The SVP Group‡

Automatic Generation of Movie Trailers using Ontologies

Abstract

With the advances in digital audio and video analysis, automatic movie summarization has become an important field of research. Much of the work has been put into movie abstracting for large media databases. Looking at the topic from a different side, the movie industry has long since perfected the art of summarization in their advertising trailers to attract an audience. In this paper we introduce the approach of automatically generating entertaining Hollywood-like trailers based on a trailer grammar, enhanced by an ontology. The extraction of features from movies using state-of-the-art image and audio processing techniques builds the foundation for the selection of meaningful and usable material, which is re-assembled according to the defined rules. User testing of our automatically produced trailers shows that they are well accepted and in many ways comparable to professionally composed trailers.

1 Introduction

Although originally intended to advertise a certain movie, the short preview of a movie, i.e., trailer or teaser, has become an attractive movie genre in itself [Kernan 2004, Arijon 1991], especially since many trailers are available on the Internet. With the development of current digital technology the question arises if and to what extent it is feasible to automate the process of trailer production based solely on a high-level analysis of the original movie. Such a system could provide improvements in different movie-related fields. For example, it could suggest innovative ways of video browsing in digital movie databases in a way that those trailers could serve as a compact overview of a certain movie or to gain more control especially in movie-on-demand system through movie indexing, retrieval, and browsing

‡ *The SVP-Group is in alphabetical order: Christoph Brachmann, Hashim I. Chunpir, Silke Gennies, Benjamin Haller, Thorsten Hermes, Otthein Herzog, Arne Jacobs, Philipp Kehl, Astrid P. Mochtarram, Daniel Möhlmann, Christian Schrumpf, Christopher Schultz, Björn Stolper, and Benjamin Walther-Franks.*

(see [Chen *et al.* 2004]). Furthermore, it could help with developing and testing of experiments to formalize existing movie editing methods (film theory), and simplify or even extend the work of editors/authors (see [Snoek & Worring 2005]).

In this paper, an automatic trailer generation system is introduced which covers three research challenges. First, a formalism is proposed that describes the basic components of a trailer. Second, video content analysis methods are presented that provide these single components. Finally, a methodology is shown that selects and composes the components according to the given formalism.

This paper is organized as follows. In section 2 a brief overview about previous work related to automatic trailer generation is given. Section 3 describes the formalism that we developed and use as a basis for our system. This approach is based on an ontology, and according to [Chen *et al.* 2004] one could say it is like a grammar for an automatic trailer generation, but we consider our approach as rule-based. Section 4 illustrates our system framework that – based on the given set of rules – is capable of analyzing a movie first, and then is able to generate trailers of this analyzed and annotated movie. In section 5 we present the application of our system to generate trailers for current Hollywood action movies along with an evaluation of the corresponding output in section 6. Finally, section 7 draws a conclusion and addresses possible aspects of future work.

2 Related Work

In this section we provide a very brief overview of touching approaches since the specific way of generating movie trailers has only little related work so far. One can say that area of automatic trailer generation is a rather untouched field of research. However, the more general task of summarizing video content is a wide field of research.

The works of [Chen *et al.* 2004] and [Lienhart *et al.* 1997] come closest to our goal. Both mention the possibility of generating a movie trailer explicitly. And furthermore, both point out to do the composition of footage according to rules derived from film theory and present ways to retrieve crucial information for trailer generation. But they do not focus on how to compose trailers. Only [Chen *et al.* 2004] uses the definition of *tempo* in order to generate action trailers. Although the film theory is valuable in the analysis of the footage deriving high-level features from low level features, it is not completely applicable for the generation of a movie trailer.

Since a movie trailer is a kind of abstract of a movie other works within the field of video abstracting or summarization rather focus on the task of pure summarizing in order to provide means to handle the increasing amount of video data. One could find three basic approaches. The first one is video skimming as, e.g., in [Christel *et al.* 1999] or [Smith & Ka-

nade 1998] where video material is analyzed and condensed to important scenes. Typically the linearity of the input video is preserved. The second basic approach is summarizing contents in a pictorial way [Uchihashi *et al.* 1999, Yeung & Yeo 1997]. In [Uchihashi *et al.* 1999] in a first step salient single frames of video sequences are captured. In a second step these frames are sized according to their importance, and finally arranged in a third step in a linear comic-like, story-telling way. The third video browsing approach is closely related to the pictorial summarization but focuses on a hierarchical, not necessarily linear way of presenting the video content [Ponceleon & Dieberger 2001, Zhang *et al.* 1993].

The degree of automation varies. Completely automatic approaches are [Lienhart *et al.* 1997, Smith & Kanade 1998, Uchihashi *et al.* 1999]. Typical for these approaches of automatic summaries is the high dependency on low level analysis of image and audio. A so-called semi-automatic approach can be found in [Zhu *et al.* 2003]. The semi-automatic summary tools provide some manual annotation framework enabling high-level analysis to conclude what is happening in a scene. Another interesting work is [Ma *et al.* 2002], focusing on the question of how a video is perceived by a user.

Finally, while some works – [Lienhart *et al.* 1997, Smith & Kanade 1998, Zhu *et al.* 2003] – can be applied to a wide variety of footage, others focus on a specific type of video data, e.g., sports [Babaguchi *et al.* 2005].

An extensive overview of “State-of-the-Art” video indexing from the author’s point of view can be found in [Snoek & Worring 2005].

3 Formalism

In the scope of re-assembling movie footage in a short video that can be labeled as a *trailer*, first the meaning of this label has to be understood. According to [Arijon 1991] films are created based on an underlying *Film Grammar* to successfully communicate with the audience. In [Kernan 2004] it is stated that a trailer is also a movie genre of its own right. Therefore we assume that trailers - being a special kind of film – can be described by syntactic elements and semantic rules. One will say that this constitutes a *trailer grammar* but we will stay to a rule-based system. To implement an automatic trailer generator these rules have to be understood and modeled in a way that a computer can execute generative algorithms according to them. The problem therefore demands understanding, extracting, and formalizing two items: the trailer’s syntactic elements (section 3.2), and its semantic rules (3.3). Before looking at these, we give a definition of the term ‘trailer’ within the context of automatic generation.

3.1 The Definition of a Trailer in Respect to Automatic Generation

[Kernan 2004] points out that the name *trailer* refers to the fact that these short movies were originally shown at the end of a film program in movie theaters. But nowadays trailers were normally shown before the main movie. During the 20th century, trailers evolved from a mere advertisement to a movie genre with its own unique conventions, based on the demand to combine an artistic form with the highly commercial need of drawing the biggest possible audience into the theaters by presenting every movie in the most attractive light.

While movies and trailers exist in many different forms according to different cultural environments, our automatic approach is based on the Western culture's most dominant trailer and movie industry: Hollywood blockbuster cinema. Although trailers from this domain have developed a general formula that pays as little attention to genre or specific target groups as possible – to attract literally everybody and lead to an "undifferentiation of audiences" – they still have to reflect the movie in question to a certain extent [Kernan 2004]. Furthermore, our aim is to produce short videos that resemble rather conventional *Theatrical Trailers* by having a length of more than one minute and featuring footage from the original movie. These are opposed to so-called *Teaser Trailers*, which are typically produced before primary shooting is finished and consist mostly of texts, voice-overs, and graphic elements, and which have a maximum running length of one minute. In the following the term *trailer* therefore will be used referring to a theatrical trailer for a contemporary Hollywood movie.

