_pub_30min_gb_ana	Magazine	X	323		
mag_eu_pub_30min_nl_ana	Magazine	X	186		
movie_eu_pub_100min_fr_ana	Movie	X	1021		
movie_eu_pub_50min_ge1_ana	Movie			X	movie_ge1
movie_eu_pub_50min_ge2_ana	Movie			X	movie_ge2
movie_eu_com_100min_nl_dig	Movie	X	1256	X	movie_nl
movie_us_pub_150min_us_dig	Movie			X	movie_us_dig
movie_us_com_150min_us_ana	Movie			X	movie_us_ana
serie_eu_com_30min_nl1_ana	Series	X	497	X	serie_nl1
serie_eu_com_30min_nl2_ana	Series	X	389	X	serie_nl2
serie_eu_com_30min_ge1_ana	Series	X	433	X	serie_ge1
serie_eu_com_30min_ge2_ana	Series	X	514	X	serie_ge2
serie_eu_pub_30min_gb_ana	Series	X	554	X	serie_gb
show_eu_com_30min_nl_ana	Show	X	359		
show_eu_com_30min_ge_ana	Show	X	1025		
sport_eu_pub_30min_nl_ana	Sport	X	164		

Table 63. Data used for audiovisual jitter analysis.

AV sub-corpus for 2nd SBD, PSD, ScBD	Total # SB (genre only)	# Shots inside PS / # PS	# of potential ScB	# ScB	# ScB (adds, EPG)
movie_ge1	900	555 / 29	374	25	0
movie_ge2	442	**242** / 21	221	30	4 (C1/2/3/4)
movie_ nl	1400	523 / 33	910	35	0
movie_ us_dig	1190	1041 / 44	193	26	0
movie_ us_ana	1260	736 / 31	555	54	0
serie_ nl1	231	151 / 15	95	7	3 (C1/2/3)
serie_ nl2	220	166 / 15	69	15	3 (E1, C2/3)
serie_ ge1	181	128 / 9	62	11	2 (C1/2)
serie_ ge2	490	307 / 23	206	17	4 (E1/2/3/4)
serie_ gb	481	360 / 21	142	19	2 (E1/2)

Table 64. Shot boundary ground-truth for audiovisual corpus sub-set.

AV sub-corpus for 2nd SBD, PSD, ScBD	# GT SBD 1st 50%	# GT SBD 100%	# Shots inside PS / # PS 1st 50%	# Shots inside PS / # PS 100%	# of potential ScB 100%	# ScB	# ScB (adds, EPG)
movie_ ge1	427	892	235 / 20	535 / 47	404		0
movie_ ge2	189	314	157 / 14	247 / 25	92		4 (C1/2/3/4)
movie_ nl	694	1352	291 / 22	487 / 42	907		0
movie_ us_dig	531	1208	456 / 26	1049 / 61	220		0
movie_ us_ana	496	1176	353 / 32	617 / 56	615		0
Total movies	2337	4942			2238		
serie_ nl1	119	227	83 / 9	154 / 18	91		3 (C1/2/3)
serie_ nl2	109	212	99 / 9	181 / 19	50		3 (E1, C2/3)
serie_ ge1	87	175	70 / 7	141 / 14	48		2 (C1/2)
serie_ ge2	295	495	171 / 15	296 / 29	228		4 (E1/2/3/4)
serie_ gb	272	482	207 / 17	354 / 39	167		2 (E1/2)
Total series	882	1591			584		

4. ANNEX: Scale invariant feature transform SIFT

In this Annex the *scale invariant feature transform* SIFT, which the author applied for key frame similarity analysis, is explained in more detail. In 1981 Moravec [145] introduced the theory of (distinct) interest points, a theory improved by Harris and Stephens in 1988 [146] resulting in 'improved corner features'. In parallel Witkin developed in 1983 [147] the theory of scale spaces, which Koenderink extended in 1984 [148] to continuous scale spaces and finally Lindenberg to normalized scale spaces in 1994 [149]. Lowe then combined the theory of normalized scale spaces, improved corner features and Torr's 'robust geometric clustering' into the theory of scale invariant feature transform.

Lowe's SIFT used a scale-space representation of a frame, e.g. F_N, and selected local extrema in this scale space as feature points. The advantages of the latter were their invariance to scaling. Invariance to rotation was successfully added.

For the selection of SIFT points the scale-space representation of Y_{FN} , i.e. the luminance plane of YUV of frame F_N , was calculated,

$$L_{F_N}(x, y, \sigma) = G(x, y, \sigma) * Y_{F_N}(x, y) \quad (5\text{-}7)$$

by convolving Y_{FN} with a family of Gaussians of different variance, where

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (5\text{-}8),$$

the Green function, constituted of a two-dimensional Gaussian of variance σ. Hereafter, the corresponding *differential* scale-space representations, as shown in Figure 134,

$$D_{F_N}(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * Y_{F_N}(x, y) = L_{F_N}(x, y, k\sigma) - L_{F_N}(x, y, \sigma) \quad (5\text{-}9)$$

were calculated, as presented in [135], by using the difference of two scales separated by a constant multiplicative factor $k=2^{1/2}$. Sub-sampling was repeated until the resolution fell below 50*50 pixels, a heuristically chosen value.

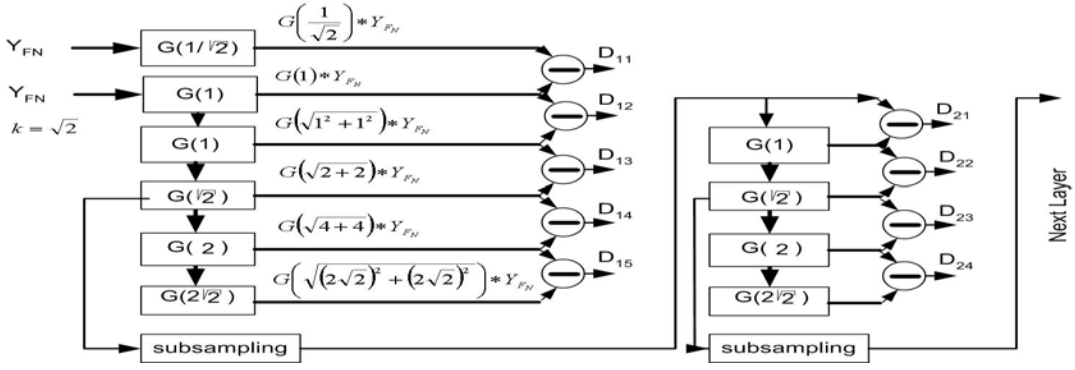


Figure 134. Difference of Gaussians DoG.

The scale space representations allowed retrieving distinctive points at multiple scale levels. To do so, local extrema within $D_{FN}(x,y,\sigma)$ were identified applying

$$\left| D_{F_N}(x, y, \sigma) \right| > a \qquad (5\text{-}10),$$

using an intensity threshold of $\alpha=5$ (derived from [135]). In addition, for each selected SIFT feature point an orientation was assigned that reflected the dominant direction of the local gradient, a kind of SIFT feature point signature, used to improve the tracking robustness. To obtain the orientation at first, the gradient magnitude $m_{FN}(x,y)$ and gradient orientation $\theta_{FN}(x,y)$ were calculated for each pixel in F_N with

$$m_{F_N}(x,y) = \sqrt{\left(L_{F_N}(x+1,y) - L_{F_N}(x-1,y)\right)^2 + \left(L_{F_N}(x,y+1) - L_{F_N}(x,y-1)\right)^2} \qquad (5\text{-}11),$$

$$\theta_{F_N}(x,y) = \arctan \frac{L_{F_N}(x,y+1) - L_{F_N}(x,y-1)}{L_{F_N}(x+1,y) - L_{F_N}(x-1,y)} \qquad (5\text{-}12).$$

In a second step, an orientation histogram was calculated around each feature point consisting of gradient orientations $\theta_{FN}(x,y)$ of all points within a specified window W_{FP} (here a Gaussian region of 4σ was chosen). The orientation histogram consisted of 36-bins covering a 360˚ orientation range. Each sample, which was added to the histogram was weighted by its $m_{FN}(x,y)$. Successively, the dominant orientation of the histogram, i.e. $\theta_{FN\_dominant}(x,y)$, was assigned to the SIFT feature point, which resulted in feature point descriptor vectors including $\theta_{FN\_dominant}(x,y)$, as shown in Figure 135 for three points in an up- and downscaled image representation. $\theta_{FN\_dominant}(x,y)$, hereafter, was used to normalize the orientation and $m_{FN}(x,y)$ to normalize the feature point representation to a sample array of heuristically chosen 15*15 (Lowe used 16*16 in [135]) pixels by extra- or interpolation. The latter step provided the desired scale invariance.

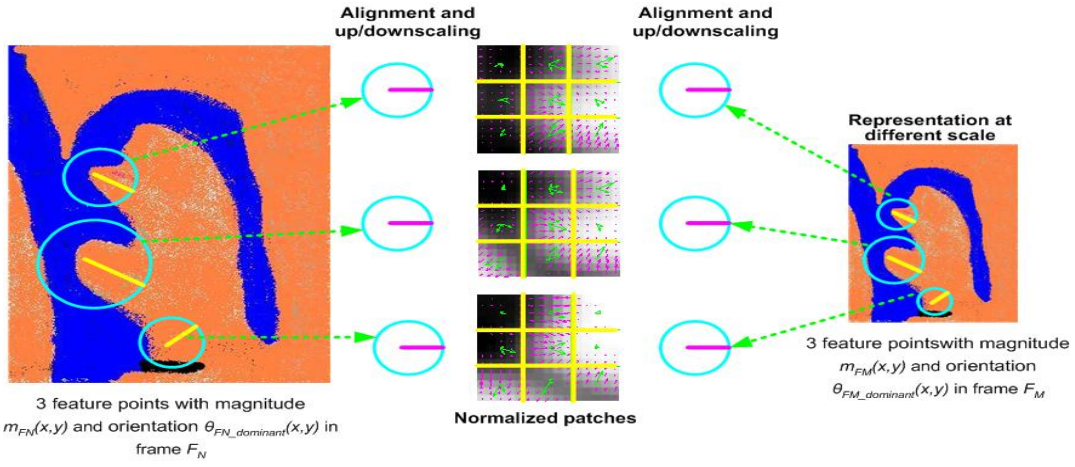


Figure 135. Scale invariant image descriptor matching.

For each pixel inside the normalized patches $m_{FN}(x,y)$ and $\theta_{FN}(x,y)$ were computed and, e.g. taking the example of Figure 136, 4*4 pixel values were accumulated into one orientation histogram with 8 orientation bins with the length of each bin (arrow in Figure 136 right) corresponding to the sum of the corresponding $m_{FN}(x,y)$ values. For the current work the 15*15 sample arrays were accumulated in 3*3 descriptor arrays with 8 orientations each resulting in a vector of *3*3*8=72* vector values *Vv* (Lowe used *4*4*8 Vv,* $Vv \in [0\ldots255]$). Finally, SIFT points of two frames, e.g. F_N and F_M, were indexed as matching if their normalized Euclidian distance fell short a chosen threshold $Th_{SIFTPoints}=0.2$.

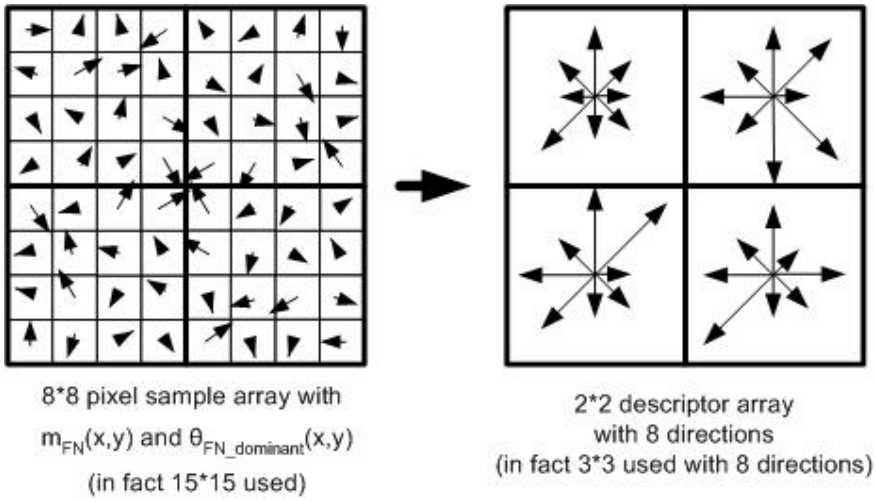


Figure 136. SiFT feature point vector arrays.

5. ANNEX: Evaluation of ScaFT and SIFT based parallel shot detector

Evaluation of ScaFT based parallel shot detector

The evaluation of the ScaFT based key frame pair similarity analysis unveils that the gradient matrix based minimum eigenvalue feature point detection performs reasonable well for sequences with areas containing rigid objects. The latter are often present in the background, as presented in Figure 137 for the correct detections during shot reverse shots and cross-cuttings rigid background areas containing correctly tracked feature points. Unfortunately, recall of the method is limited to about 50%, as summarized in Table 65, based on several shortcomings, i.e. (a) limiting the detection to luminance only lead to missed detection in case of illumination variations (Figure 137 SRS missed in movies), (b) limiting the method to one scale space leading to feature point tracking problems between e.g. zoomed sequences, and (c) applying minimum eigenvalue favoured areas with blocking artefacts (which were present especially in series) to place feature points.

Figure 137. ScaFT result examples for shot reverse shots and cross-cuttings (placeholder in final version).

Table 65. Parallel shot link detection results with ScaFT in series.

Series	‘nl1’	‘nl2’	‘ge1’	‘ge2’	‘gb’	Total
Ground Truth links	227	212	175	495	482	1591
Correct	119	106	78	322	254	877
False	72	33	31	238	59	433
Missed	108	106	97	173	228	714
Recall [%]	52.3	50.0	44.3	65.0	52.6	55.1
Precision [%]	62.2	76.3	71.7	57.4	81.2	67.0

Table 66. Parallel shot link detection results with ScaFT in movies.

Movies	‘nl’	‘ge1’	‘ge2’	‘us_ana’	‘us_dig’	Total
Ground Truth links	890	314	1352	1176	1208	4940
Correct	449	183	692	587	642	2554
False	147	10	145	180	92	573
Missed	441	131	660	589	566	2386
Recall [%]	50.5	58.4	51.2	49.9	53.2	51.7
Precision [%]	75.4	94.7	82.7	76.6	87.5	82.0

Precision, on contrary, varies between 60 and 90% with various reasons why false links are established. Most often textured areas, e.g. hair, foliation, but also blocking artefacts, lead to feature point detection at similar locations, but in completely unrelated key frame pairs, as captured in ‘false’ in Figure 137, an error which can be minimized by using feature point signatures as SIFT does.

Evaluation SIFT based parallel shot detector

Lowe’s scale invariant feature transform method is very well suited to identify robust feature points in textured rigid areas, but especially in shot reverse shot loaded series with unstable non-rigid persons in the foreground, as pictured in Figure 138, the method unveils its limitations.

Table 67. Parallel shot link detection results with SIFT in series.

Series	‘nl1’	‘nl2’	‘ge1’	‘ge2’	‘gb’	Total
Ground Truth links	227	212	175	495	482	1591
Correct	88	53	63	148	149	501
False	44	7	9	72	23	155
Missed	139	159	112	347	333	1090
Recall [%]	38.6	25.0	36.1	29.9	30.9	31.5
Precision [%]	66. 7	87.9	87.5	67.4	86.6	76.4

Table 68. Parallel shot link detection results with SIFT in movies.

Movies	‘nl’	‘ge1’	‘ge2’	‘us_ana’	‘us_dig’	Total
Ground Truth links	890	314	1352	1176	1208	4940
Correct	510	178	723	534	631	2576
False	216	12	218	205	96	747
Missed	380	136	629	642	577	2364
Recall [%]	57.3	56.8	53.5	45.4	52.3	52.1
Precision [%]	70.2	93.8	76.9	72.3	86.8	78.3

Recall, therefore, drops, in comparison to ScaFT, from 50% to 30% in series, but reaches about 50% in movies, as summarized in Table 67 and Table 68. Precision, reaches only 80% for both genres, because SIFT tracks the required number of feature points, i.e. TH_{PS}, from a non-textured to a textured key frame rather easy, as shown for 'false' cases in Figure 138 with pink lines for tracked points. Three of the deficiency of SIFT are that (a) only luminance, but no colour values, (b) no spatial constellation, and (c) not texture dependent thresholds are taken into consideration. Especially the two latter could improve precision drastically.

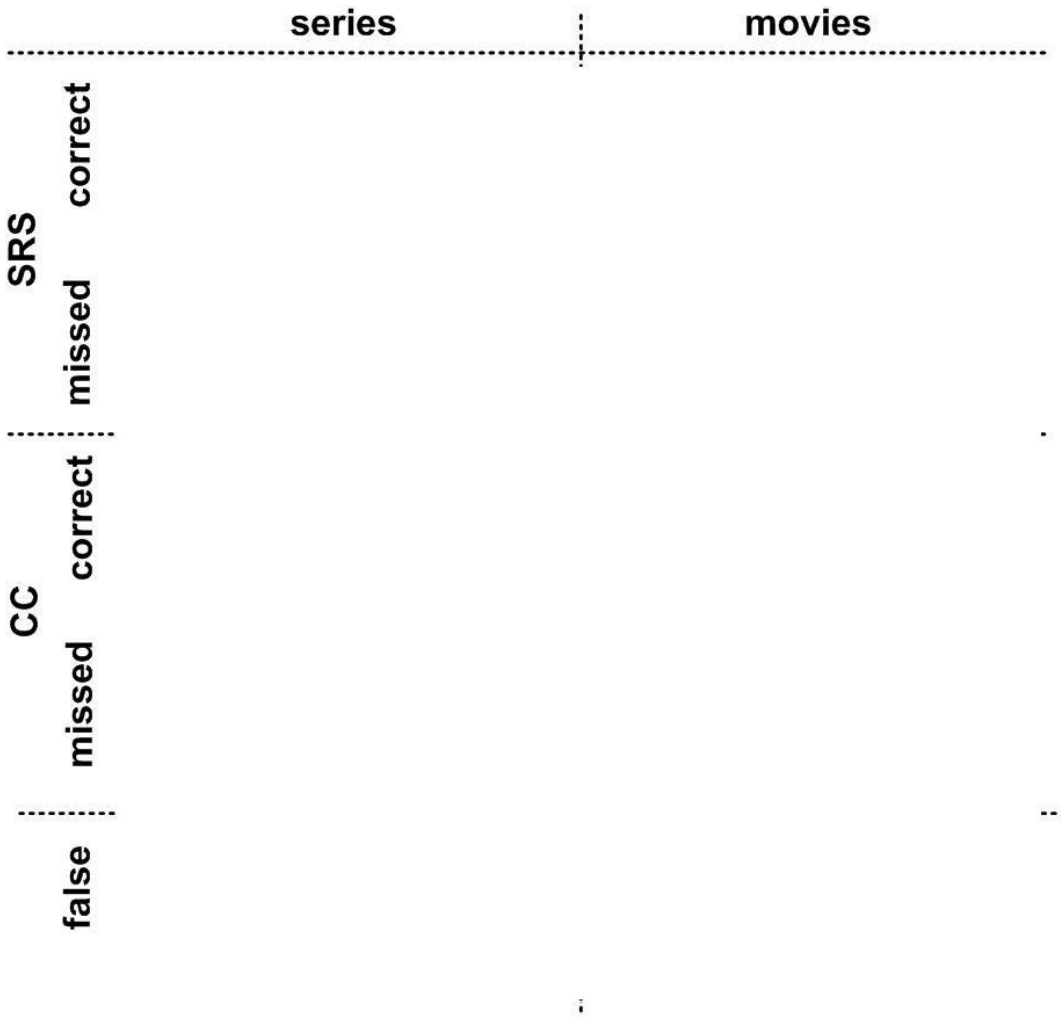


Figure 138. SIFT result examples for shot reverse shots and cross-cuttings (placeholder in final version).

6. ANNEX: Scene description in AV corpus

In this Annex we summarized the attributes and ground truth of our AV corpus, i.e. five series and five movies. The ten graphs show the content items in their entity as stored in the archive. The subsequent tables contain not only the increasing audiovisual scene boundary number (column #1) and its associated frame index (column #2), but also the increasing audio-only scene boundary number (column #4) and its associated frame index (column #5). The transition type of the audio-only scene boundary instance is specified in column #6 (N..noise, M..music, Si..silence, Sp..Speech, Cr..Crowd, U..Undefined). The final column #7 specifies the dislocation between the audio transition and the audiovisual scene boundary.

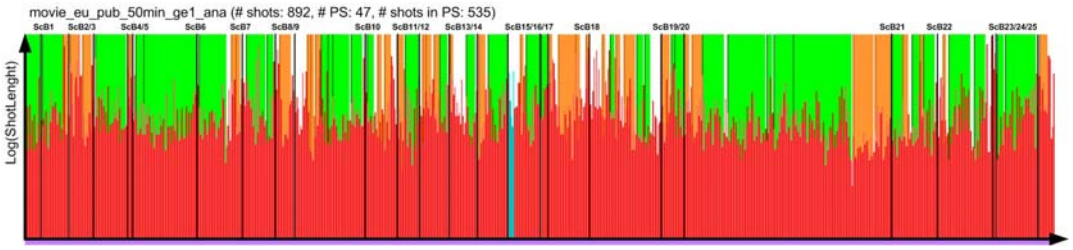


Figure 139. 'movie_eu_pub_50min_ge1_ana', a.k.a. *movie_ge1*.

Table 69.Scene description for 'movie_eu_pub_50min_ge1_ana'.

AV ScB Number	AV ScB Frame Index	Remarks	A ScB Number	A ScB Frame Index	A ScB transition type	AV ScB to A ScB [# shots]
1	898		1	900	N→M	+1
			2	1655	Si→Sp	--
2	3037		3	3072	Cr→N	+1
			4	5862	N→M	--
			5	6788	M→Si	--
3	7092		6	7092	Sp→N	0
4	9775		7	9775	Sp→M	0
5	10557		8	10475	M→Sp	-1
6	14244		9	14250	M/Sp→N	+1
7	17874		10	17874	M/Sp→M/Sp	0
8	21843		11	21962	Sp→N	+2
			12	22710	N→M	--
9	24146		13	24347	M→Sp	+3
10	29519		14	29519	Sp→M	0
			15	31450	M→Sp	--
11	32249		16	32249	N→Si→Mu	0
			17	32820	N→N/M	--
12	34059		18	34059	U→U	0
13	36734		19	37263	Sp→N	+1
14	39432		20	39432	N→M	0
15	41775		21	41819	Sp→M	+1
			22	43015	Sp→N	--
16	45283		23	45283	M→Sp	0
17	46225		24	46252	Sp→N	+1
18	51395		25	51443	N→Sp	+1
19	58511		26	58511	Sp→Sp	0
20	61080		27	61130	M→N	+1
			28	63800	Sp→N	--
			29	67205	Sp→M	--
			30	70350	N→M	--
21	71466		31	71510	M→Sp	+1
22	74051		32	74051	M→N	+1
23	79402		-	-		--
24	80363		33	80363	N→M	+1
25	83498		34	83530	M/Sp→M	+1

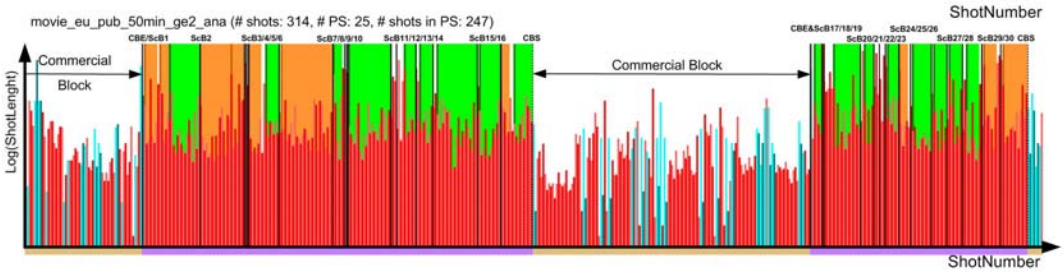


Figure 140. 'movie_eu_pub_50min_ge2_ana', a.k.a. *movie_ge2*.

Table 70.Scene description for 'movie_eu_pub_50min_ge2_ana'.

AV ScB Number	AV ScB Frame Index	Remarks	A ScB Number	A ScB Frame Index	A ScB transition type	AV ScB to A ScB [# shots]
1	6122	Dissolve (11 frames)	1	6653	M→N	+1
			2	8315	N→M	
2	10798	At two PSs	3	10800	N→Sp	+1
			4	12415	M→N	
3	14040		5	14052	M→Sp/N	+1
4	15033		6	15050	N→U→Sp	+1
5	15475		7	15505	N→Sp	+1
			8	16655	M/Sp→N	
6	17719		9	17718	N→Sp	-1
			10	18355	Sp/M→U/M	
7	20942	At two PSs	11	20942	N→Sp	0
			12	22243	N→U	
8	23068		13	23068	U→Sp	0
9	23840		14	23840	Sp/M→Sp/N	0
10	24528		15	24528	N→Si→M	0
11	26962		16	26962	N→N	0
			17	27710	M/N→U	
12	28234		18	28240	N→N	+1
13	31396		19	31396	Cr→Si	0
14	33584		20	33584	Sp→M	0
15	36618		21	36618	M→N	0
16	38057		22	37980	Sp→M	-1
17	48058	[23]	23	48060	M→Si→(Sp,N)	0
18	49465		24	49465	Sp→M	0
19	51487		25	51569	M→Sp	+1
			26	52011	M→M	
			27	53876	N→Si→Sp	
20	56329		28	56290	Sp→M	-1
21	58759		29	58759	Sp→M/N	0
22	60919		30	60919	N→Si	0
			31	61600	M→M/Sp	
23	62932		32	62932	M→M/Sp	0
24	64923		33	64923	Sp→N	0
25	66209		34	66209	Sp/M→N/M	0
			35	66380	N→Sp	
26	67084		36	67084	Sp→Sp	0
27	68238		37	68238	Sp→M	0
28	70458		38	70458	M→N/M	0
29	72241		39	72241	Sp→U	0
			40	75000	M→Sp	
30	76332		41	76332	Cr→M	0
Start / stop of non-content sequences:						
	2928		Commercial 1 start			
	6122		Commercial 1 end			
	40881	Fade to commercial	End of Scene (X) - Commercial 2 start			
	48060	Fade out commercial	Commercial 2 end – Start of Scene (X+1)			
	79729	Fade in commercial	EPG End, Commercial start			

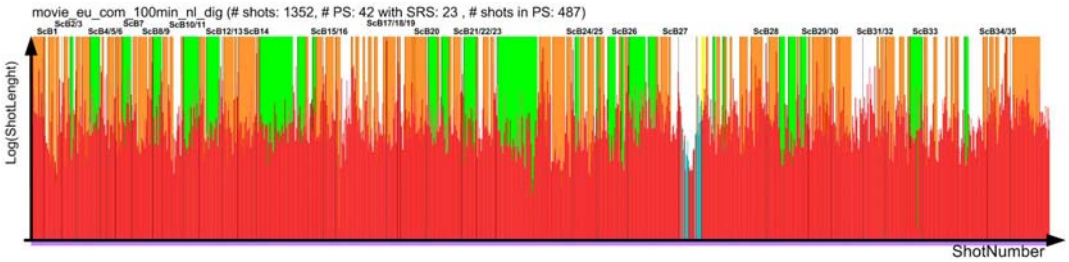


Figure 141. 'movie_eu_com_100min_nl_dig', a.k.a. *movie_nl*.

Table 71.Scene description for 'movie_eu_com_100min_nl_dig'.

AV ScB Number	AV ScB Frame Index	Remarks	A ScB Number	A ScB Frame Index	A ScB transition type	AV ScB to A ScB [# shots]
			1	1120	Sp→M	
			2	3238	M→Sp	
1	3888		3	3900	U→N	+1
2	5529		4	5540	N→Sp	+1
3	7771		5	7860	M→N	+1
4	10970		6	10970	N→N	0
5	13099		7	13110	U→N	+1
6	13980		8	14081	M→M/Sp	+1
7	16567		9	16567	Si→N	0
			10	18259	U→N	
8	19128		11	19128	N→Si	0
			12	20490	Si→N→Sp	
9	20753		13	20760	Sp→M	+1
10	23087		14	23087	N→Si	0
11	24799		15	24800	U→U	+1
12	27807		16	27643	N→N→Sp	-1
			17	28100	M→Cr	
13	29749		18	29560	Sp→Sp/M	-1
14	31622		19	31620	M→Sp	-1
15	40574		20	39704	N→M	-5
16	42410		21	42420	M→U→Sp	+1
			22	44120	Sp→M	
17	49762		--	--		
18	51988		23	51988	Si→N	0
19	53175		24	53170	Sp→Si→Sp	-1
20	56422		25	56544	M→N	+1
			26	60392	M→Sp	
21	61224		27	61224	Sp→M	0
22	63036		28	63036	M→Sp	0
			29	65400	Si→Sp	
23	65712		30	65712	Sp→U	0
			31	68754	Sp→M	
			32	70220	M→Sp	
			33	71900	M→U	
24	73656		34	73680	M→Sp	+1
			35	74440	M→Sp	
			36	74910	Sp→M	
25	76154		37	76150	N→Sp	-1
26	79664		38	79680	Sp→M	+1
27	87795		39	87790	Cr→M	-1
			40	89489	M→Sp	
			41	92110	Sp→Sp	
28	100640		42	100720	M→M/N	+1
			43	102750	Sp→M	
			44	105341	Sp→N	
29	105505		45	105500	N→N	-1
30	109219		46	109210	Sp→N	-1

[23] Sc(X): fade-out to CB, Sc(X+1): fade-in after CB (12 frames)

			47	111540	U→Cr	
			48	111690	Cr→M	
			49	111951	M→Cr	
31	112577		50	112577	Sp→M	0
32	116545		51	117930	Sp→M	+14
33	121019		52	121000	Sp/M→N/M	-1
34	126201					
			53	128744	M→N	
35	132126		54	132600	M→U.M	+5
			55	136430	Si→Sp	
			56	137080	M→M	
			57	137881	M→Sp/M	
Potential semantic scenes						
	68654	0	Inside PS, inside car, new semantic action, silence at SB			
	71895	-23	Inside PS, inside car (same location), new semantic action			
	136250	-76	Inside a PS, chaptering on DVD, discussion with semantic change, Long shot, music change shortly after			

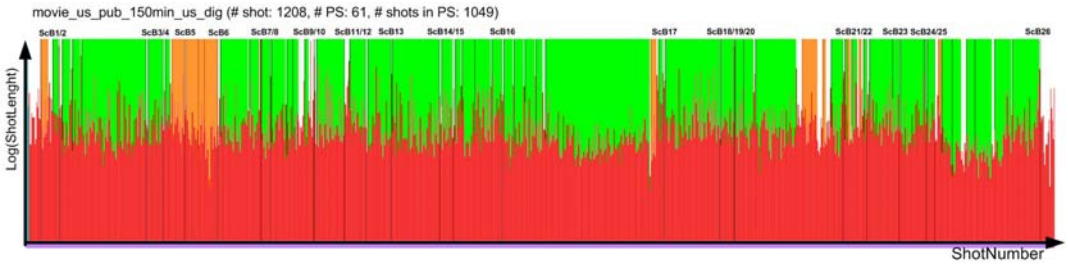


Figure 142. 'movie_us_pub_150min_us_dig', a.k.a. *movie_us_dig*.