3.2 The Syntactic Elements of a Trailer

The basic elements of any edited movie are usually shots and transitions. We assume that within these elements certain types of shots and transitions can be identified by a shot-by-shot analysis of original movie trailers. In order to determine these types, an appropriate set of descriptions, i.e., an appropriate vocabulary, has to be defined. Since our goal is to implement a completely automatic system for trailer generation, we consider the restrictions imposed by the technical feasibility when setting up such a vocabulary. This inevitably causes a quality loss but cannot be avoided in our case. Furthermore, there is a trade-off concerning the level of detail when defining the appropriate descriptions for shots. If the detail is too high, the shot descriptions are only suitable for a very specific situation in one trailer. On the other hand, if the level of detail is too low, they are too general and have very little meaning. That is why the resulting types of shots have to:

a) be able to cover all shots of a trailer,
b) be clearly distinguishable from each other (no redundancy),

c) have a well-defined meaning,
d) apply to as many existing trailers as possible, and
e) be defined based on the information which will be extracted from the movie by our automatic analysis tools (technical feasibility).

Besides this, describing the transitions is easier since they follow the known conventions of the film grammar. Well-known transitions are for example *hard-cuts*, *fade-ins*, and *fade-outs*.

In order to distinguish between the original movie and trailer shots, and the shots we produce for our trailers we refer to the latter ones as *clips*, and in order to fulfill the requirements listed above we define the types of the *clips* by the following *properties*:

- a **category** (reflecting the shot's formal features),
- the playback **speed** (to model effects like slow-motion or acceleration),
- the **volume** of the original footage sound (so that clips can be muted or amplified),
- and **location**, corresponding to the footage location in the source movie.

3.3 The Semantic Elements of a Trailer

Once *clips* and *transitions* are identified and described as syntactic elements of a trailer, semantic rules are needed to assemble these elements in a trailer-like way. We propose to represent these rules as a hierarchy of super- and sub-patterns as shown in Figure 1. Each super-pattern consists of a number of sub-patterns either in a certain order or as a random choice.

The highest level of patterns is the *Trailer Pattern*. Since there is not only one universal pattern that can describe all trailers at once, this pattern can be used to distinguish between different types of trailers. For example, one Trailer Pattern could stand for action movie trailers, and another Trailer Pattern could stand for a romantic movie trailer.

In our model every trailer can be subdivided into a number of different narrative blocks, which we call *Phases*. These *Phase Patterns* could one of the five following phases we identified in contemporary trailers: *Intro*, *Story*, *Break*, *Action*, and *Outro* (see also section 5.1).

The Phase Patterns again are composed of *Sequence Patterns*, which in turn consist of a number of *Clip/Transition Pairs*. These Pairs are the lowest level of the hierarchy. Therefore, a trailer is described by a linear list of clips joined by transitions. The intention behind the modeling of a trailer in such a way is to represent the trailer grammar as precise as possible, while preserving the highest possible amount of flexibility.

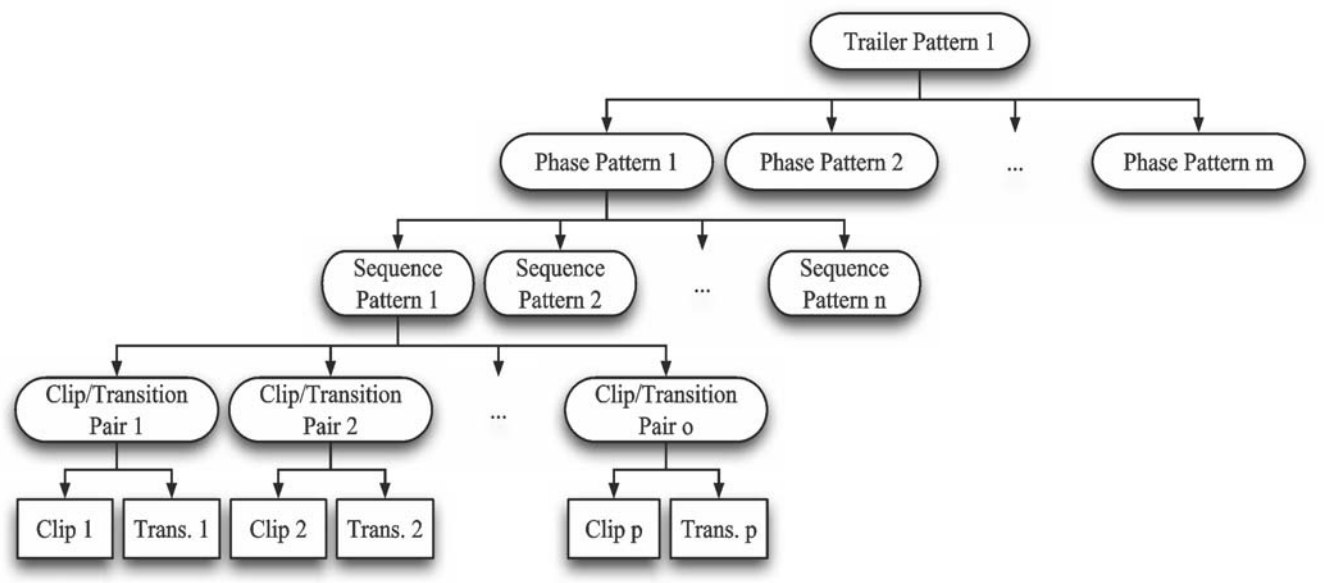

Figure 1: A branch of the hierarchical view of a generic trailer structure

4 System Framework

Our system framework comprises two major components. The first one is a collection of various low- and high-level image and audio processing modules, which provides information about a given movie by extracting a set of features. Each analysis module is described in the following sections 4.1.2 to 4.1.13. The second component provides an implementation of the proposed trailer rule base, which is described in detail in sections 4.2.1 to 4.2.6. This component is able to categorize the annotated information of the first component and to use that data to automatically assemble a full trailer.

4.1 Extracting and Annotating Movie Features

In order to extract features of a given movie we use on the one side methods of image and audio analysis on different levels of abstraction, and on the other side we derive data from Internet resources. By combining the output of several modules with each other we extend the complexity and reliability of the annotated data. Figure 2 gives an overview of the interdependencies among the single modules.

As a ground truth we manually annotated the movie *The Transporter (2002)* and we adapted the performance scale of precision ***P*** and recall ***R*** to frame ranges as following

$$\underline{\boldsymbol{P}} = \frac{\text{number of relevant frame ranges retrieved}}{\text{number of frame ranges retrieved}}$$

$$\underline{\boldsymbol{R}} = \frac{\text{number of relevant frame ranges retrieved}}{\text{number of relevant frame ranges}} \qquad (1)$$

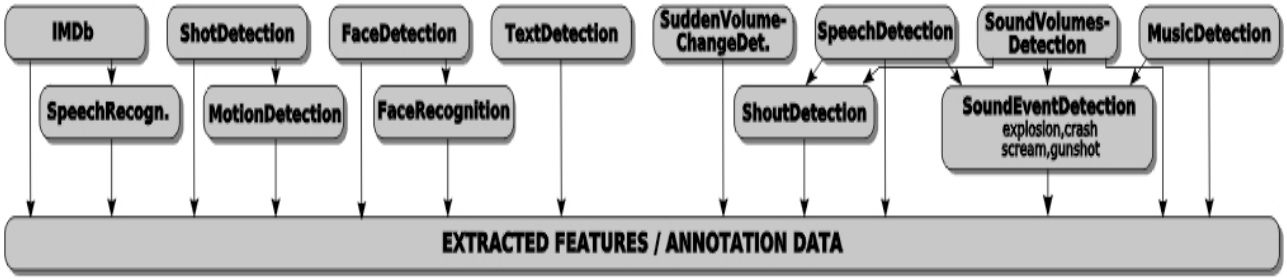

Figure 2: Overview of the modules for extracting features and their dependencies on each other

Therefore, we have a basis for evaluating the output of every module on basis of the approach of information retrieval. In the following section each single module is described along with the corresponding movie feature(s) it is providing.