Table 72.Scene description for 'movie_us_pub_150min_us_dig'.

AV ScB Number	AV ScB Frame Index	Remarks	A ScB Number	A ScB Frame Index	A ScB transition type	AV ScB to A ScB [# shots]
1	4122		1	4480	Sp→N	+2
2	9166		2	9166	Sp/M→M	0
			3	9800	N→Sp	
			4	10756	Sp→N	
3	22749		5	22749	M→Cr	0
4	24837		6	24837	Sp/M→M	0
			7	26052	Sp→M	
5	28892		8	28900	Si→M	+1
6	31854		9	31854	Cr/M→Si→Sp	0
			10	37675	Sp→N→Sp	
7	40264		11	40270	N→Sp	+1
8	41080		12	41080	Sp→M/Sp	0
9	49138		13	49138	Sp→M	0
10	50347		14	50347	M/Sp→Sp	0
11	56281		15	56281	Sp→M	0
			16	57237	Sp/M→U	
12	61322		17	61322	Sp→M	0
13	65025		18	65025	Sp→Cr/Sp	0
14	71816		19	71523	Sp→M/Sp	-2
15	73992		20	74000	U→N	+1
16	82912		21	82950	Sp→N→M	+1
			22	95954	Sp→Cr	
17	98683		23	98786	M→M/N	+1
18	109856		24	109912	M→M/Sp	+1
19	111492		25	111500	Sp→N	+1
20	113267		26	113290	N→M/Sp	+1
			27	125540	M→Cr	
21	127699		28	127844	M→Sp	+1
			29	132300	M→M/Sp	
22	133498		30	133498	M→N	0
23	139603		--	--		
24	143165		31	143032	Si→M/Sp	-2
25	144408		--	--		
			32	149225	N→N	
26	153914				Sp/U→N	0

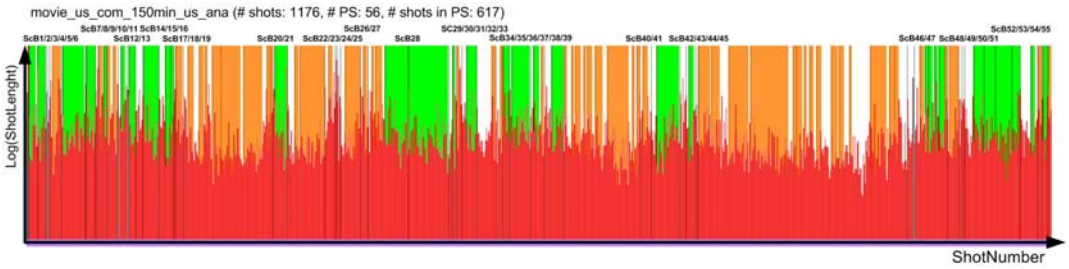


Figure 143. 'movie_us_com_150min_us_ana', a.k.a. *movie_us_ana*.

Table 73.Scene description for 'movie_us_com_150min_us_ana'.

AV ScB Number	AV ScB Frame Index	Remarks	A ScB Number	A ScB Frame Index	A ScB transition type	AV ScB to A ScB [# shots]
			1	886	Si→M	
			2	3300	M→Si	
1	5563		3	5141	N→Sp	+1
			4	6660	Sp→N	
2	7111		5	7018	Sp→M	-2
3	9921		6	9921	M→M/N	0
4	10709	Computer Generated Dissolve (26 frames)	7	10810	Sp→M	-1
5	11889		8	11889	Sp→M	0
			9	12959	Si→N	
			10	16030	Sp→N	
6	16292		11	16292	Sp→U	0
7	19948		12	19926	Sp→N	-1
8	20821		13	20806	Sp→Sp/M	-1
9	23193		14	23193	Sp/M→N	0
10	25966		15	25966	U→N	0
11	30169		16	30410	U→N	+1
			17	32500	N→	
12	33255	Fade-out / Fade-in (35 frames)	18	33350	M→N	-1
			19	35170	U→Cr	
			20	35369	Cr→Sp/M	
13	36370	Fade-out / Fade-in (42 frames)	21	37100	N→N	+1
			22	37813	Sp→N	+2
14	39071		23	39071	Sp→U	0
			24	40105	Sp→N/Sp	
15	41300		25	41247	N→U/N	-1
16	44858		26	44885	Sp→N	+1
			27	45995	N→Sp	
17	48144		28	48144	Sp→U	0
18	49357		29	49357	N→Sp	0
			30	51107	Sp→N	
	51872	Narrative ScB, but inside 2nd PS GT ➔ deleted	31	52250	N→N	+2
19	63017		32	63036	Sp→N	+1
20	67885		--	--		
21	71309		33	71309	Sp→U→Sp	0
			34	72253	Sp→Sp	
22	76595		35	76595	Sp→Sp	0
23	79794		36	79794	Sp→Sp/M	0
24	82118		37	81981	Sp→M	-1
25	84569		38	84569	M→Sp	0
			39	85358	Sp→N	
26	87253		40	87253	Sp→N	0
			41	88020	Sp/N→N	
27	96096	No video cut but ScB	42	96431	M/Sp→Sp	-1
28	101514		43	101514	Cr→Sp	0
29	101961		44	101961	Sp→N	0
30	103468		45	103468	Sp→N	0
31	108324		46	108324	Sp/Cr→Sp	0
32	111548		47	111471	Sp→Si→S	-1

					p	
33	114684		--	--		
34	115631		48	115631	Sp→Si→Sp	0
35	119282		49	119282	Sp/M→Sp	0
36	123783	Same room, different time	--	--		
37	126417		50	126350	Si→Sp	-1
			51	127015	N→M/N	
			52	128139	Sp→Sp	
38	131052		53	131052	N→Sp	
			54	132860	Sp→Cr	
			55	135253	Sp→Si→Sp	
			56	139299	N→Sp	
39	139909		57	139909	M→Sp	0
40	142132		58	142132	U→N	0
41	146322		59	146322	Sp/M→N	0
			60	147366	N→Sp	
42	149598		61	149598	Sp→N	0
			62	151402	Sp→M	
43	151736		63	151736	N→Sp	0
44	152434		64	152434	Sp→N	0
			65	156197	N→N	
			66	156948	N/Sp→N	
			67	157718	Sp→N	
			68	168000	N→Si→N	
	171316	After 2nd PS GT annotation deleted	Confirmed new AScB	171511	U → M	
45	174681	Computer Generated Wipe Eye transition into eye and abject from black eye (40 frames)	69	173936	M→N	0
46	182671		70	182500	M→N	-1
47	184079		--	--		
48	185301		71	185301	Sp→Sp	0
49	186584		72	186584	Sp→U→Sp	0
			73	189744	Sp→N/Sp	
50	195775		74	195775	Sp/M→Cr	0
51	198926		--	--		
52	200075		75	200075	Sp→Si→U	0
53	201649					

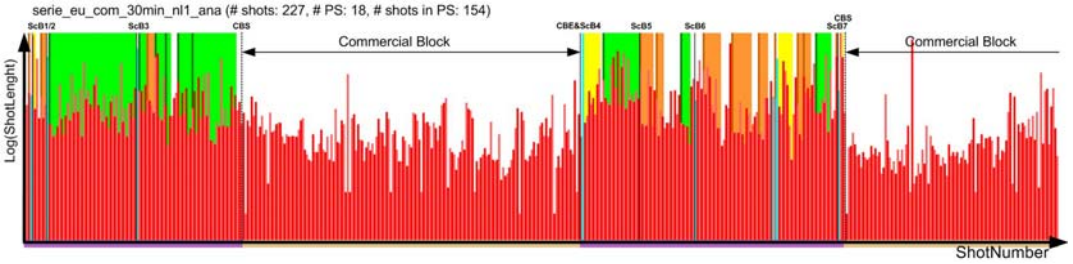


Figure 144. serie_eu_com_30min_nl1_ana, a.k.a. *serie_nl1*.

Table 74.Scene description for 'serie_eu_com_30min_nl1_ana'.

AV ScB Number	AV ScB Frame Index	Remarks	A ScB Number	A ScB Frame Index	A ScB transition type	AV ScB to A ScB [# shots]
1	433		1	314	M→Cr	-1
2	2125 - 2167	Dissolve: 42	2	2084	Sp→U	-1
3	7137 - 7160	Dissolve: 23	Sp → Sp with no break			
4	20560	Sc(X): CB, Sc(X+1):				
5	26635	At 2 PSs	3	26610	Sp→M	-1
6	29376 - 29393	Computer Generated Wipe: 17	4	29380	M→Cr	0
			5	33140	M→U→ Sp	
7	39105 - 39135	Dissolve: 30	6	39150	Sp→N	0
Start / end of non-content sequences						
	2		EPG start			
	13900		Commercial 1 start (Cut)			
	20559		Commercial 1 end – Start of Scene (X+1)			
	40631		Commercial 2 start			
	44563		Commercial 2 end			

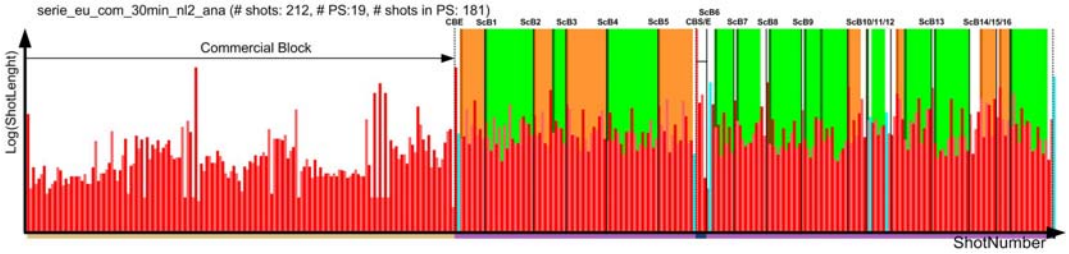


Figure 145. 'serie_eu_com_30min_nl2_ana', a.k.a. *serie_nl2*.

Table 75.Scene description for 'serie_eu_com_30min_nl2_ana'.

AV ScB Number	AV ScB Frame Index	Remarks	A ScB Number	A ScB Frame Index	A ScB transition type	AV ScB to A ScB [# shots]
			1	10690	M→M	
1	11982		2	11981	M→Sp	0
2	14009		3	14009	Sp/M→Sp	0
3	15816		4	15816	Cr→Sp	0
4	17832		5	17832	M→Sp	0
5	19661		6	19661	M→Sp	0
6	30391		7	30400	M→Si→M	0
7	32190		8	32190	Sp→N/Sp	0
8	33822		9	33822	M→M	0
9	35528		10	35528	Sp→N	+1
			11	36440	Sp→U	
10	37050		12	37050	N→Sp	0
11	38666		13	38663	Sp→M	+1
12	39949		14	49933	M→M	+1
13	42936		15	42960	M→M	+1
14	44817		16	44817	M→N→Sp	0
15	46934		17	46935	Sp/M→Sp/M	+1
16	48127		18	48127	M→Sp	0
			19	50187	M→M	
Start / end of non-content sequences						
	27			Commercial 1 start		
	9172			Commercial 1 end, EPG start		
	21658			Commercial 2 start (End Scene X)		
	30377			Commercial 2 end (start Scene X+1)		
	51063			EPG End		

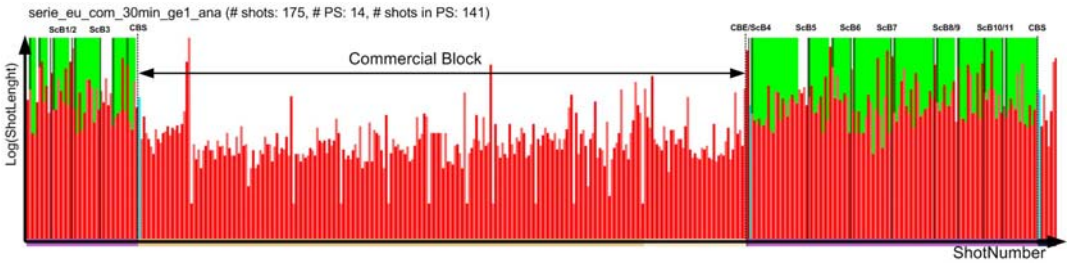


Figure 146. 'serie_eu_com_30min_ge1_ana', a.k.a. *serie_ge1*.

Table 76.Scene description for 'serie_eu_com_30min_ge1_ana'.

AV ScB Number	AV ScB Frame Index	Remarks	A ScB Number	A ScB Frame Index	A ScB transition type	AV ScB to A ScB [# shots]
1	1431		1	1430	Sp/M→Si	-1
2	3011		2	3010	Sp→U→M	-1
3	4747		3	4750	M→Si	+1
4	17241	[24]	--	--		
5	19509		4	19540	M→N→Sp	+1
6	22131		5	22130	Si→M	-1
7	24199		6	24200	M→U	+1
8	26609		7	26610	Sp→M	+1
9	27937	At 2 PSs	8	27937	Sp/M→Sp	0
10	29751	At 2 PSs	9	29750	N→M	-1
11	31040		10	31040	Sp/M→Sp	0
Start / end of non-content sequences						
	32			EPG start		
	6825			End of Scene (X) - Commercial start		
	17258			Commercial end – Start of Scene (X+1)		
	33080			EPG end		

[24] Sc(X): fade-out to CB, Sc(X+1): fade-in after CB (23 frames).

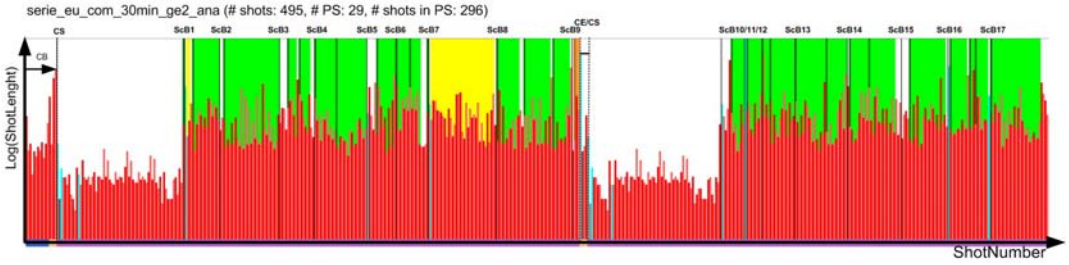


Figure 147. 'serie_eu_com_30min_ge2_ana', a.k.a. *serie_ge2*.

Table 77.Scene description for 'serie_eu_com_30min_ge2_ana'.