4.1.1 Internet Resources

The Internet resources of the *Internet Movie Database*[1] (IMDb) are used to augment and enhance the generation of the trailer with automatically extracted data such as movie title, director, actors, genre, awards won and production company, which are used to generate credits for the trailer. In addition, famous quotes of the selected movie are extracted and used to perform a keyword-spotting in the speech recognition module.

This module is realized as a Python script using the IMDbPY[2] package to retrieve and manage the desired data of the IMDb.

4.1.2 Shot Detection

In order to detect shot boundaries we use an existing tool that was developed by other members of our research group [Miene *et al.* 2001]. However, our system just incorporates the shot boundary detection by the Gray Histogram X^2 Feature extraction so only hard cuts are detected. We set the adaptive threshold $Th_{percentage}$ to 7 and the minimum frame number to 7, which results in precision and recall values of 0.93 each. In addition, we calculate average color values for each detected shot. In future, the precision and recall values should be improved by extracting other features and detecting other transition types, e.g., dissolves.

4.1.3 Motion-based Segmentation

We also segment a movie into frame ranges with homogeneous motion intensities. First, motion intensities are calculated for each pair of adjacent frames by the pyramidal imple-

[1] Cf. http://www.imdb.com
[2] Cf. http://www.imdbpy.sourceforge.net

mentation of the Lucas Kanade feature tracker provided by the OpenCV library[3]. Due to its pyramidal approach, it deals with small and large motions in a balanced way. After the calculation, we add up all feature motions except for the ones below a certain threshold $T_{min-FeatMot}$ in order to disregard hardly noticeable motion. Furthermore, the calculated sum is reduced if the image brightness is low, as black frames tend to cause high motion intensity because of encoding artefacts. We also reduce this sum for a frame pair corresponding to a hard cut, as such a transition naturally causes high motion intensity.

Now, we use a grid of several motion classes in order to classify each frame pair motion according to its motion intensity per feature. Adjacent frames with the same class are then combined to frame ranges. In order to avoid too short motion frame ranges we define a minimum length. If a frame range has a length below this threshold T_{minLen} then it is combined with the neighboring frame range. The results of the motion-based segmentation run on nine test movies were very satisfying, particularly concerning frame ranges with very low and very high motion intensity, respectively. Future work should extend the movement information by differentiating between camera (zoom, pan, etc.) and object motion.

4.1.4 Face Detection

In order to find actor appearances within a movie, we use the basic face detection algorithm provided by the OpenCV library, which uses the Haarcascade classifier [Lienhart & Maydt 2002] for detection with a minimum of three neighbors used for grouping. After the detection process we cluster single detected faces to sequences so that we can define a frame range for an appearance of an actor. Variables considered during this clustering process are face deviation in size and position within a frame along with a threshold for closing gaps between successive frames caused by occlusions or head movement. For computing a mean face image and for the later definition of a subspace during the face recognition we use a publicly available face database[4]. The faces out of this database are all frontal faces in different lighting conditions without any rotation. We compute the distance of every face in a sequence to the mean image, i.e., to the mean face. The face that performs best is chosen to be the representative for the sequence. We assume that a small distance indicates a frontal face barely rotated. This distance along with face size in comparison to the frame width of the movie gives hints on close-up shots and not being a false positive.

The face detection achieves a precision of 0.8 and the distance to the mean image prevents many of the false positives of being used. However, future work should certainly involve a human skin detection to provide even more robust results.

[3] Cf. http://www.intel.com/technology/computing/opencv/
[4] Cf. http://www.equinoxsensors.com/products/HID.html

4.1.5 Face Recognition

For face recognition we decided on using Principal Component Analysis (PCA) as used in [Yambor 2000]. For the normalization process we implement a search for corners with big eigenvalues within an image to identify eye and mouth candidates. A problem arises, when only parts of a face are exposed to sunlight because they tend to produce stronger corners than parts left in shade. In order to compensate this, we impose a minimum distance between two points. We now search for strong corners as candidates with different distances imposed, compute a transformation matrix for every possible combination of candidates and apply it onto the face image. Our normalization outcome is a face image of 25x25 pixels with a mask applied on it occluding the background. We use 90 eigenfaces for the projection and do a k-means clustering on the results. We assume that the biggest cluster will be the cluster containing the first main actor, so that we achieve a precision of 0.59 and a recall of 0.16. Future work will be to implement a clustering that can deal with an unknown number of classes, and to produce a better homogeneity. Furthermore, it is necessary to improve the results of the face recognition by utilizing other techniques such as Elastic Bunch Graph Matching (EBGM; see [Wiskott *et al.*]1997).

4.1.6 Text Detection

The text detection is done by an existing tool of our research group first realized in [Wilkens 2003], which is specifically designed to find overlaid text in video. We use a 3x3 edge filter subtracting the lowest value from the center value for preprocessing. We then consider any group of more than three characters as text and choose rather strict settings in terms of deviation in tracking, horizontal spacing, vertical and horizontal scaling to keep the false positive rate as low as possible. This is important as shots containing text mostly end up in the black list during the generation process and we do not want to loose valuable shots. In case of our reference movie, the tool achieves a precision of 0.92 and a recall of 0.78.

4.1.7 Sound Volume-based Segmentation

Quiet portions of the movie will probably not contain action sequences but rather dialogs or scenery shots. Low volume can therefore be a very reliable indicator for falsely detected explosions, gunfire or other action-related elements. On the other hand, high volume can be a clue for action scenes, loud music or other noisy settings. One problem for the measurement of the audio volume is its quick fluctuation. It can vary significantly from one movie frame to the next. It is necessary to smooth the intensities over a range of many frames to get more stable and meaningful values. While smoothing the audio intensities and grouping them into frame regions it is desirable to assign these regions to portions of audio that maintain a relatively constant level. The borders to the neighboring regions

should be placed wherever there is a significant change in audio volume. We propose to set the starting points for the regions to the points of minimal change in the smoothing function. The borders between the regions are then adjusted so that the error in respect to the region's average is minimized.

4.1.8 Sudden Volume Change Detection

A sudden increase in the loudness of movie audio is a clue for a deliberately integrated surprise element. We define such an increase as an extended period of quiet audio, e.g., one second, followed directly by a noisy part, where a high level of audio is sustained for another second. This definition makes sure that short bursts of loud audio will not be counted. For various movies, these sudden volume changes are often explosions, crashes or surprise attacks. However, not all volume increases are necessarily due to spectacular effect scenes. Sometimes the contrast of volume is used to emphasize the harsh cut from a quiet scene to a loud setting, such as to a disco or a factory hall.