AV ScB Number	AV ScB Frame index	Remarks	A ScB Number	A ScB Frame Index	A ScB transition type	AV ScB to A ScB [# shots]
1	3104 - 3154	Dissolve: 50	1	3140	M→Si→N	0
2	5697		2	5734	M→N	+1
3	8970	At 2 PSs	3	8969	M→Sp	+1
4	11961		4	11984	M→Sp	+1
5	14154 - 14241	Fade-out → Fade-in: 87	5	14230	M→Si→Sp	0
6	17108		6	17200	M→N→ Sp	+1
7	20012 – 20130	Fade-out → Fade-in: 113	7	20110	M→Si--<N	0
8	24826		8	24861	M→N	0
9	29875		9	29875	N→M	0
10	32939 - 32991	Fade-out → Fade-in: 52	10	32970	M→Si→N	0
11	35155 - 35195	Dissolve: 40	Sp (35078) → M → M/Sp (35195)			
12	38553		11	38625	M→Sp	+1
13	42010	At 2 PSs	12	42040	M→N	+1
14	45799		13	45805	M→N	+1
15	50246		14	50280	M→Sp	+1
16	53421 - 53516	Fade-out → Fade-in: 95	15	53491	M→Si→M	0
17	57620		16	57650	M→Sp	+1
Start / end of non-content sequences						
	2130	EPG 1start				
	30398	EPG 1 end				
	31920	EPG 2 start				
	61870	EPG 2 end				

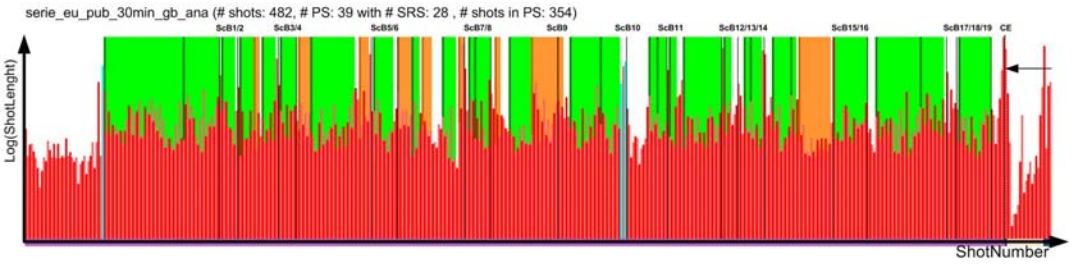


Figure 148. 'serie_eu_pub_30min_gb_ana', a.k.a. *serie_gb*.

Table 78.Scene description for 'serie_eu_pub_30min_gb_ana'.

AV ScB Number	AV ScB Frame index	Remarks	A ScB Number	A ScB Frame index	A ScB transition typ	AV ScB to A ScB [# shots]
			1	2089	M→M	
			2	2439	M/N→Sp	
1	6212		3	6230	Sp→N	+1
2	7065	At 2 PSs	4	7065	N→Sp	+1
3	8671		5	8647	Sp→M	-1
4	9467	At 2 PSs	6	9467	Sp→N	0
			7	11122	Sp→M	
5	12352		8	12380	M→N	-1
6	13137		9	13137	N→N	0
			10	13863	Sp→M	
			11	15045	M→N	
7	15629		12	15629	N→Sp	0
8	16927		13	16927	N→Sp	0
9	19031		14	19031	N→U	0
			15	19539	Sp→Sp	
			16	21700	Sp/M→Sp	
10	22117		17	22117	Sp→U	0
			18	22581	M/Sp→Sp	
11	23146		19	23146	Sp→N	0
			20	23637	Sp→M	
			21	24217	M→Sp	
12	25185		22	25185	N→Sp	0
			23	25870	N→Sp	
13	26410		24	26409	N→Sp	-1
14	27926		25	27926	N→Sp	0
			26	28467	U→Sp	
			27	29502	Sp→M/Cr	
15	30152		28	30180	M→Sp	+1
16	31983		29	32000	U→Sp	+1
			30	33338	Cr→Sp	
17	34412		31	34412	N→U	0
18	35132		32	35132	N→N	0
19	35981		33	36000	N→Sp	+1
			34	36695	N→M	
Start / end of non-content sequences						
	1031		EPG start			
	37678		EPG end			

7. ANNEX: Formulae for AV Jitter

Here we present the development of the maximum likelihood test from Louis [37]. For the calculation of the maximum likelihood thresholds, i.e. the cross-over points of the following two probability functions

$$L_{ScB} = \frac{pdf\left(j \mid H_{ScB}\right)pdf\left(H_{ScB}\right)}{pdf\left(j\right)}$$
$$L_{NScB} = \frac{pdf\left(j \mid H_{NScB}\right)pdf\left(H_{NScB}\right)}{pdf\left(j\right)} \qquad (5\text{-}13),$$

we applied the equation $L_{ScB} = L_{NScB}$, which equals to $L_{ScB} / L_{NScB} = 1$. It can be rewritten (given an Guassian distribution) as

$$\frac{\frac{1}{\sqrt{2\pi\sigma^{2}_{ScB}}}e\left(-\frac{\left(j-\mu_{ScB}\right)^{2}}{2\sigma^{2}_{ScB}}\right)}{\frac{1}{\sqrt{2\pi\sigma^{2}_{NScB}}}e\left(-\frac{\left(j-\mu_{NScB}\right)^{2}}{2\sigma^{2}_{NScB}}\right)} * \frac{P\left(H_{ScB}\right)}{1-P\left(H_{ScB}\right)} \begin{cases} < \\ > \end{cases} 1 \qquad (5\text{-}14).$$

This equation can be further developed into

$$-\frac{1}{2}\ln\sigma^{2}_{ScB} - \frac{1}{2}\frac{\left(j-\mu_{ScB}\right)^{2}}{\sigma^{2}_{ScB}} + \frac{1}{2}\ln\sigma^{2}_{NScB} + \frac{1}{2}\frac{\left(j-\mu_{NScB}\right)^{2}}{\sigma^{2}_{NScB}} \begin{cases} < \\ > \end{cases} \ln\left(\frac{1-P\left(H_{ScB}\right)}{P\left(H_{ScB}\right)}\right) \qquad (5\text{-}15),$$

$$\ln\sigma^{2}_{NScB} + \frac{\left(j-\mu_{NScB}\right)^{2}}{\sigma^{2}_{NScB}} - \ln\sigma^{2}_{ScB} - \frac{\left(j-\mu_{ScB}\right)^{2}}{\sigma^{2}_{ScB}} \begin{cases} < \\ > \end{cases} 2*\ln\left(\frac{1-P\left(H_{ScB}\right)}{P\left(H_{ScB}\right)}\right) \qquad (5\text{-}16),$$

$$\left(j-\mu_{NScB}\right)^{2}*\sigma^{2}_{ScB} - \left(j-\mu_{ScB}\right)^{2}*\sigma^{2}_{NScB} \begin{cases} < \\ > \end{cases} \sigma^{2}_{ScB}*\sigma^{2}_{NScB}*\left(\ln\sigma^{2}_{ScB} - \ln\sigma^{2}_{NScB} + 2*\ln\left(\frac{1-P\left(H_{ScB}\right)}{P\left(H_{ScB}\right)}\right)\right) \qquad (5\text{-}17),$$

resulting in

$$j^{2}\sigma^{2}_{ScB} - 2j\mu_{NScB}\sigma^{2}_{ScB} + \mu_{NScB}{}^{2}\sigma^{2}_{ScB} - j^{2}\sigma^{2}_{NScB} + 2j\mu_{ScB}\sigma^{2}_{NScB} - \mu_{ScB}{}^{2}\sigma^{2}_{NScB} \begin{cases} < \\ > \end{cases} X \qquad (5\text{-}18),$$

with

$$X = \sigma^{2}_{ScB} * \sigma^{2}_{NScB} * \left(\ln\sigma^{2}_{ScB} - \ln\sigma^{2}_{NScB} + 2*\ln\left(\frac{1-P\left(H_{ScB}\right)}{P\left(H_{ScB}\right)}\right)\right) \qquad (5\text{-}19).$$

8. ANNEX: Formulae for Shot Length

For the calculation of the maximum likelihood thresholds, i.e. the cross-over points of the following two probability functions, as in Annex 7,

$$L_{ScB} = \frac{pdf(j \mid H_{ScB})pdf(H_{ScB})}{pdf(j)}$$
$$L_{NScB} = \frac{pdf(j \mid H_{NScB})pdf(H_{NScB})}{pdf(j)} \quad (5\text{-}20),$$

the equation $L_{ScB} = L_{NScB}$, which equals to $L_{ScB} / L_{NScB} = 1$, can be rewritten as

$$\frac{\frac{j}{\sigma^2_{ScB}} e\left(-\frac{j^2}{2\sigma^2_{ScB}}\right)}{\frac{j}{\sigma^2_{NScB}} e\left(-\frac{j^2}{2\sigma^2_{NScB}}\right)} * \frac{P(H_{ScB})}{1-P(H_{ScB})} \lessgtr 1 \quad (5\text{-}21).$$

Finally, this equation can be further developed into

$$\ln\frac{\sigma^2_{NScB}}{\sigma^2_{ScB}} - \frac{1}{2} * \frac{j^2}{\sigma^2_{ScB}} + \frac{1}{2} * \frac{j^2}{\sigma^2_{NScB}} \lessgtr \ln\left(\frac{1-P(H_{ScB})}{P(H_{ScB})}\right) \quad (5\text{-}22),$$

and

$$\frac{j^2}{\sigma^2_{NScB}} - \frac{j^2}{\sigma^2_{ScB}} \lessgtr 2 * \left[\ln\left(\frac{1-P(H_{ScB})}{P(H_{ScB})}\right) - \ln\frac{\sigma^2_{NScB}}{\sigma^2_{ScB}}\right] \quad (5\text{-}23).$$

9. ANNEX: MPEG-7 descriptors for Service Units

In this Annex we specify the MPEG-7 compliant output of several of our service units of our analysis framework.

MPEG-7 compliant output of service unit Cut Detector

For reasons of interoperability we convert the output stream of the service unit shot boundary detector into an XML-based MPEG-7 format, as described in [150] and [151]. Abrupt transitions between shots, i.e. cuts, are represented in MPEG-7 as a *Global Transition* of *Cut* type, see Figure 149. MPEG-7 provides the Description Scheme (DS) 'Analytic Edited Video' to describe video items with gradual and abrupt video editing effects. In the specific case of a shot decomposition, the DS 'Analytic Editing Temporal Decomposition' describes the segmentation in shots and transitions and DS 'Global Transition' the transition itself. The Descriptor (D) 'Media Time' and 'Media RelIncr Time Point' specifies the frame at which the transition starts, as shown underneath for the case of a cut instance.

MPEG-7 compliant output of service unit Gradual Transition Detector

An XML-based MPEG-7 compliant description example of a gradual transition had been sketched in Figure 149, using as defined by the standard Description Scheme *GlobalTransition* and Descriptors *MediaTime* (*MediaRelIncTimePoint*, *MediaIncDuration*) and *EvolutionType* (*Name*).

Abrupt Shot Boundary: 'Cut'

```
<Description xsi:type="ContentEntityType">
 <MultimediaContent xsi:type="VideoType">
  <AnalyticEditedVideo xsi:type="EditedVideoType">
   <AnalyticEditingTemporalDecomposition gap="false" overlap="false">
    <GlobalTransition>
     <MediaTime>
      <MediaRelIncrTimePoint mediaTimeUnit="PT1N1000000F">1
      </MediaRelIncrTimePoint>
     </MediaTime>
     <EvolutionType href="urn:mpeg:mpeg7:cs:EvolutionTypeCS:2001">
      <Name xml:lang="en">Cut</Name>
     </EvolutionType>
    </GlobalTransition>
   </AnalyticEditingTemporalDecomposition>
  </AnalyticEditedVideo>
 </MultimediaContent>
</Description>
```

Gradual Shot Boundary: 'Dissolve', 'Fade',..

```
<Description xsi:type="ContentEntityType">
 <MultimediaContent xsi:type="VideoType">
  <AnalyticEditedVideo xsi:type="EditedVideoType">
   <AnalyticEditingTemporalDecomposition gap="false" overlap="false">
    <GlobalTransition>
     <MediaTime>
      <MediaRelIncrTimePoint mediaTimeUnit="PT1N1000000F">1
      </MediaRelIncrTimePoint>
      <MediaIncrDuration>30
      </MediaIncrDuration>
     </MediaTime>
     <EvolutionType href="urn:mpeg:mpeg7:cs:EvolutionTypeCS:2001">
      <Name xml:lang="en">dissolve</Name>
     </EvolutionType>
    </GlobalTransition>
   </AnalyticEditingTemporalDecomposition>
  </AnalyticEditedVideo>
 </MultimediaContent>
</Description>
```

Figure 149. XML-based MPEG-7 description of cut and gradual transition instances.

MPEG-7 compliant output of Audio Decoding Time Stamp

Audio DTS represents the time instant to decode and display an audio segment. It is specified using basic MPEG-7 data types MediaTime, MediaRelIncrTimePointType, and mediaDurationType, assuming the notion of time encoded within the audio-visual sequence being described.

MPEG-7 compliant output of Audio Silences

Audio silence reflects the property of a segment that only sounds of intensity smaller than a defined threshold occur. The MPEG-7 compliant output is shown in Figure 150.

MPEG-7 compliant output of Commercial Cut Silence Detection

For the commercial cut silence detection we specified an *Audio Descriptor* of type *CommercialBlockSilenceType*, as shown in Figure 151.

```
<Description xsi:type="ContentEntityType">
 <MultimediaContent xsi:type="AudioType">
  <Audio>
   <MediaTime>
    <MediaRelIncrTimePoint mediaTimeUnit="PT1N30F">10
    </MediaRelIncrTimePoint>
   </MediaTime>
  </Audio >
 </MultimediaContent>
</Description>
```

MPEG-7: *Audio Decoding TimeStamp*

```
<Description xsi:type="ContentEntityType">
 <MultimediaContent xsi:type="AudioType">
  <Audio>
   <TemporalDecomposition gap="true" overlap="false">
    <AudioSegment>
     <MediaTime>
      <MediaRelIncrTimePoint mediaTimeUnit="PT1N1000000F">2
      </MediaRelIncrTimePoint>
      <MediaIncrDuration mediaTimeUnit="PT1N1000F">3
      </MediaIncrDuration>
     </MediaTime>
     <AudioDescriptor xsi:type="SilenceType"/>
    </AudioSegment>
   </TemporalDecomposition>
  </Audio>
 </MultimediaContent>
</Description>
```

MPEG-7: *Audio silence*

Figure 150. MPEG-7 compliant description of commercial block silences in the framework.

```
<?xml version="1.0" encoding="iso-8859-1"?>
 <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001" xsi:schemaLocation="urn:mpeg:mpeg7:schema:2001 Mpeg7-2001.xsd">
  <Description xsi:type="ContentEntityType">
   <MultimediaContent xsi:type="AudioType">
    <Audio>
     <TemporalDecomposition>
      <AudioSegment>
       <MediaTime>
        <MediaRelIncrTimePoint mediaTimeUnit="PT1N1000000F">1</MediaRelIncrTimePoint>
        <MediaIncrDuration mediaTimeUnit="PT1N1000F">3</MediaIncrDuration>
       </MediaTime>
       <AudioDescriptor xsi:type="CommercialBlockSilenceType"/>
      </AudioSegment>
     </TemporalDecomposition>
    </Audio>
   </MultimediaContent>
  </Description>
</Mpeg7>
```

MPEG-7: *Commercial Cut Silence*

Figure 151. MPEG-7 compliant description of commercial cut silence.

10. ANNEX: Manual post-annotation tool for Consumer Recording Devices (Application for AV Segmentation)

The problem with statistical-analysis-based genre classifiers, such as commercial block detectors as described in 4.4 or EPG content boundaries, are their long decision window resulting in a time-wise imprecision at the boundaries. Consequently, this leads to untimely start and end of e.g. commercial block detections degrading the reliability of the detector and consequently, to displeaseness at the consumer side. Today's analysis methods can offer only limited accuracy and therefore easy-to-use and intuitive manual post-annotation tools are a viable offer to the consumer to solve the problem.