4.1.9 Speech Detection

We perform a segmentation of the movie into speech and non-speech. A segmentation using the Bayesian Information Criterion (BIC) and zero-crossings as described in [Biatov & Köhler 2003] is only applicable on radio or TV broadcast news, consisting of clear speech, silence and music without overlay. For a good and reliable segmentation of Hollywood movies these methods give no valuable results. In our approach, a speech recognition is performed on the movie using the CMU Sphinx 3.5[5] speech recognition system in combination with the pre-trained open source HUB4 acoustic models and the small AN4 3-gram language model based on 130 words and numbers. Finding speech in the movie, even when disturbing music and background noise is present, works very well because only the found frame ranges are used and the content of the speech is not important.

After running the speech recognition, the extracted frame ranges represent single words recognized by the speech recognition system. In order to reduce false positives, all frame ranges $F\ (f_s < F < f_e)$ with $F > 18$ frames are removed. By combining frame ranges that have a distance between each other of $f_{2s} - f_{1e} < 50$ frames a complete dialogue structure can be formed. The evaluation of the accuracy of the system results in a precision of 0.79 and recall of 0.77 at a real time computation speed. Most of the false positives are singing artists in the background music.

[5] Cf. http://cmusphinx.sourceforge.net

4.1.10 Speech Recognition

The speech recognition module performs a keyword spotting to find the frame ranges that comprise the given famous quotes extracted from the IMDb by the Internet resources module. A typical speech recognition system uses phoneme-based acoustic models in combination with a word or syllable-based language model [Schrumpf *et al.* 2005]. In research, the most often used data is clearly spoken broadcast news without background noise or music. In contrast, Hollywood movies contain lots of overlays of speech, music, and special sound effects. In addition, some other factors like slang, blurring of word boundaries, strong variations in articulation, and speaker- or character-dependent characteristics make it even more difficult to achieve good results in this scenario.

Our module uses the CMU Sphinx 3.5 speech recognition system in combination with the pre-trained open source HUB4 acoustic models and a language model built out of the extracted quotes. By means of the CMU-Cambridge Statistical Language Modeling toolkit[6] the language model is built and the text-to-phone software addttp4[7] is used to build the word-phoneme dictionary. The difference to other language models is the fact that our language model uses each quote as one entity in the model and we only build uni-gram models, because there are no dependencies between single quotes.

After performing the speech recognition, the frame range with the highest probability is selected for every quote. This results in a precision and a recall of 0.67 each at a computation speed of 5 times real time. The result of the recognition highly depends on the quality of the IMDb quotes, which sometimes are not verbatim and thus cannot be found by the system. The next step to enhance the results would be to train acoustic models on manually annotated Hollywood movies. This would incorporate background noise and music into the acoustic models and make it more fitting for this domain.

4.1.11 Shout Detection

We also try to locate frame ranges in the movie where people shout by combining the output of the speaker detection and the sound volume-based segmentation. The program searches for frame ranges where the normalized sound volume v with $0 \leq v \leq 1$ exceeds a threshold $v_t \geq 0.5$. Only frame ranges of the speech detection that are at the same range as the thresholded sound volume are extracted. The recognition works with a precision of 0.5 and a recall of 0.15. Half of the falsely classified ranges are screams that are very close to shouts.

[6] Cf. http://svr-www.eng.cam.ac.uk/~prc14/toolkit.html
[7] Cf. http://www.nist.gov/speech/tools/addttp4-11tarZ.html

4.1.12 Music Detection

The module detects music in the audio signal. The method we use was first proposed by [Minami *et al.* 1998]. Additionally, considering the method proposed in [Hawley 1993] we take stable power spectrum peaks as an indicator for music. An image-based approach is used to measure the presence of horizontal lines in the spectrum. The detection algorithm takes slices of a length of 10 seconds and calculates the power spectrum for 371 frequency bands up to 4kHz. A strong horizontal blurring operator is applied to the resulting spectral gray-scale image in order to emphasize horizontal lines and reduce other patterns (e.g., speech or noise). After an edge detection the image is binarized so that only horizontal lines longer than a certain threshold t_{length} are kept. For each time frame of a length of 0.5 seconds the sum of the edge pixels within this frame is considered to indicate music if it exceeds a defined threshold t_l. In addition, we use the distance of the sum to t_l as a degree of disturbance of the music. With $t_{length} = 16\ px$ and $t_l = 450\ px$ we achieve a precision of 0.95 and a recall of 0.87. This could be improved by including a beat detection algorithm as an additional feature.

4.1.13 Sound Event Detection

Our system implements a high-level sound detection method that can search the audio track of a movie for a number of previously learned sounds. We search for gunshots, explosions, crashes and screams in order to identify the movie's most dramatic and entertaining scenes.

Our approach is similar to the method proposed in [Hoiem *et al.* 2005]. For training, a small number of short example sounds between 0.5 and 2.5 seconds is cut from typical action movies and transformed into a simplified spectral representation with 17 frequency bands. A set of 63 descriptive features is then calculated from each sound. Among these features are the intensities in the different bands along with their standard deviation, a measure for the fluctuation of the overall intensity and rising or falling energy from start to end. The feature set is designed to be robust against differences in volume or length of the samples. The feature vectors from the positive and a great number of negative examples are then used to train a Support Vector Machine (SVM) for each sound type separately, using the LIBSVM[8] implementation.

In order to search a movie for any of the sound types, we traverse the movie in small steps of 0.1 seconds and calculate the above feature vectors over a length of 800ms (for gunshots), 1200ms (crashes and screams) to 2000ms (explosions). The compared length should roughly match the length of the training samples.

[8] Cf. http://www.csie.ntu.edu.tw/~cjlin/libsvm

The test movie performs with a precision of 0.4 and a recall rate of 0.4 for portions with gunshots. Problems arise if movies use sounds that are too different from the training examples, like futuristic weapons, or have very different audio characteristics in general (e.g., older movies in comparison to today's movies). The shrill nature of sounds is often mistaken for screaming by the SVM classification. Loud music is also a source for misclassification, as beats can be mistaken for gunshots or other types of crashing sounds. In general, the selection of training sounds has the greatest effect on the performance of the classification.

To improve the precision of the sound detection we use the output of other modules to filter out false positives. The sound volumes are used as a filter to count only loud enough sounds. Results from the music detection help to clean the list of detected gunshots from music beats. Explosion sounds will only be counted if they are accompanied by a sudden increase of brightness in the image histogram.

4.2 Generating Trailers of an Annotated Movie

After extracting the various features mentioned above, the second component of our system comes into play. The annotated movie containing the extracted features is used in combination with our semantic patterns in order to generate a trailer of the particular movie. [Zhu *et al.* 2003] uses a hierarchy for video summarization quite similar to that defined by our trailer rules base. However, it does not discuss video summarization for the movie trailer format. Also, they solely work with available video and audio footage. Our approach uses additional automatically generated animations and adds music and sound effects from footage-unrelated audio sources. The process that we propose of automatically generating trailers from annotated movies is split into the following sub-components:

1. Using a trailer rule base to create an abstract trailer structure that is used as a basis for
2. the selection of video and animation footage as well as music and sound effects to assemble a final trailer.