Manual Post-Annotation Tool

Fortunately, the offset of the automatically detected e.g. CB boundaries is in the range of few shots. Consequently, it is possible to implement an easy-to-use and intuitive manual post-annotation tool to shift time-wise dislocated boundaries to their appropriate position. The tool becomes feasible due to the simultaneous availability of a shot-, see section 0, and a scene boundary detector, as described in 4.6.

To start the procedure the viewer has to inform the system, e.g. HDD recorder, that he would like to readjust the boundaries of the automatic segmentation.

To understand the principle some examples will be given. In the first example the CBD identifies position 'A' as the end of the CB, as shown in Figure 152 and Figure 153. The real CB ends at the end of scene 1 (Sc 1), visualized in grey. As sketched in Figure 152, the *Graphical User Interface* (GUI) enables the user to relocate the boundary with minimal effort. The initial GUI (left rectangular) plays-back the video sequence in its main screen starting from instance 'A'. The upper-left *Picture-in-Picture* (PiP) displays scene 2 (Sc 2) and the upper-right PiP scene 3 (Sc 3). The user sees that scene 2 and sequence 'A' are both non-commercial content. Therefore, after pressing the 'backwards button' the GUI adapts (right GUI) and the upper-right PiP plays-back the last part of the CB. This provides the viewer with the confirmation that he/she retrieved the exact end instance of the CB, i.e. here sh1, and he /she freezes the new boundary by pressing the 'confirm' button.

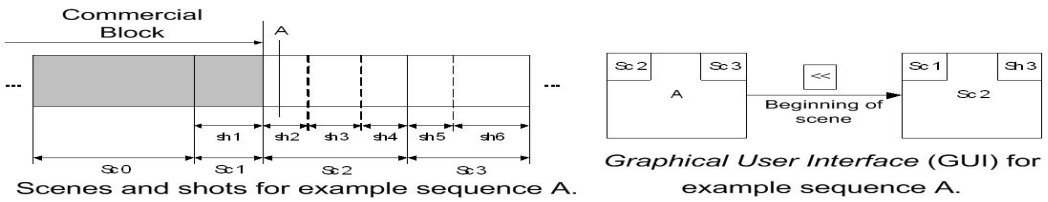


Figure 152. Example sequence 'A'.

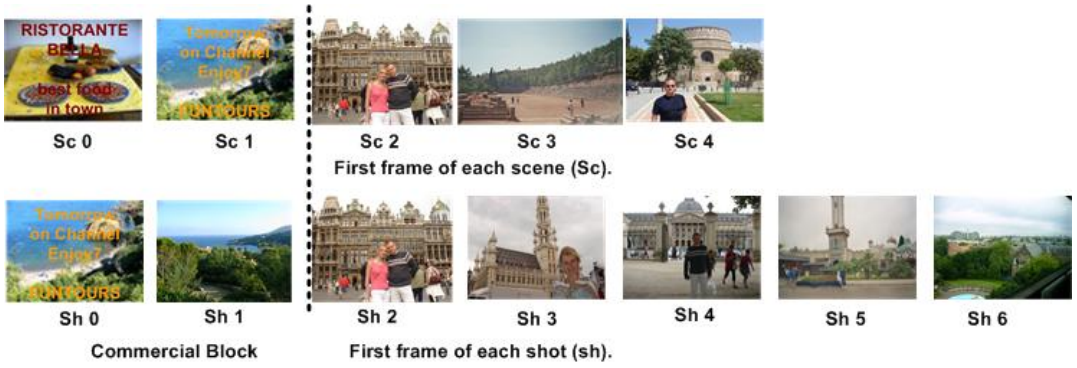


Figure 153. Scenes and shots for manual post-annotation[1].

Figure 154 –GUI of Figure 152 for sequence A with scenes and shots of Figure 153[1].

In the next example the CBD defines instance 'B' as the end of the CB, as shown in Figure 155. The initial GUI (left GUI) displays the video sequence starting from 'B' in the main field and scene 3 and 4 in the two upper PiPs. Because all three are non-CB contents the user will press the 'backwards' button. The resulting GUI will play-back scene 3 in the main window, scene 2 in the upper-left PiP and shot 6 (Sh 6) in the upper-right PiP. Due to the fact that all windows display non-CB content the user will press once more the 'backwards' button. In this case the main window will display scene 2, the upper-left PiP scene 1 and the other shot 3. Consequently the user will press the 'confirm' button to freeze the new CB boundary between shot 1/2.

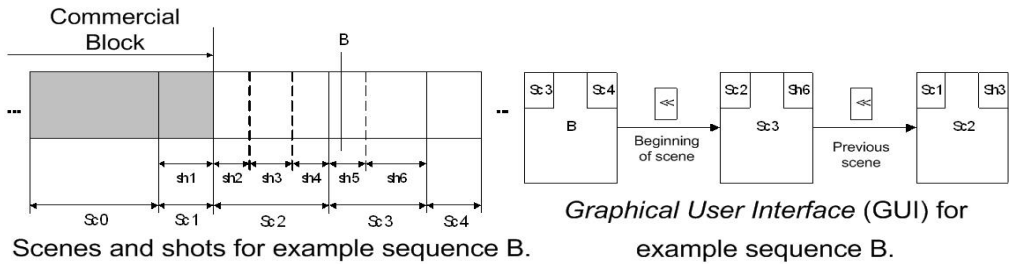


Figure 155. Example sequence 'B'.

Figure 156. – Example GUI for sequence B as shown in Figure 155[1].

Unfortunately, the current scene boundary detector has not reached 100% reliability and, therefore, misses sometimes a scene boundary *ScB*. The following example explains the situation in such a case. In the main window of the GUI, see Figure 157, the detected CB end, here from instance 'C' onwards, is displayed. The viewer doubts that instance 'C' is the real start of the non-CB content. Therefore, he/she presses the 'backwards' button and the GUI (central) displays scene 0 and 1 (PiP), both containing CB content. Consequently, he/she presses the 'forward' button and the system automatically switches from scene level to shot level. Trough this GUI (right one) the viewer gets the confirmation that he found the CB end and freezes the instance between shot 1 and 2.

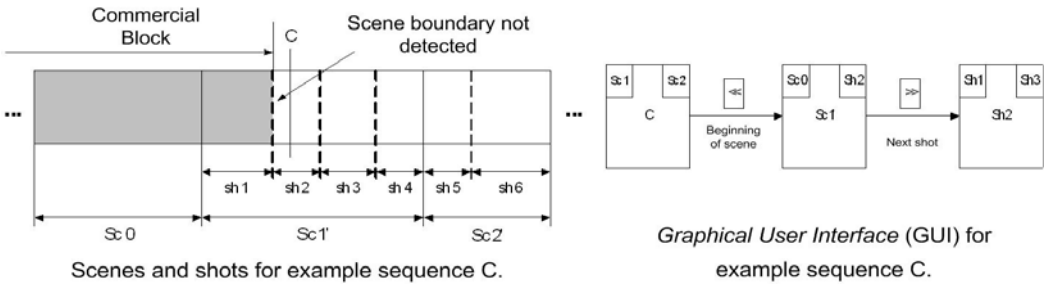


Figure 157. Example sequence 'C'.

Figure 158. – Example GUI for sequence C as shown in Figure 157[1].

The following examples will deal with the situation that the CB end is detected to early, as shown in Figure 159. In this case the initial GUI (left one) will display with the D-sequence CB content. Naturally, the viewer will press the 'forward' button and will see that in scene 2 already non-CB content is displayed. Consequently he/she will press the 'confirm' button to relocate the CB end instance to the end of scene 1.

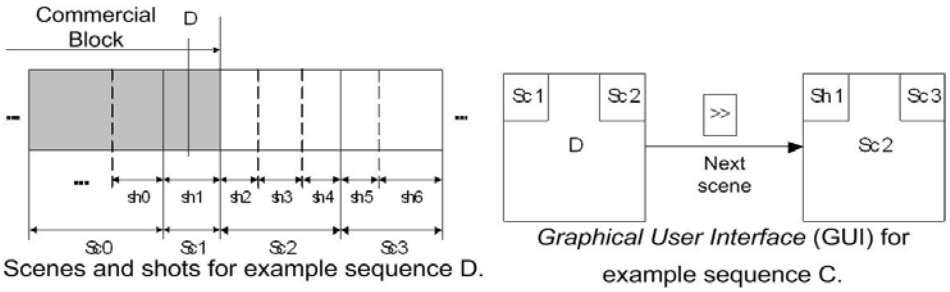


Figure 159. Example sequence 'D'.

Figure 160. – Example GUI for sequence ‘D’ as shown in Figure 159[1].

In the next example the CB end is again identified to early (Figure 161). The viewer sees in the initial GUI (left one) only CB contents in all windows. Consequently, he/she will press the ‘forward’ button and sees CB-containing scene 1 in the main window. Therefore, he/she will press again the ‘forward’ button resulting in a GUI playing-back scene 2 (as shown in right GUI). This confirms the viewer that he/she found the CB end between shot 1and 2, which, subsequently, will be confirmed.

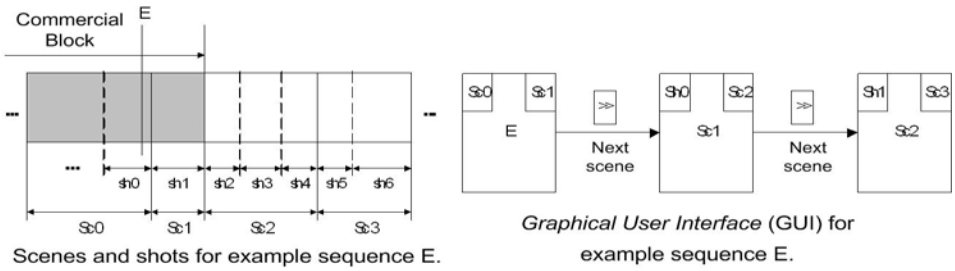
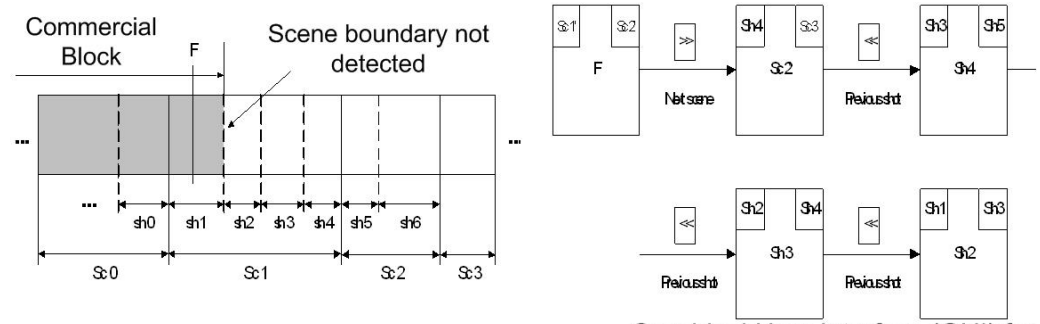


Figure 161. Example sequence ‘E’.

Figure 162. – Example GUI for sequence ‘D’ as shown in Figure 161[1].

In the last example the CB end is detected to early and a scene boundary *ScB* has been missed. The initial GUI (left GUI in Figure 163) displays with the F-sequence CB content, therefore, the viewer will press the ‘forward’ button. In the resulting GUI (second one) will display scene 2, shot 4 and scene 3, all non-CB content. Subsequently, the viewer will press the ‘backwards’ button, which will initiate the shift from scene level to shot level. This will result in the third GUI playing-back shot 3, 4 and 5 all of them non-CB sequences. Another ‘backward’ will result in a GUI displaying shot 2, 3 and 4. After the final ‘backward’ shot 1, 2 and 3 will be displayed providing the viewer with the confirmation that he/she retrieved the correct CB end, so he can freeze the instance finally.



Scenes and shots for example sequence F.

Graphical User Interface (GUI) for example sequence F.

Figure 163. Example sequence 'E'.

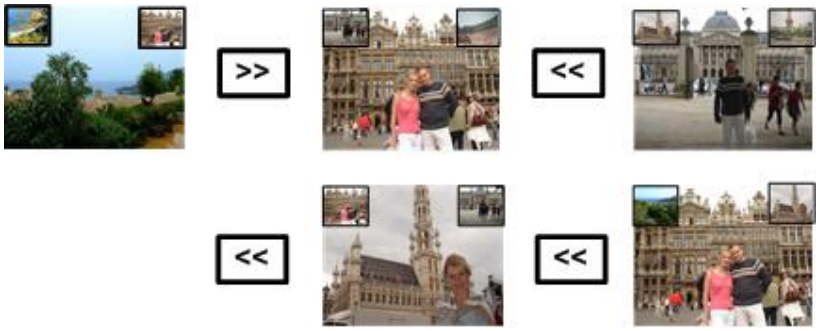


Figure 164. – Example GUI for sequence 'D' as shown in Figure 163[1].

Applications

The presented user interface solution based on available shot and scene boundary detection algorithms enables the realization of an easy-to-use and intuitive manual annotation tool to relocate improper e.g. commercial block boundaries. The solution can be used not only for commercial blocks, but also for EPG boundaries, an application to position fuzzy DVB-SI-originated EPG content item boundaries correctly. Furthermore, this solution can be used to relocate boundaries of any inaccurate classification system.

11. ANNEX: Abbreviations

AAC	Advanced Audio Coding
AC-3	aka Dolby Digital, Audio compression by Dolby Labs
API	Application Programming Interface
AV	AudioVisual
B-frame	Bi-directional (coded) frame
BD	Blu-ray Disc
CB	Commercial Block
CBD	Commercial Block Detector
CBIR	Content Based Image Retrieval
CBR	Constant Bit Rate
CD	Compact Disk
CDDB	CD Database
CE	Consumer Electronics
CIB	Content Item Boundary
CIBD	Content Item Boundary Detector
CPI	Characteristic Point Information
CPU	Central Processing Unit
CVBR	Constraint Variable Bit Rate
DAB	Digital Audio Broadcast
DCT	Discrete Cosine Transformation
DSP	Digital Signal Processor
DTS	Decoding Time Stamp
DVB	Digital Video Broadcast
DVB-SI	DVB Service Information
DVD	Digital Versatile Disk
EPG	Electronic Program Guide
GOP	Group Of Pictures
HD	High Definition
HDD	Hard Disk Device
HDTV	High Definition TeleVision
I^2S	Inter IC Sound
I-frame	Intra-(coded)-frame
IPG	Internet Program Guide
ISBN	International Standard Book Number

JPEG	Joint Photographic Experts Group
MAD	Mean Absolute Difference
MB	MacroBlock
MBP	MacroBlock Processor
MFCC	Mel-Frequency Cepstral Coefficient
MIPS	RISC CPU
MPEG	Motion Picture Expert Group
NTSC	National Television System(s) Committee
ppm	Portable Pixel Map
png	Portable Network Graphics
P-frame	Predicted-(coded)-frame
PAL	Phase Alternating Line
PC	Personal Computer
PCI	Peripheral Component Interface
RAM	Random Access Memory
RDF	Resource Description Framework
ROM	Read Only Memory
ScB	Scene Boundary
ScBD	Scene Boundary Detector
SB	Shot Boundary
SBD	Shot Boundary Detector
SD	Standard Definition (= resolution, e.g. 720*576)
SIF	Standard Interchange Format (=resolution, e.g. 352*288)
SSA	Solid State Audio
STB	Set Top Boxe
STPS	Short time power spectrum
TM	TriMedia
TOC	Table Of Contents
UI	User Interface
URL	Uniform Resource Locator
VBR	Variable Bit Rate
VCOMP	Video COMPressor
VFEND	Video FrontEND module
WMA	Windows Media Audio
XML	eXtensible Markup Language

REFERENCES

Chapter 1: Introduction

[1] J. Nesvadba, D. Kelly, I. Nagorski, 'Device and Method for Recording Information', European patent application EP 1.606.817.