In Figure 3, the components of the generation process are displayed.

In more detail: In order to build a trailer we define a knowledge base that contains models for trailer structure elements and defines parameters for categories of video footage frame ranges. Before the generation process is started, we filter the movie annotation into the syntactic elements *clips* and classify them into *categories* with the mentioned parameters. Next, the trailer model is created based on rules in the knowledge base and influenced by the availability of footage and certain random events. In this way, the system generates a unique trailer model that is built to fit the available footage. The composition framework

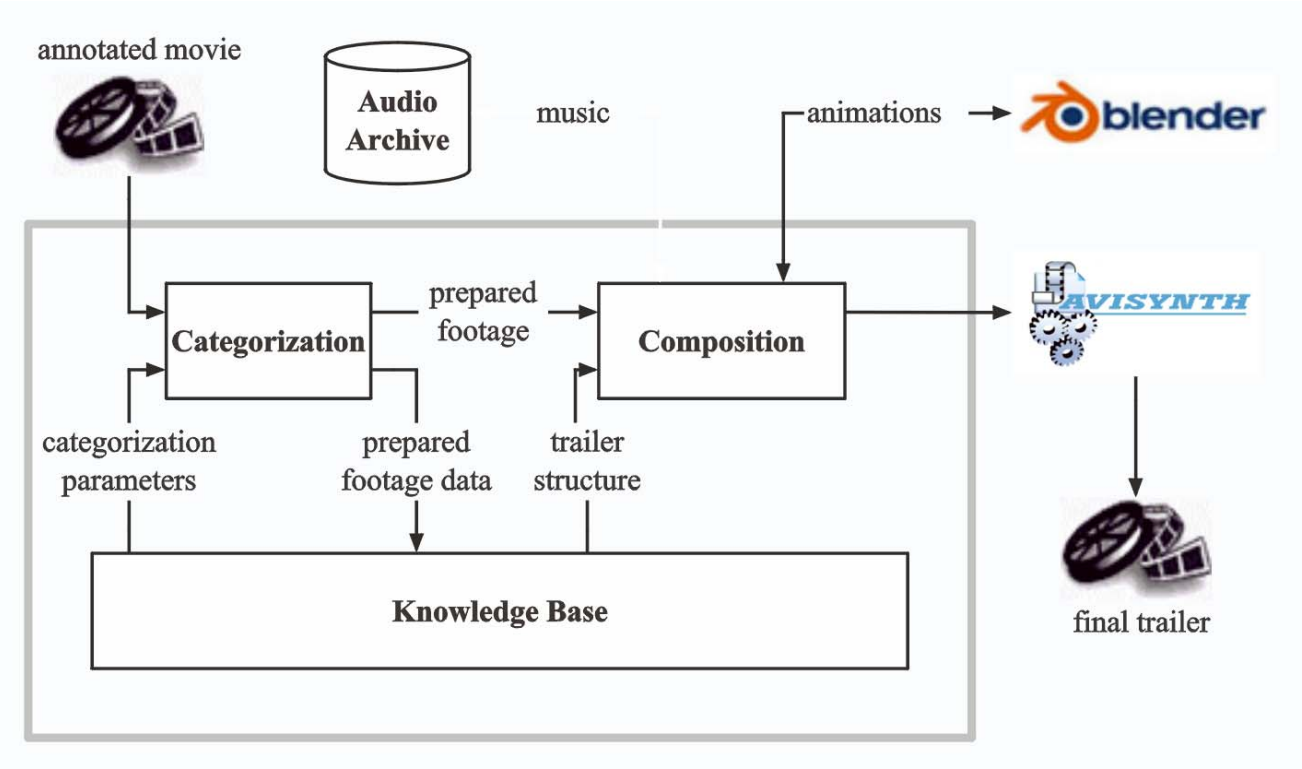

Figure 3: Diagram of the trailer generation process

translates the established trailer structure into specific trailer elements: Apart from video footage we incorporate runtime-created text animations for movie title, credits etc., as well as prepared music and sound effects content from our own audio archive. Footage, text animations, and audio are finally composed to a unique, fully automatically generated trailer based on trailer semantics.

4.2.1 Knowledge Base Functionality

In order to incorporate trailer semantics, we implement a knowledge base that is designed to hold the knowledge for trailer construction using the public domain software CLIPS[9]. The underlying knowledge of trailer syntax and semantics is modeled in an ontology. This trailer ontology includes classes for semantic structure elements (*patterns*) and syntactic elements (*clips*, *transitions*). We use relations to model our hierarchical view of the trailer structure with different types of patterns at the four upper levels and with different types of clips and transitions at the lowest level. The properties of clips (category, speed, volume, location) are implemented as slots and among these the category is implemented as a class of several slots again, specifying a list of video analysis attributes for its classification. The combination of several annotation attributes to a category leads to semantic higher-level knowledge about the footage.

[9] Cf. http://www.ghg.net/clips/CLIPS.html

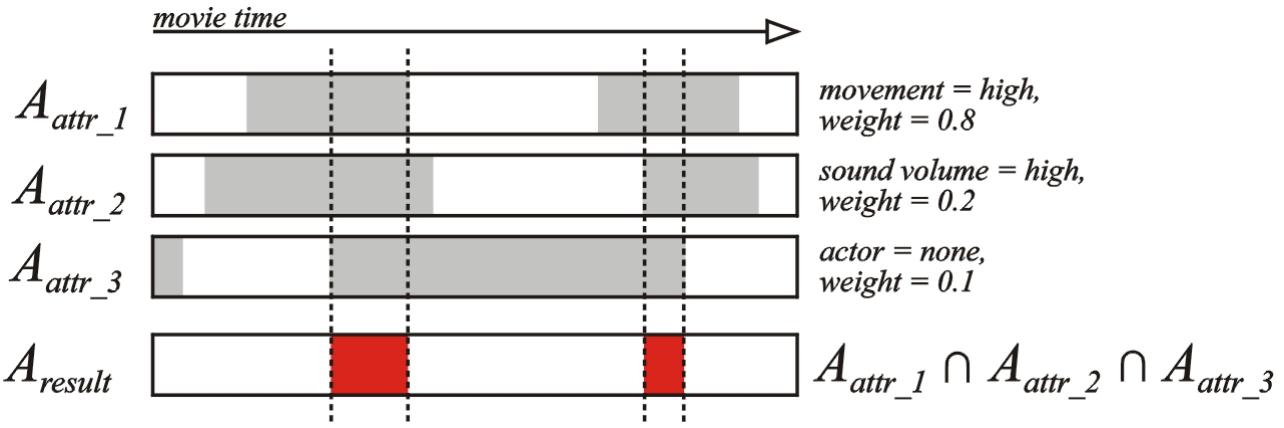

Figure 4: The intersection approach of the categorizer for sample footage

4.2.2 Categorization

Given a set of category definitions from the knowledge base the categorization module processes the annotation data in order to build clips for each category. We propose to build the clips based on frame ranges as opposed to a shot-based approach. As [Davis 1993] points out, a frame range approach allows the categorization process to be independent from scene/shot information and can provide categorized footage that starts or ends within a shot, or ranges over several shots. Hence, the main challenge is to determine frame ranges of the annotated movie, which match a certain category description from the knowledge base. Let A be a set of video frames and A_{movie} the set containing all original movie frames, then the frame set of an analyzed movie feature $A_{feature\ x}$ (e.g., all frames showing a face) is a subset of A_{movie}. The first step of the categorization process is to filter out the desired frames by corresponding thresholds (e.g., only get big faces indicating a close-up shot). This results in a new set which we refer to as an *attribute* frame set $A_{attribute\ x}$ with the following relations:

$$A_{attribute\ x} \subseteq A_{feature\ x} \subseteq A_{movie}$$

As illustrated in Figure 4, we process these attribute frame sets as tracks and perform an intersection of them. Furthermore, for each clip we calculate a probability value based on weighting factors assigned to the attributes. The result of the categorization process is a certain amount of footage clips for each category defined in our knowledge base.