[2] E. Thelen, J. Nesvadba, et.al., 'Recording Content on a Record Medium that contains desired Content Descriptors', European patent application EP 1.680.785.

Chapter 2: Section CASSANDRA Framework

[3] H. Moravec, 'Robot', ISBN: 0-19-511630-5, Oxford University Press, 1999.

[4] H. Moravec, 'When will Computer Hardware Match the Human Brain', Journal of Evolution and Technology, Vol. 1, 1998.

[5] Report Digital Strategy Group of the European Broadcasting Union, 'Media with a Purpose', November 2002.

[6] EPG providing company Gemstar, http://www.gemstartvguide.com.

[7] National broadcast company BBC, http://www.bbc.co.uk.

[8] Metadata standard 'TV-Anytime, http://www.tv-anytime.org.

[9] Metadata standard MPEG-7, http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm.

[10] Metadata standard Digital Video Broadcast – Service Information DVB-SI, http://www.dvb.org/technology/standards_specifications/multiplexing/dvbsi/index.xml.

[11] Electronic Program Guide EPG, http://en.wikipedia.org/wiki/Electronic_program_guide.

[12] J. Nesvadba, D. Burazerovic, 'Apparatus for analyzing a content stream comprising a content item', Patent Application WO 2006/077533.

[13] J. Nesvadba, D. Burazerovic, 'Extensions to EPG Boundary Enhancer', Patent Application NL000886.

[14] J. Nesvadba, P. Fonseca, 'Manual Post Annotation of AV Content', Patent Application NL004957.

Chapter 2: Scene Boundary Defintion

[15] F. Beaver, 'Dictionary of Film Terms: The Aesthetic Companion to Film Analysis', Twayne Publishing, ISBN: 0805793348, New York, 1994.

[16] D. Bordwell, K. Thompson, 'Film Art – An Introduction', Mc Graw Hill, ISBN: 0-07-248455-1, 2004.

Chapter 2: Protoyping framework

[17] J. Nesvadba, F.de Lange, 'Cassandra Framework: A Service Oriented Distributed Multimedia Content Analysis Engine', Proc. of Image Analysis for Multimedia Interactive Services, pp. 178-184, 2007.

[18] W. Halal, 'The Intelligent Internet: The Promise of Smart Computers and E-Commerce', Journal 'The Futurist', The World Future Society, pp. 78-83, 2007.

[19] M. Huhns, 'The Sentient Web', IEEE Internet Computing, Vol. 7, Nr 6, pp. 82-84, 2003.

[20] H. Moravec, 'Robots, After All', Comm. ACM, vol. 46, Nr. 10, pp. 91-97, 2003.

[21] F. de Lange, J. Nesvadba, 'Rapid Prototyping of Multimedia Analysis Systems - A Networked Hardware/Software Framework', Proc. Conf. on Web Information Systems and Technologies, pp. 104-109, 2005.

[22] F de Lange, J. Nesvadba, 'Applying PC network technology to assess new multimedia content analysis applications for future consumer electronics storage devices', Proc. Conf. on Intelligent Multimedia Computing and Networking, pp. 304-309, 2005.

[23] Universal Home API, 'A new Application Programming Interface for the CE Industry, http://www.uhapi.org.

[24] F. de Lange, J. Nesvadba, 'Early Evaluation of Future Consumer AV Content Analysis Applications with PC networks', Journal Multimedia Tools and Applications, vol. 34, issue #2, pp. 201-220, 2007.

[25] J. Brunel , 'YAPI: application modeling for signal processing systems', Proc. 37th Design Automation Conference, pp. 402-405, 2000.

[26] UPnP, 'The Universal Plug and Play Forum', http://www.upnp.org.

[27] W. Fontijn, J. Nesvadba, A. Sinitsyn, 'Integrating Media Management towards Ambient Intelligence', Journal Lecture Notes in Computer Science 'Title: Adaptive Multimedia Retrieval: User, Context, and Feedback', Springer-Verlag, ISSN: 1380-7501, vol. 3877 / 2006, pp. 102 - 111, 2006.

[28] A. Korostelev, J. Lukkien, J. Nesvadba, 'Error Detection in Service-Oriented Distributed Systems', Proc. Conf. on Dependable Systems and Networks, pp. 261-265, 2006.

[29] A. Korostelev, J. Lukkien, J. Nesvadba, Y. Qian, 'QoS Management in Distributed Service Oriented Systems', Proc. Conf. on Parallel and Distributed Computing and Networks, paper #551-071, ISBN: 978-0-88986-637-9, 2007.

[30] J. Nesvadba, et al., 'Real-Time and Distributed AV Content Analysis System for Consumer Electronics Networks', Proc. Int. Conf. for Multimedia and Expo, pp. 1549-1552, 2005.

[31] J. Nesvadba, et. al., ‘Method and Apparatus for Intelligent Channel Zapping’, Patent application EP1763953.

[32] F. Snijder, J. Nesvadba, ‘Similar content hopping’, Patent application US 2006/184963.

[33] A. Stella, J. Nesvadba, et. al., ’Silence detection’, European Patent 1,393,480, 2001.

[34] N. Dimitrova, J. Nesvadba, et. al., ‘Video content detection method and system leveraging data compression parameters’, US patent 6.714.594.

[35] E. Jaspers, J. Nesvadba, et.al, ‘CANDELA – Storage, Analysis and Retrieval of Video Content in Distributed Systems’, Book chapter in 'Adaptive Multimedia Retrieval: User, Context and Feedback', ISBN: 978-3-540-71544-3 pp 116 - 131, Springer, 2005.

[36] J. Nesvadba, N. Louis, J. Benois-Pineau, M. Desainte-Catherine, M. Klein Middelink, 'Low-level cross-media statistical approach for semantic partitioning of audio-visual content in a home multimedia environment', Proc. Workshop on Systems, Signals and Image Processing, pp. 235-238, 2004.

Chapter 3: State-of-the-art SBD

[37] N. Louis, ‘Indexation Cross-modale video / son des Contenus Multimedia Numerique’, PhD thesis, University of Bordeaux I, 2005.

[38] R. Brunelli, O. Mich, C. M. Modena, ‘A Survey on the Automatic Indexing of Video Data’, Journal of Visual Communication and Image Representation, Vol. 10 (3), pp. 78 – 112, 1999.

[39] R. Lienhart, ‘Comparison of Shot Boundary Detection Algorithms’, Storage and Retrieval for Still Image and Video Databases VII, Proc. SPIE, pp 290 – 301, 1999.

[40] I. Koprinska, S. Carrato, ‘Temporal video segmentation: a survey’, Signal Processing: Image Commun. 16, pp. 477-500, 2001.

[41] A. Hanjalic, ‘SBD: Unraveled and Resolved?’, Trans on Circuits and Systems for Video Technology (CSVT), v.12, N2, pp. 90-104, 2002.

[42] http://www-nlpir.nist.gov/projects/tv2004/tv2004.html

[43] N. Dimitrova, S. Jeannin, J. Nesvadba, et al., ‘Real-time commercial detection using MPEG features’, Proc. 9th Conf. on information processing and management of uncertainty in knowledge-based systems, pp. 481-486, 2002.

[44] M. Luo, D. DeMenthon, D. Doermann, ‘Shot Boundary Detection using Pixel-to-Neighbor Image Differences in Video’, TRECVid Workshop, 2004.

[45] T. Kikukawa, S. Kawafuchi, ‘Development of an automatic summary editing system for the audio-visual resources’, Trans, Electron. Inform. J75-A, pp 204-212, 1992.

[46] H. Zhang, A. Kankanhalli, S.W. Smoliar, ‘Automatic partitioning of full-motion video’, Multimedia Systems, vol. 1, pp. 10-28, 1993.

[47] B. Shahraray, 'Scene change detection and content-based sampling of video sequences', Proc. IS&T/SPIE, vol. 2419, pp. 2-13, 1995.

[48] B. Yeo, B. Liu, 'Rapid scene analysis on compressed video', Trans on Circuits Syst. Video Technology, vol. 5, pp. 533-544, 1995.

[49] F. Ernst, et. al., 'Dense structure-from-motion: an approach based on segment matching', Proc. ECCV, LNCS 2531, pp. II-217-II-231, Springer, 2002.

[50] U. Gargi, S. Oswald, D. Kosiba, S. Devadiga, R. Kasturi, 'Evaluation of video sequence indexing and hierarchical video indexing', Proc. Conf on Storage and Retrieval in Image and Video Databases, pp. 1522-1530, 1995.

[51] A. Nagasaka, Y. Tanaka, 'Automatic video indexing and full-video search for object appearances', Visual Database Systems II, E. Knuth, L.M. Wegner, Eds. Amsterdam, The Netherlands, pp 113-127, 1992.

[52] R. Zabih, J. Miller, K. Mai, 'Feature based algorithms for Detecting and Classifying Scene Breaks', Proc on ACM Conf. On Multimedia, pp. 189-200, 1995.

[53] Y. Yusoff, W. Christmas, J. Kittler, 'Video shot cut detection using Adaptive thresholding', Proc. British Machine Conference, pp. 68-74, 2000.

[54] J. Meng, Y. Juan, S.-F. Chang, 'Scene change detection in a MPEG compressed video sequence', Proc. IS&T/SPIE Symp. Electronic Imaging 2417, pp.14-25, 1995.

[55] N. Patel, I. K. Sethi, 'Video Shot Detection and Characterization for Video Databases', Proc. Pattern Recognition, vol. 30, pp. 583-592, 1997.

[56] L. Primaux, J. Benois-Pineau, P.Krämer, J-P.Domenger, 'Shot Boundary Detection In The Frameworkof Rough Indexing Paradigm', TRECVID'04 Workshop, 2004.

[57] R. Ruiloba, P. Joly, 'Framework for Evaluation of Video Shots Segmentation Algorithms – A Description Scheme for Editing Work', Journal of Networking and Information Systems, Hermes Science Publications, vol.3, no. 1/2000, pp. 77-103, 2001.

[58] W. Fernando, C. Canagarajah, D. Bull, ' Scene Change Detection Algorithms for Content-based Video Indexing and Retrieval', Journal Electronics & Communication Engineering, pp. 117-126, 2001.

[59] U. Naci, A. Hanjalic, 'Low Level Analysis of Video using Spatiotemporal Pixel Block', Proc Workshop on Multimedia Content Representation, Classification and Security, 2006.

[60] U. Naci, A. Hanjalic, 'TU Delft at TrecVid 2005: Shot Boundary Detection', Proc. Of TrecVid 2005, 2005.

[61] R. Ruiloba, P. Jolly, S. Marchand-Maillet, G. Quenot, 'Towards a Standard Protocol for the Evaluation of Video-to-Shots Segmentation Algorithms', Proc. Workshop on Content-based Multimedia Indexing, pp. 123-127, 1999.

[62] J. Corridoni, A Del Bimbo, 'Film Semantic Analysis', Proc. on Computer Architectures for Machine Perception, pp. 202-205, 1995.

[63] P. Joly, J. Benois-Pineau, et.al., 'The Argos Campaign: Evaluation of Video Analysis Tools', Proc. of Int. Workshop on Content-Based Multimedia Indexing, pp. 130-137, 2007.

Chapter 3: State-of-the-art Camera Motion

[64] D. Farin, 'An Automatic Video-object Segmentation System' PhD thesis, Eindhoven University of Technology, 2005.

[65] G. Hager, P.N. Belhumeur, 'Efficient Region Tracking with Parametric Models for Geometry and Illumination', Trans. on Pattern Analysis and Machine Intelligence, vol. 20, issue 10, pp. 1025 - 1039, 1998.

[66] M. Durik, J. Benois-Pineau, 'Robust Motion Characterisation for Video Indexing based on MPEG2 Optical Flow', Proc. of Content Based Multimedia Indexing, pp. 57-64, 2001.

[67] R. Hartley, A. Zisserman, 'Multiple-View Geometry in Computer Vision', Cambridge University Press, ISBN: 0521540518, 2000.

[68] M. Irani, P. Anandan, 'All about Direct Methods', Proc. Conf. on Computer Vision, pp. 626-633, 1999.

[69] P. Kraemer, J. Benois-Pineau, M. Garcia, 'Indexing Camera Motion Integrating Knowledge of Quality of Encoded Video', Proc. of Conf. on Sematic and Digital Media Technologies, pp. 78-83, 2006.

[70] P. Kraemer, J. Benois-Pineau, 'Camera Motion Detection in Rough Indexing Paradigm', Proc of TrecVid, 2005.

Chapter 3: State-of-the-art audio analysis and classification

[71] L. Rabiner, R. Schafer, 'Digital processing of speech signals', Prentice-Hall Inc., New Jersey, ISBN: 978-0132136037, 1978.

[72] K. Biatov, J. Koehler, 'An Audio Stream Classification and Optimal Segmentation for Multimedia Applications', Proc. Conf. ACM on Multimedia, vol. 3, pp. 37-40, 2003.

[73] S. Pfeiffer, 'Pause concepts for audio segmentation at different semantic levels', Proc. Conf. ACM on Multimedia, pp. 187 – 193, 2001.

[74] M. McKinney, J. Breebaart, 'Features for Audio and Music Classification', Proc. of Symposium on Music Information Retrieval, pp. 151-158, 2003.

[75] H. Kim, T. Sikora, 'Comparison of MPEG-7 audio spectrum projections and MFCC applied to speaker recognition, sound classification and audio segmentation', Proc. on Conf. on Acoustics, Speech, and Signal Processing, vol. 5, pp. 925-928, 2004.

[76] S. Pfeiffer, S. Fischer, W. Effelsberg, 'Automatic Audio Content Analysis', Proc. of 4th Conf on ACM Multimedia, pp. 21-30, 1997.

[77] Y. Li, C. Jay Kuo, ‘Video content analysis using multimodal information’, Kluwer Academic Publishers, ISBN: 978-1402074905, 2003.

[78] T. Zhang, C. Kuo, ‘Audio Content Analysis for Online Audiovisual Data Segmentation and Classification’, Transactions on Speech and Audio Processing, vol. 9, issue 4, pp. 441-457, 2001.

[79] N. Nitanda, M. Haseyama, H. Kitajima, ‘Audio-cut detection and audio-segment classification using fuzzy c-means clustering’, Proc. Conf. on Acoustics, Speech, and Signal Processing, vol. 4, pp. 325-328, 2004.

[80] H. Sundaram, S. Chang, ‘Audio scene segmentation using multiple features, models and time scales’, Proc. Conf. on Acoustics, Speech and Signal Processing, vol. 4, pp. 2441-2444, 2000.

[81] J. Foote, ‘Automatic audio segmentation using a measure of audio novelty’, Proc. Conf. on Multimedia and Expo, vol. 1, pp. 452 - 455, 2000.

[82] M. Cettolo, M. Vescovi, ‘Efficient audio segmentation algorithms based on the BIC’, Proc. Conf. on Acoustics, Speech, and Signal Processing, vol. 6, pp. 537-540, 2003.

Chapter 3: Video high-level features for segmentaion

[83] A. Hanjalic, R. Lagendijk, J. Biemond, 'Automated high-level movie segmentation for advancedvideo-retrieval systems', Transactions on Circuits and Systems for Video Technology, vol. 9, issue 4, pp. 580 - 588, 1999.

[84] J. Vendrig, M.Worring, 'Systematic evaluation of logical story unit segmentation', Proc. Transactions on Multimedia, vol. 4, Issue 4, pp. 492 - 499, 2002.