4.2.3 Trailer Structure

Once the movie footage has been segmented and categorized, information about the amount of clips within each category is handed to the knowledge base. The system then builds the trailer structure on an abstract level. In order to introduce variety into the trailer models, each semantic element in our hierarchy has a selection choice of lower level elements assigned to it. These lists specifically define which items of a lower level a higher

one may select. Thus, while offering multiple choices at each node in the trailer structure tree, the sequence of patterns can still be controlled to ensure consistency with the given trailer grammar. This approach grants easy and fast altering of the structure by linking more sub-patterns to a super-pattern or by deleting links. To avoid a purely random selection of a linked sub-pattern and to emphasize patterns that are more frequently used in movie trailers, a weighting system is attached to the selection logic. Based on the trailer ontology and on the availability of categories the knowledge base reasons about which parts of the trailer structure fulfill all requirements. In case of certain parts failing due to lack of footage, fallback structures are considered first. If no such fallback exists clip attributes are loosened: clips can then be chosen from random categories rather than specific ones. The result is a finished model of a trailer structure giving detailed information about which transitions to use, which background music to play, which clips of which category to show, what position within the movie they should come from, what speed they should be played at, and how high the volume of the original footage should be.

4.2.4 Selection of Clips

The footage clip objects in the trailer model come with properties regarding clip category, footage volume, speed, and location for footage selection. Footage clips are always selected from those matching the category of the clip object. Within this limitation, our system has three methods for clip selection:

1. Preferred location selection, based purely on the requested location and clip location in the movie, so the clip chosen is the one closest to the requested location.
2. Best clip from preferred location, which is similar to the preferred location selection with an addition of taking the quality of the clip into consideration so the clip chosen is the best clip available starting from the requested location.
3. A random clip of a given category is selected.

4.2.5 3D Text Animations and Audio

Text animations displaying information on movie title, release date, actor names, movie company as well as legal disclaimers are one distinctive feature of movie trailers and an essential component of a trailer structure. Our system uses the 3D software Blender[10], which offers animation and render control via Python scripts. Given a set of certain animation templates the composition system dynamically creates a script from which Blender produces one digital video file per animation ready to be used for final composition.

[10] Cf. http://blender.org

For additional music soundtrack and sound effects we provide the possibility to incorporate pre-produced sound files. Music files can be assigned to different trailer phases (that have to be predefined for a specific trailer pattern), while sound effects are divided into types. For every phase or element we have a choice of audio files.

4.2.6 Final Composition

Selected footage, animation clips and audio soundtrack are composed into a final video using Avisynth[11] scripts. Text animations are given sound effects to improve their effect. Changes in trailer soundtrack are masked by special transition sound effects. Fade/flash shot transitions (as determined by the trailer structure model) are implemented. The result is the finished trailer modeled according to a trailer structure created using our trailer ontology.

5 Automatic Trailers for Action Movies

Most trailers try to summarize the plot and setting of the announced movie and to introduce the relations between the main characters. Presently, the automatic extraction, analysis and generation of a narrative, or at least some kind of dramatic arc, seem hardly feasible. Therefore, our approach focuses on a genre that relies significantly more on visual sensation, speed and effects, than on narrative: the action movie. In order to generate trailers for this genre, specific grammar elements of action movie trailers need to be identified. In the next section we present parts of our sample pattern using a specific set of *sub-patterns*, *clips*, and *transitions*, which is based on a shot-by-shot analysis of various action movie trailers from the last 15 years.

5.1 Applying the Rules to the Construction of a Trailer

We explicitly define 3 transition and 38 clip types, derived from 26 clip categories listed in Table 1. The definition of each category includes an appropriate set of attributes along with specific value ranges for the annotated features (t_{lo}, t_{hi}) and weighting factors (w_{attr}). One example of such a definition is shown in Table 2. An extension of our set of categories is possible and would be necessary to model and generate more complex trailers. On the second level of our hierarchy we identify five different phases (which are composed by a number of sub-patterns) as the basic structure in most action trailers. They are:

- Intro (slow and moody shots of locations and people together with speech establishing a conflict or introducing the main characters)

[11] Cf. http://www.avisynth.org

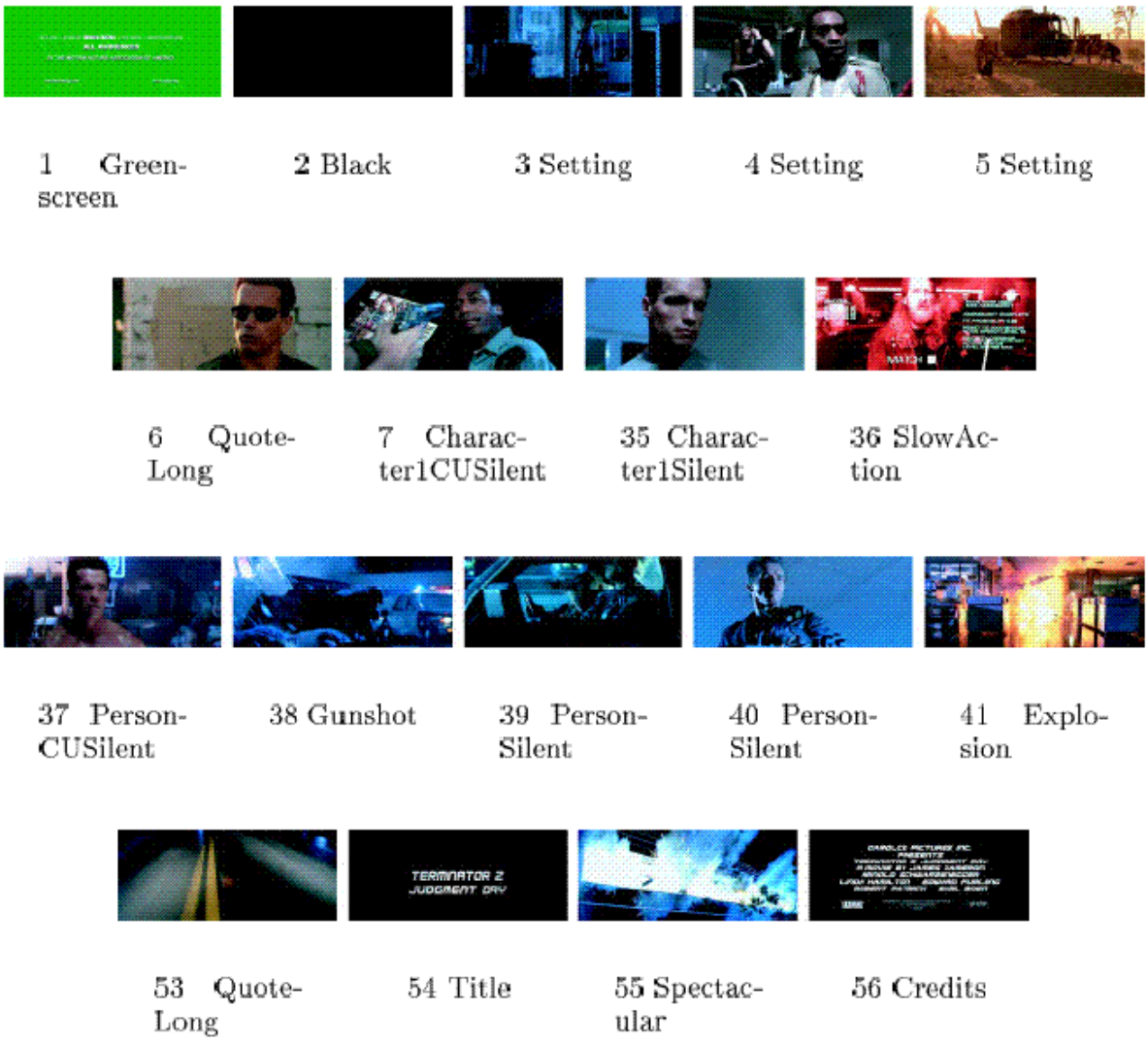

Figure 5: 18 of 56 clips showing parts of our automatically generated Terminator 2 *trailer (complete Intro Phase: 1-6, middle part of the Action Phase: 35-41, and complete Outro Phase: 53-56). The corresponding type of category is given below each clip.*

- Story (medium fast shots of action and people together with dialogue to wrap up the task the main characters have to face)
- Break (a long and very significant or dramatic comment by one of the main characters - typically without background music)
- Action (a fast montage with loud sound of the fastest action scenes together with close-ups of the main characters)
- Outro (typically very calm or without any music and shows – sometimes mixed with close-ups or a short shot of one of the main characters uttering an extremely comic or tough comment – the title and credits of the movie together with a release date)

	I	S	B	A	O
Transition					
FadeBlack	x	x			x
FlashWhite				x	
HardCut	x	x	x	x	x
Footage Clip					
Character1CUSilent	x	x		x	x
Character1CUSpeaking	x	x			
Character1Silent	x	x		x	x
Character1Speaking	x	x			
PersonCUSilent	x	x		x	x
PersonCUSpeaking	x	x			
PersonSilent	x	x		x	x
PersonSpeaking	x	x			
Quote	x	x	x		x
QuoteLong	x		x		x
Explosion		x		x	
Fire		x		x	
Gunshot				x	
FastAction				x	
SlowAction		x		x	
Spectacular				x	x
Shout		x		x	
Scream				x	
Setting	x	x			
Animation Clip					
ActorName				x	
CompanyName	x	x			
Credits					x
DirectorProducer		x			
Greenscreen	x				
Tagline	x	x			
Title					x

Left
Table 1: Clip Category and Phase Pattern relations in our sample Trailer Pattern (CU: close-up). I stands for Intro, S for Story, B for Break, A for Action, and O for Outro

Attribute	t_{lo}	t_{hi}	w_{attr}
Movement	0.000	0.003	0.2
SoundVolume	0.0	0.1	0.1
Text	-1.0	0.0	0.1
Duration	60	500	0.1
CharacterFace	-1.0	0.0	0.3
CharacterSpeech	-1.0	0.0	0.1
MovieLocation	0.2	0.9	0.1

Table 2: Parameters for the Setting category

With the defined elements (clips, transitions and patterns), as well as their relations to each other we are finally able to describe a simple action movie trailer in a formal way. This description can be used by our system to generate an action trailer from any movie

(as long as the automatic analysis provides enough footage for the different categories). In a simplified schematic way the relation between categories (being the basis for the clips), transitions and patterns that constitute our trailer structure can be described in a two-dimensional matrix as in Table 1.

5.2 3D Text Animations and Audio for Action Movie Trailers

In order to include text animations we provide our system with four animation templates which all have a different artistic style. Our audio archive is a collection of pre-produced sound files and consists currently of 37 music files and 22 sound effect files. Currently we use four categories of music files according to the mood of our trailer phases (Intro, Story, Action, Outro) and three sound effect types that are mostly used in professional trailers ("boom", "woosh" and "wooshbang").

6 Experimental Results

In order to evaluate the quality of our trailers, we asked 59 people to evaluate seven test trailers. For each of the trailers, the test people were asked to state whether they have seen the movie and to rate the same six aspects (with 1 as the lowest and 10 as the highest score). The test set comprised:

- Two professional trailers: *War of the Worlds* (Golden Trailer Award winner 2005) and *Miami Vice*
- One trailer for *The Transporter* produced by the video generation software muvee™ (random shot selection)[12]
- Two trailers produced by our system with different levels of randomness: *Bad Boys* (random frame ranges), *Blade* (random *clip* selection)
- Two trailers produced by our system based on our Trailer Patterns: *Transporter 2* and *Terminator 2*

Except for the last trailer (*Terminator 2*, see Fig. 5), the test people did not know how the trailers were produced. After the impression of all trailers, the test people had the opportunity to watch any trailer again in case they wanted to adjust the ratings. The detailed scores of all trailers are shown in Table 3. As expected, the overall rating of the random trailers is significantly lower than any of the others, while *War Of The Worlds* performed best (see Fig. 6). The *Miami Vice* trailer that had been chosen as an example for a low-quality professional trailer was indeed given a bad score. Our own generated trailers reached an average score of 7.29 and 7.26, respectively. More than 80% of the viewers

[12] Cf. http://www.movee.com

	ss	co	ce	ci	pi	av
War of the Worlds	8.41	7.91	7.79	7.47	7.40	8.16
Miami Vice	4.97	6.27	6.27	3.27	3.59	4.95
The Trans-porter	5.05	4.03	4.22	2.97	3.59	3.95
Bad Boys	4.64	3.67	3.41	3.19	3.22	3.52
Blade	4.16	3.24	3.24	4.07	4.07	3.43
The Trans-porter 2	7.47	7.54	7.90	6.80	6.80	7.37
Terminator 2	7.58	7.63	6.36	6.88	7.46	7.37

Table 3: Detailed scores from the user testing; ss stands for scene selection, co for composition, ce for cuts & effects, ci for character introduction, pi for plot introduction, and av for advertisement value

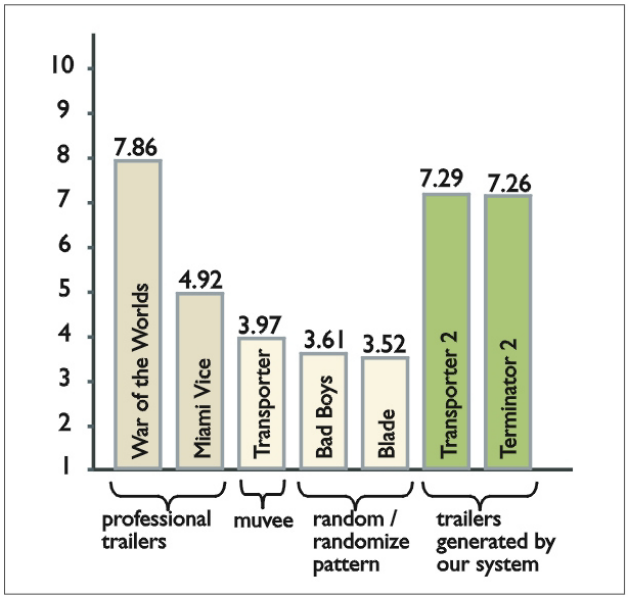

Figure 6: Mean score of the test trailers

rated them at score 7 or better. The question whether people had seen a movie or not seemed to have little or no impact on their judgment.