[85] J. Boggs, D. Petrie, ‘The Art of Watching Films’, McGraw-Hill Humanities, ISBN: 0-767-405-323, 2000.

[86] J. Benois-Pineau, W. Dupuy, D. Barba, ‘Outils de Structuration des Documents Video en vue d’Indexation basee sur une approche du Signal 1D’, technique et Science Informatique, vol. 22, no. 9/2003, pp. 1167-1200, Hermes-Lavoisier, 2003.

Chapter 3: State-of-the-art CBD

[87] D. Blum, ‘Method and Apparatus for Identifying and Eliminating Specific Material from Video Signals’, US patent 5,151,788, 1992.

[88] J. Iggulden, K. Fields, A. McFarland, J. Wu, ‘Method and Apparatus for Eliminating Television Commercial Messages’, US patent 5,696,866, 1997.

[89] R. Lienhart, C. Kuhmunch, W. Effelsberg, ‘On the Detection and Recognition of Television Commercials’, Proc. Conf. On Multimedia Computing Systems, pp. 509-516, 1997.

[90] S. Marlow, D. Sadlier, N. O’Connor, N. Murphy, ‘Automatic TV Advertisement Detection from MPEG Bitstream’, Journal of Patt. Rec. Society, vol. 35, no. 12, pp. 2-15, 2002.

[91] A. Albiol, María José Fullá, A. Albiol, L. Torres, 'Detection of TV commercials', Proc. Conf. on Acoustics, Speech and Signal Processing, vol. 3, pp. 541-544, 2004.

[92] A. Hanjalic, L. Xu, 'Affective Video Content Representation and Modeling', IEEE Transactions on Multimedia, Vol. 7, Issue 1, pp. 143 - 154, 2005.

[93] A. Hanjalic 'Adaptive Extraction of Highlights from a Sport Video based on Excitement Modeling', IEEE Transactions on Multimedia, Vol. 7, Issue 6, pp. 1114 - 1122, 2005.

Chapter 3: State-of-the-art ScBD

[94] J. Wang, T. Chua, 'A cinematic-based framework for scene boundary detection in video', Journal The Visual Computer, vol. 19, nr. 5, pp. 329-341, Springer, 2004.

[95] H. Kang, 'A Hierarchical Approach to Scene Segmentation', Proc. Workshop on Content-Based Access of Image and Video Libraries, pp. 65-71, 2001.

[96] Z. Rasheed, M. Shah, 'Scene Detection In Hollywood Movies and TV Shows', Proc. Conf. on Computer Vision and Pattern Recognition, vol. 2, pp. 343-348, 2003.

[97] Z. Rasheed, M. Shah, 'A Graph Theoretic Approach for Scene Detection in Produced Videos', Proc. Workshop on Multimedia Information Retrieval, pp. 145-150, 2003.

[98] J. Huang, Z. Liu, L. Wang, 'Integration of Audio and Visual Information for Content-Based Video Segmentation', Proc. Conf on Image Processing, vol. 3, pp. 526-529, 1998.

[99] Y. Zhu, D. Zhou, 'Scene Change Detection Based on Audio and Video Content Analysis', Proc. Conf. on Computational Intelligence and Multimedia Applications, pp. 229-234, 2003.

[100] S. Rho, E. Hwang, 'Video Scene Determination using Audiovisual Data Analysis', Proc. Conf. on Distributed Computing Systems Workshops, vol. 7, pp. 124-129, 2004.

[101] S. Chen, et al., 'Scene Change Detection by Audio and Video Clues', Proc. Conf.on Multimedia and Expo, vol. 2, pp. 365-368, 2002.

[102] R. Leonardi, P. Migliorati 'Semantic indexing of multimedia documents', IEEE Transactions on Multimedia, Vol. 9, Issue 2, pp. 44-51, 2002.

Chapter 4: Shot Boundary Detection

[103] J. Nesvadba, F. Ernst, J. Perhavc, J. Benois-Pineau, L. Primaux, 'Comparison of Shot Boundary Detectors', Proc. Int. Conf. for Multimedia and Expo, pp 788-791, 2005.

[104] TrecVid: http://www-nlpir.nist.gov/projects/trecvid/

[105] US Census Bureau: http://www.census.gov/ipc/www/idbpyr.html

[106] G. de Haan, et.al., 'True-Motion Estimation with 3-D Recursive Search Block Matching', Trans. on Circuits and Systems for Video Technology, Vol. 3, No. 5, pp. 368-379, 1993.

[107] A. Dommisse, 'Film detection for advanced scan rate converters', Philips Report 2002/418, 2002.

[108] F. Ernst, J. Nesvadba, 'Segmentation based Shot Boundary Detection', European Patent 1,597,914, 2003.

[109] B. Yeo, B. Liu, 'Rapid scene analysis on compressed video', Trans. on Circuits Syst. Video Technology, vol. 5, pp. 533-544, 1995.

[110] J. Boreczky, L. Roewe, 'Comparison of Video Shot Boundary Detection Techniques', SPIE Conf. Storage & Retrieval for Image & Video Databases, vol. 2670, pp. 170-179, 1996.

[111] C. Varekamp, S. Borchert, 'Pan/Zoom Motion Compensation for Frame Rate Up-conversion of Low Bitrate Video', Proc. Conf. On Image Processing, no.2257, pp. 104-108, 2005.

Chapter 4: Face Detection

[112] J. Nesvadba, P. Fonseca, et al., 'Face Related Features in Consumer Electronic (CE) device environments', Proc. Conf. on Systems, Man, and Cybernetics, pp 641-648, 2004.

[113] F. de Lange, J. Nesvadba, 'A Networked Hardware/Software Framework for the Rapid Prototyping of Multimedia Analysis Systems', Proc. Conf. on Web Information Systems and Technologies, pp. 312-318, 2005.

Chapter 4: Audio low-midlevel features

[114] MPEG-1 Layer II, ISO/IEC 11172-3, International standard Part 3: Audio, 1993.

[115] J. McKinney, R. Hopkins, 'Digital Audio Compression Standard AC-3', Advanced Television System Committee, Document A/52, 1995.

[116] A. Stella, J. Nesvadba, et.al., 'Estimating signal power in compressed audio', European Patent 1393301, 2001.

[117] J. Nesvadba, et al., 'Low-level cross-media statistical approach for semantic partitioning of audio-visual content in a home multimedia environment', Proc. Workshop on Systems, Signals and Image Processing, pp. 235-238, 2004.

Chapter 4: Commercial Block Detection

[118] J. Nesvadba, P. Fonseca, B. Kroon, 'Scrolling text detection and extraction by accurate estimation of scrolling text displacement', European patent application: NL008523, 2007.

[119] J. Nesvadba, F. Bruls, at.al., 'Detecting subtitles in a video signal', US patent 7.023.917, 2002.

[120] S. Forbes, F. Forbes, 'Video Signal Identifier for Controlling a VCR and Television based on The Occurance of Commercials', US patent 5,708,477, 1997.

[121] J. Nesvadba, F. Snijder, M. Barbieri, A. Stella, 'Content analysis apparatus', European Patent 1,436,370, 2001.

[122] N. Dimitrova, T. McGee, J. Nesvadba, et al., 'Video content detection method and system leveraging data compression parameters', US 6,714,594, 2001.

[123] N. Dimitrova, S. Jeannin, J. Nesvadba, T. McGee, L. Agnihotri, G. Mekenkamp, 'Real-Time Commercial Detection Using MPEG Features', Proc. Conf. On Information Processing and Management of Uncertainty in knowledge-based systems, pp. 481-486, 2002.

[124] L. Agnihotri, N. Dimitrova, T. McGee, S. Jeannin, D. Schaffer, J. Nesvadba, 'Evolvable Visual Commercial Detector', Proc. Conf. on Computer Vision and Pattern Recognition, Vol 2, pp. II-79 – II-84, ISSN: 1063-6919, 2003.

Chapter 4: Film Grammar and Parallel Shot Detection

[125] S. Field, 'The Screenwriter's Workbook', ISBN: 0-385-339-046, 2006.

[126] S. Jeong, R. Gray, C. Sun Wong, 'Histogram-based Image Retrieval Using Gaussian Mixture Vector Qnatization", Proc. Conf of Acoustic, Speech and Signal Processing, vol. 3 (3): 677-680, 2003.

[127] P. Kuhn, 'Camera Motion Estimation using Feature Points in MPEG Compressed Domain', Proc. Conf. On Image Processing, pp. 596-599, vol. 3, 2000.

[128] K. Matsuyama, H. Namada, 'Segmentation of Multiple Objects based on Orthogonal Projection and Camera Motion', Proc. Workshop on Nonlinear Circuits and Signal Processing, pp.179-182, 2005.

[129] N. Sebe, et al, 'Salient Points for Content Based Retrieval', Proc. on Computer Vision and Pattern Recognition, pp. 401-410, 2001.

[130] C. Harris, M. Stephens, 'A Combined Corner and Edge Detector', Proc. 4th Alvey Vision Conference, Manchester, pp 147-151, 1988.

[131] J. Shi, C. Tomasi, 'Good Features to Track', Proc. Conf. on Computer Vision and Pattern Recognition, pp. 593-600, 1994.

[132] P. Anandan, 'A Computational Framework and an Algorithm for the Measurement of Visual Motion', Journal of Computer Vision, vol. 2, nr. 3, pp. 283-31-, 2004.

[133] I. Patras, E. Hancock, 'Regression tracking with Data Relevance Determination', Proc. of Computer Vision and Pattern Recognition, pp. 1-8, 2007.

[134] M. Brown, D. Lowe, 'Automatic Panoramic Image Stichting using Invariant Features', Journal of Computer Vision, vol. 74, nr. 1, pp. 59-73, 2007.

[135] D. Lowe, 'Distinctive Image Features From Scale-Invariant Keypoints', Journal of Computer Vision, vol. 20, pp. 91-110, 2004.

[136] J. Nesvadba, et. al., 'Parallel Shot Detector', Patent application NL004360.

[137] B. Kroon, J. Nesvadba, A. Hanjalic, 'Dialog Detection in Narrative Video by Shot and Face Analysis', Proc. IS&T/SPIE Electronic Imaging, Vol. 6506, 2007.

Chapter 4: AV ScBD with audio classes

[138] E. Mamdani, S. Assilian, 'An experiment in linguistic synthesis with a fuzzy logic controller,' Journal of Man-Machine Studies, vol. 7, no. 1, pp. 1-13, 1975.

[139] T. Seng, M. Khalid, R. Yusof, S. Omatu, 'Adaptive Neuro-fuzzy Control System by RBF and GRNN Neural Networks', J. of Intelligent and Robotic Systems, vol. 23, pp. 267-289, 1998.

Chapter 4: AV ScBD with shot Length analysis

[140] W. Faulstich, H. Korte, 'Filmanalyse Interdisziplinaer', Göttingen: Vandenhoeck & Ruprecht, pp. 73-93 (Zeitschrift für Literaturwiyssenschaft und Linguistik. Beihefte. 15.), 1987.

[141] W. Faulstich, 'Film aesthetics and new methods of film analysis', *Empirical Studies of the Arts* 7,2, pp. 175-190, 1989.

[142] W. Weibull, 'A statistical distribution function of wide applicability', *J. Appl. Mech.-Trans. ASME* 18(3), 293-297, 1951.

ANNEX 1 MPEG2

[143] MPEG-1 ISO/IEC standard: ISO/IEC 11172-1:1993; http://www.iso.org/iso/en/

[144] MPEG-2 ISO/IEC standard: ISO/IEC 13818-1:2000; http://www.iso.org/iso/en/

Annex SIFT

[145] H. Moravec, 'Rover Visual Obstacle Avoidance', Proc. Conf. On Artificial Intelligence, pp. 785-790, 1981.

[146] C. Harris, M. Stephens, 'A combined Corner and Edge Detector', Proc. Ivey Vision Conference, pp. 147-151, 1988.

[147] A. Witkin, ' Scale-space Filtering', Proc. Conf. On Artificial Intelligence, pp. 1019-1022, 1983.

[148] J. Koenderink, ' The Structure of Images', Proc. Conf on Biological Cybernetics, vol. 50, pp. 363-396, 1984.

[149] T. Lindenberg, 'Scale-space Theory: A basic Tool for analyzing structures at different Scales', Journal of Applied Statistics, Vol. 21(2), pp.224-270, 1994.

Annex MPEG-7

[150] ISO/IEC 15938-2. Information Technology – Multimedia Content Description Interface – Part 2: Description Definition Language, September 2001.

[151] ISO/IEC 15938-3. Information Technology – Multimedia Content Description Interface – Part 3: Visual, September 2001.

List of the author's publications

Book Chapters

- R. Jasinschi, J. Nesvadba, 'Ambient Vision: Interface, Adaptation, and Management', Book: 'The new everyday', page: 72-77, Publisher: 0I0, ISBN '978 90 6450 502 7', The Netherlands, 2002.

Journals and Magazines:

- W. Fontijn, J. Nesvadba, A. Sinitsyn, 'Integrating Media Management towards Ambient Intelligence', Journal Lecture Notes in Computer Science, Title: 'Adaptive Multimedia Retrieval: User, Context, and Feedback', Springer-Verlag, ISBN: '978-3-540-71544-3', Vol. 3877 / 2006, pp. 102 - 111, 2006.
- J. Nesvadba, F de Lange, 'Early Evaluation of Future Consumer AV Content Analysis Applications with PC networks', Journal Multimedia Tools and Applications, vol 34, nr.2, pp. 201-220, ISSN 1380-7501, Springer, 2007.
- P. Fonseca, J. Nesvadba, 'Face Tracking in the Compressed Domain', EURASIP Journal on Applied Signal Processing, Vol. 2006, Article ID 59451, pp. 1-11, Hindawi, DOI: 10.1155/ASP/2006/59451, 2006.

Conference and Workshops papers:

- J. Nesvadba, F.de Lange, ' Cassandra Framework: A Service Oriented Distributed Multimedia Content Analysis Engine', Proc. of Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS 2007), pp. 81-85, DOI: '10.1109/WIAMIS.2007.29', Greece, 2007.
- B. Kroon, J. Nesvadba, A. Hanjalic, ' 'Dialog Detection in Narrative Video by Shot and Face Analysis'', Proc. of Advanced School for Computing and Imaging (ASCI 2007), vol. 1, pp. 203-213, The Netherlands, 2007.
- A. Korostelev, J. Lukkien, J. Nesvadba, Y. Qian, 'QoS Management in Distributed Service Oriented Systems', Proc. of IASTED Int. Conf. on Parallel and Distributed Computing and Networks (PDCN 2007), ISBN: '978-0-88986-639-3', vol. 551, pp. 71-75, Austria, 2007.
- B. Kroon, J. Nesvadba, A. Hanjalic, 'Dialog Detection in Narrative Video by Shot and Face Analysis', Proc. of IEEE Int. Conf. on Consumer Electronics (IS&T/SPIE Electronic Imaging 2007: Multimedia Content Analysis), vol. 6506, pp. 315-325, USA, 2007.

- A. Korostelev, J. Lukkien, J. Nesvadba, 'Error Detection in Service-Oriented Distributed Systems', Proc. of IEEE Int. Conf. on Dependable Systems and Networks (DSN 2006), vol. 2, pp. 278-282, USA, 2006.
- J. Nesvadba, F. d. Lange, A. Sinitsyn, J. Lukkien, A. Korostelev, 'Distributed and Adaptive Multimedia Content Analysis Prototyping Framework for Consumer Electronics', Invited Paper, Proc. of IEEE Int. Conf. on Consumer Electronics (ICCE 2005), vol. 1, pp. 473- 474, ISBN: '0-7803-9459-3', USA, 2006.
- A. Hanjalic, J. Nesvadba, J. Benois-Pineau, 'Moving away from narrow-scope Solutions in Multimedia Content Analysis', Proc. of European Workshop on the Integration of Knowledge, Semantic and Digital Media Technologies, vol. 1 pp. 1-6, ISBN: '0-86341-595-4', UK, 2005.
- E. Jaspers, R. Wijnhoven, R. Albers, J. Nesvadba, A. Sinitsyn, J. Lukkien, X. Desuremont, P. Pietarila, R. Truyen, J. Palo, 'CANDELA – Storage, Analysis and Retrieval of Video Content in Distributed Systems', Proc. of Int. Workshop on Adaptive Multimedia Retrieval (AMR 2005), Journal of Computer Science "Adaptive Multimedia Retrieval: User, Context and Feedback", Springer, Vol. 3877, pp. 112-127, ISBN: '978-3-540-32174-3', UK, 2005.
- W. Fontijn, J. Nesvadba, A. Sinitsyn, 'Integrating Media Management towards Ambient Intelligence', Proc. of Int. Workshop on Adaptive Multimedia Retrieval (AMR 2005), Journal of Computer Science "Adaptive Multimedia Retrieval: User, Context and Feedback", Springer, Vol. 3877, pp. 102-111, ISBN: '978-3-540-32174-3', UK, 2005.
- J. Nesvadba, A. Hanjalic, P. Fonseca, B. Kroon, H. Celik, E. Hendriks, 'Towards a real-time and distributed system for face detection, pose estimation and face-related features', Invited Paper, Proc. Int. Con. on Methods and Techniques in Behavioral Research, vol. 1, pp. 3-7, ISBN: '90-74821-71-5', The Netherlands, 2005.
- J. Nesvadba, et al., 'Real-Time and Distributed AV Content Analysis System for Consumer Electronics Networks', Proc. of Int. Conf. for Multimedia and Expo (ICME 2005), vol. 1, pp. 1549-1552, ISBN: '0-7803-9331-7', The Netherlands, 2005.
- J. Nesvadba, F. Ernst, J. Perhavc, J. Benois-Pineau, L. Primaux, 'Comparison of Shot Boundary Detectors', Proc. of Int. Conf. for Multimedia and Expo (ICME 2005), vol. 1, pp 788-792, ISBN: '0-7803-9331-7', The Netherlands, 2005.
- F. de Lange, J. Nesvadba, 'Rapid Prototyping of Multimedia Analysis Systems - A Networked Hardware/Software Framework', Proc. Int. Conf. On Web Information

Systems and Technologies (WEBIST 2005), pp. 104-109, ISBN: '972-8865-20-1', USA, 2005.

- F de Lange, J. Nesvadba, 'Applying PC network technology to assess new multimedia content analysis applications for future consumer electronics storage devices', Proc. of 4th Int. Conf. On Intelligent Multimedia Computing and Networking (IMMCN), USA, 2005.
- F de Lange, J. Nesvadba, 'Applying PC technology to assess new AV content analysis applications for future CE storage products', Philips Software conference 2005, The Netherlands, 2005.
- J. Nesvadba, et al., 'CANDELA deliverable D1.1B State-of-the-art report, Annex', European Report, Deliverable D1.1B, 2004.
- J. Nesvadba, N. Louis, J. Benois-Pineau, M. Desainte-Catherine, M. Klein Middelink, 'Low-level cross-media statistical approach for semantic partitioning of audio-visual content in a home multimedia environment', Proc. IEEE Int. Workshop on Systems, Signals and Image Processing (IWSSIP'04), pp. 235-238, Poland, 2004.
- J. Nesvadba, P. M. Fonseca, R. Kleihorst, H. Broers, J. Fan, 'Face Related Features in Consumer Electronic (CE) device environments', Invited paper, Proc. IEEE Int'l Conf. on Systems, Man, and Cybernetics (IEEE SMC 2004), Special Session on Automatic Facial Expression Recognition, vol. 1, pp 641-648, ISBN: '0-7803-8566-7', Netherlands, 2004.
- P. Miguel Fonseca, J. Nesvadba, 'Face Detection in the Compressed Domain', Proc. of IEEE Int. Conf. on Image Processing (IEEE ICIP 2004), vol. 3, pp.2015-2018, ISBN: '0-7803-8554-3', Singapore, 2004.
- J. Nesvadba, J. Perhavc, F. Ernst, A. Dommisse, 'Comparison of Shot Boundary Detectors', Proc. of Conf. SOIA, pp. 165-177, The Netherlands, 2004.
- L. Agnihotri, N. Dimitrova, T. McGee, S. Jeannin, D. Schaffer, J. Nesvadba, 'Evolvable Visual Commercial Detector', Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, vol. 2, pp. 79-84, ISBN: '0-7695-1900-8', USA, 2003.
- N. Dimitrova, S. Jeannin, J. Nesvadba, T. McGee, L. Agnihotri, G. Mekenkamp, 'Real-Time Commercial Detection Using MPEG Features', Invited paper, Proc. 9th Int. Conf. On Information Processing and Management of Uncertainty in knowledge-based systems (IPMU 2002), pp. 481-486, France.
- M. Barbieri, M. Ceccarelli, G. Mekenkamp, J. Nesvadba, 'A Personal TV Receiver with Storage and Retrieval Capabilities', User Modeling - Proc. of Workshop on

personalization in future TV, 8th Conference on User Modeling (UM2005), Germany, 2001.

- M. Barbieri, G. Mekenkamp, M. Ceccarelli, J. Nesvadba, 'The Color Browser: A content driven linear browsing tool', Proc. Conf. on Multimedia and Expo (IEEE ICME2001), pp. 627-630, Japan, 2001.

Presentations and exhibitions

- J. Nesvadba, Invited speaker: 'Multimedia content analysis for CE' IST 2006: Knowledge in Multimedia Content, IST 2006: Knowledge in Multimedia Content, Helsinki, Finland, November 20-23, 2006.
- J. Nesvadba, Invited lecture: 'Semantic Multimedia Analysis', Summer School on Multimedia Semantics - Analysis, Annotation, Retrieval and Applications (SSMS'06), Chalkidiki, Thessaloniki, Greece, September 4-8, 2006.
- J. Nesvadba, D. Farin, Tutorial: 'Content Analysis for Multimedia Applications and Home Servers', IEEE Int. Conf On Consumer Electronics, Las Vegas, USA, January 7-11, 2006.
- J. Nesvadba, 'Towards Distributed and Generic Multimedia Content Analysis Solutions', Invited Speaker, European Workshop on the Integration of Knowledge, Semantic and Digital Media Technologies, London, UK, November 30 - December 1, 2005.
- J. Nesvadba, Open Innovation Day, 2005, High Tech Campus Eindhoven, The Netherlands, 2005.
- J. Nesvadba, Measuring Behaviour 2005, Special Symposium: 'Automatic Facial Expression Analysis and Synthesis' , Wageningen, The Netherlands, 2005.
- J. Nesvadba, et.al., 'CASSANDRA Framework - distributed content analysis demonstrator', 6rd International conference on Multimedia and Expo (IEEE ICME 2005), Amsterdam, The Netherlands, 2005.
- J. Nesvadba, MultimediaN Kick-off meeting October 2005, 'The Future of Multimedia', Utrecht, The Netherlands, 2005.
- J. Nesvadba, 'Multimedia Content Analysis in Candela', ITEA Candela Seminar, Oulu, Finnland, September 9, 2004.
- J. Nesvadba, 'Content Management in Consumer Environment', MediaMill workshop, January 30, 2003.
- J. Nesvadba, Philips press conference, Tokyo, Japan, 2003.
- J. Nesvadba, 'CASSANDRA demonstrator', 3rd International conference on Multimedia and Expo (IEEE ICME 2002), Lausanne, Switzerland, 2002.

- J. Nesvadba, ‘Personal TV receiver with storage and retrieval capabilities’, UM2001 (International conference on User Modeling), Sonthofen, Germany, 2001.
- J. Nesvadba, 'AVIR demonstrator', 1st International conference on Multimedia and Expo (IEEE ICME 2000), New York, USA, 2000.

Granted US Patents

- US 7.023.917, ‘Detecting subtitles in a video signal’, Jan Nesvadba, Fons Bruls, Gert Vervoort, Bernhard Penz
- US 7.243.111, ‘Content Analysis Apparatus’, Jan Nesvadba, F. Snijder, M. Barbieri
- US 6.180.144, ‘Detecting a cartoon in a video data stream’, Radu Jasinschi, Jan Nesvadba, Thomas McGee, Lalitha Agnihotri
- US 2002/186768 (EP1393569), ‘Video content detection method and system leveraging data compression parameters’, N. Dimitrova, T. McGee, J. Nesvadba, et al.

(Pending) Patents with EU Patent number

- EP 1.247.380, ’Digital transmission system having disparity dependent channel code words’, J. Kahlman, A. Jansen van Doorn, J. Nesvadba
- EP 1.393.301, ‘Estimating signal power in compressed audio’, Alessio Stella, Jan Nesvadba, Mauro Barbieri, Freddy Snijder
- EP 1.393.480, ’Silence detection’, Alessio Stella, Jan Nesvadba, Mauro Barbieri, Freddy Snijder
- EP 1.537.689, ‘Method of content identification device and software’, F. Snijder, J. Nesvadba
- EP 1.579.451, ’Creating Edit Effects on MPEG-2 compressed Video’, D. Kelly, J. Nesvadba, J. v. Gassel
- EP 1.584.048, ‘Similar content hopping’, F. Snijder, J. Nesvadba, M. Barbieri
- EP 1.597.914, ‘Shot-Cut Detection’, F. Ernst, J. Nesvadba
- EP 1.606.817, ’Device and Method for Recording Information’, J. Nesvadba, D. Kelly, I. Nagorski
- EP 1.616.272, ’System and Method for Performing Automatic Dubbing on an Audio-Visual Stream', J. Nesvadba, D.J. Breebaart, M. McKinney
- EP 1.618.743, ’Content Analysis of Coded Video Data’, D. Burazerovic, J. Nesvadba, F.Snijder
- EP 1.680.785, ‘Recording Content on a Record Medium that contains desired Content Descriptors’, E. Thelen, D. Klakov, C. Luijks, J. Nesvadba

- EP 1.763.953, ‘Method and Apparatus for Intelligent Channel Zapping’, J. Nesvadba, I. Nagorski
- EP 1.330.826, ‘Reproducting apparatus providing a colored slider bar’, M. Barbieri, J. Nesvadba, G.E. Mekenkamp, M.P. Ceccarelli, W.F.J. Fontijn
- WO 2004/109549 A3, ‘System and Method for Performing Media Content Augmentation on Audio Signal’, M. McKinney,J. Nesvadba, D.J. Breebaart
- EP 1.815.621, ‘Apparatus for analyzing Audio Content and Reproducing desired Audio Data’, F de Lange, J. Nesvadba, I. Nagorski
- EP 1.779.659, ‘Selection of Content from a Stream of Video or Audio Data', J. Nesvadba, I. Nagorski, R. Aarts

Pending Patents

- WO 2005/041455, ‘Video Content Detection’, J. v.Gassel, D. Kelly, J.Nesvadba
- WO 2006/077533, ‘Apparatus for analyzing a content stream comprising a content item’, J. Nesvadba, D. Burazerovic,
- WO 2006/103625, 'Method and Apparatus for the Detection of Text in Video Data', J. Nesvadba, I. Nagorski
- WO 2007/036843, ‘Method and Apparatus for Retrieving a Text Associated with an Object or Subject’, G. Hollemans, J. Nesvadba
- WO 2007/039871, ‘A Device for Handling Data Items that can be Rendered to a User’, J. Nesvadba, D. Burazerovic, C. Mol
- NL021404, ‘Identifying video content using audio fingerprint’, J. v.Gassel, D. Kelly, J. Nesvadba
- NL030386, ‘Interoperable content analysis of standard block-based video’, D. Burazerovic, J. Nesvadba, F. Snijder
- NL000886, ‘Extensions to EPG Boundary Enhancer’, J. Nesvadba, D. Burazerovic
- WO 2007/072347, ‘System and Method for Processing Video: Parallel Shot Detector’, J. Nesvadba, Y.S. Joshi, S. Pfundtner
- WO 2007/093932, ‘A Device for and a Method of Managing Auxiliary Data Assigned to main Data’, W. Fontijn, A. Sinitsyn, A. Kobzhev, J. Nesvadba
- NL004957, ‘Manual Post Annotation of AV Content', J. Nesvadba, P. Fonseca
- NL005782, ‘Speedy Access by Automatic Database Reduction', A. Sinitsyn, W. Fontijn, J. Nesvadba
- NL006513, ‘Semantic information retrieval of videos of public using histogram-based foreground object analysis', J. Nesvadba, H. Celik, A. Hanjalic

- NL007596, ‘Method and apparatus for Smoothing a Transition between first and second video Segment', D. Buracerovic, P. Fonseca, J. Nesvadba
- NL008523, ‘Scrolling text detection and extraction by accurate estimation of scrolling text displacement', J. Nesvadba, P. Fonseca, B. Kroon

Segmentation Sémantique des Contenus Audio-Visuels

Résumé:

Dans ce travail, nous avons mis au point une méthode de segmentation des contenus audiovisuels applicable aux appareils de stockage domestiques pour cela nous avons expérimenté un système distribué pour l'analyse du contenu composé de modules individuels d'analyse : les *Service Unit*. L'un d'entre eux a été dédié à la caractérisation des éléments hors contenu, i.e. les publicités, et offre de bonnes performances. Parallèlement, nous avons testé différents détecteurs de changement de plans afin de retenir le meilleur d'entre eux pour la suite. Puis, nous avons proposé une étude des règles de production des films, i.e. grammaire de films, qui a permis de définir les séquences de *Parallel Shot*. Nous avons, ainsi, testé quatre méthodes de regroupement basées similarité afin de retenir la meilleure d'entre elles pour la suite. Finalement, nous avons recherché différentes méthodes de détection des frontières de scènes et avons obtenu les meilleurs résultats en combinant une méthode basée couleur avec un critère de longueur de plan. Ce dernier offre des performances justifiant son intégration dans les appareils de stockage grand public.

Discipline: Informatique

Mots-clés: indexation multimédia, segmentation temporelle des flux multimédias, classification multimédias, analyse des contenus audiovisuels numériques, analyse sémantique des scènes

Semantic Segmentation of Audiovisual Content

Abstract:

In this work we elaborated a method for semantic segmentation of audiovisual content applicable for consumer electronics storage devices. For the specific solution we researched first a service-oriented distributed multimedia content analysis framework composed of individual content analysis modules, i.e. *Service Units*. One of the latter was dedicated to identify non-content related inserts, i.e. commercials blocks, which reached high performance results. In a subsequent step we researched and benchmarked various *Shot Boundary Detectors* and implement the best performing one as *Service Unit*. Here after, our study of production rules, i.e. film grammar, provided insights of *Parallel Shot* sequences, i.e. *Cross-Cuttings* and *Shot-Reverse-Shots*. We researched and benchmarked four similarity-based clustering methods, two colour- and two feature-point-based ones, in order to retain the best one for our final solution. Finally, we researched several audiovisual *Scene Boundary Detector* methods and achieved best results combining a colour-based method with a shot length based criteria. This *Scene Boundary Detector* identified semantic scene boundaries with a robustness of 66% for movies and 80% for series, which proofed to be sufficient for our envisioned application *Advanced Content Navigation*.

Discipline: Computer Science

Keywords: multimedia indexing, temporal segmentation of multimedia streams, multimedia classification, digital audiovisual content analysis, semantic scene analysis

LaBRI - Université Bordeaux 1,
351, cours de la libération
33405 Talence Cedex (France)