According to our survey, the weakest elements of our trailers are the introduction of main characters and topic / storyline with each around 6 points on average. This score is still good regarding the fact that we could not deliberately include those aspects into our trailers. A possible conclusion is that an illusion of storytelling and character introduction is created by extensively using quotes. At the end of the test screening, we asked the test viewers to rate the importance of six different aspects for Hollywood trailers. In average, people voted mostly for an even balance of the aspects. A slightly larger share was given to ‘dramaturgy’ (20.51%) and ‘action scenes’ (18.77%). The importance of ‘voice-over’ and ‘illusion of speed’ was rated a little lower (12.58%, 13.13%). In relation to this, we asked the viewers to rate the quality of integration in our *Terminator 2* trailer for the same aspects. The smallest share of votes was achieved for ‘voice-over’ (9.28%). This was to be expected since it is basically missing in any of our trailers. The best rates were given to ‘action scenes’ (22.26%), ‘music, animation and sound effects’ (18.6%) and ‘distinctive pieces of dialogs and statements’ (18.19%). This shows that our attempts of automatic categorization and composition appear to be successful in general. When interpreting the test results, the number of test people and the fact that they were not chosen representatively should be considered. However, the results suggest that our automatic trailers are in most respects comparable to original trailers and may even be more accepted by audience than low-quality professional trailers.

Sample trailers produced by our system can be downloaded from http://www.tzi.de/svp.

7 Conclusion and Future Work

This paper presents a novel approach of intensively using a knowledge base (rules) in combination with data automatically extracted out of a movie by different image and audio analysis techniques for generating a Hollywood-like movie trailer. First, the rules were defined which can be applied to various movie genres. Second, a system was implemented, which provides means for using extracted features to build a trailer according to any defined Trailer Pattern based on our trailer grammar. One such Trailer Pattern was created by manually analyzing several action movie trailers. Using our system we generated trailers for some action movies according to this pattern, and we evaluated our outcome.

After all, with our action movie trailers we proved that automatic trailer generation is not only possible, but can even achieve good results. This has been proven by tests we conducted with human subjects. Still, our trailers lack some elements a manually edited trailer comprises, e.g., telling a coherent story or voice-over narration.

The system can be improved by enhancing the modules extracting the data and by expanding the trailer grammar. Extensions of our modeled knowledge to incorporate editing knowledge for trailers of other movie genres, less standardized trailers or even other video summary formats are conceivable and can be achieved by adding more patterns, clips and transitions. Also, the classification of movie footage into semantic categories could be expanded by adding more categories (e.g., “kissing”, “fight”) based on more sophisticated image and audio processing techniques. Concerning the composition framework, improvements and extensions for the animation and audio inclusion are conceivable, notably to add more animation styles as templates and have a way of matching styles to movie content. Finally, the effect of a generated trailer can also be vastly improved by adding pre-produced generic voice-overs to the soundtrack.

A general different approach would be to manually add special information that might lead to more artistic trailers but would result in the loss of total automatism.

References

Arijon, D.: *Grammar of the Film Language*. Silman-James Press, 1991.

Babaguchi, N. & Kawai, Y. & Kitahashi, T.: Event Based Video Indexing by Intermodal Collaboration. In *Proc. of 1st Int. Workshop on Multimedia Intelligent Storage and Retrieval Management*, Orlando, 1999, 1–9.

Biatov, K. & Kohler, J.: An audio stream classification and optimal segmentation for multimedia applications. In *Proc. of 11th ACM Int. Conf. on Multimedia*, Fraunhofer Institute for Media Communication, Sankt Augustin, 2003, 211–214.

Chen, H.-W. & Kuo, J.-H. & Chu, W.-T. & Wu, J.-1.: Action movies segmentation and summarization based on tempo analysis. In *MIR ‘04: Proc. of 6th ACM SIGMM int.*

Workshop on Multimedia information retrieval, New York, NY. ACM Press, 2004, 251–258.

Christel, M. G. & Hauptmann, A. G. & Warmack, A. & Crosby, S. A.:. Adjustable Filmstrips and Skims as Abstractions for a Digital Video Library. In *ADL*, 1999, 98–104.

Davis, M.: Media streams: An iconic visual language for video annotation. *IEEE Symposium on Visual Language*, 1993, 196–202.

Hawley, M.: *Structure out of sound*. PhD thesis. MIT, Massachussetts, 1993.

Hoiem, D. & Ke, Y. & Sukthankar, R.: SOLAR: Sound Object Localization and Retrieval in Complex Audio Environments. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP*), 2005, vol. 5, 429–432.

Kernan, L.: *Coming Attractions – Reading American Movie Trailers*. Univ. of Texas Press, Austin, 2004.

Lienhart, R. & Maydt, J.: An Extended Set of Haar-like Features for Rapid Object Detection. In *IEEE ICIP*, 2002, vol. 1, 900–903.

Lienhart, R. & Pfeiffer, S. & Effelsberg, W.: Video Abstracting. *Communications of the ACM,* 1997 40(12):54-62.

Ma, Y. & Lu, L. & Zhang, H. & Li, M.: A User Attention Model for Video Summarization. In *Proceedings of ACM Multimedia*, 2002, 533–542.

Miene, A. & Dammeyer, A. & Hermes, T. & Herzog, O.: Advanced and Adapted Shot Boundary Detection. In Fellner, D. W. & Fuhr, N. & Witten, I. (eds.): *Proc. of ECDL WS Generalized Documents*, 2001, 39–43.

Minami, K. & Akutsu, A. & Hamada, H. & Tonomura, Y.: Video Handling with Music and Speech Detection. *IEEE MultiMedia*, 1998 5(3):17–25.

Ponceleon, D. & Dieberger, A.: Hierarchical brushing in a collection of video data. *Proc. of 34th Hawaii Int. Conf. on System Sciences*, 2001, 1654–1661.

Schrumpf, C. & Larson, M. & Eickeler, S.: Syllable-based Language Models for English Spoken Document Retrieval. In *Proc. of 7th Int. Workshop of the EU Network DELOS on Audio-Visual Content and Information Visualization in Digital Libraries*. Fraunhofer Institute for Media Communication, Sankt Augustin, 2005, 196–205.

Smith, M. & Kanade, T.: Video skimming and characterization through the combination of image and language understanding. *IEEE Int. Workshop on Content-Based Access of Image and Video Database*, 1998, 61–70.

Snoek, C. & Worring, M.: Multimodal Video Indexing: A Review of the State-of-the-Art. *Multimedia Tools and Applications,* 2005 25:5–35.

Uchihashi, S. & Foote, J. & Girhensohn, A. & Boreczky, J.: Video Manga: Generating Semantically Meaningful Video Summaries. *Proc. ACM Multimedia 99*, 1999, 383–392.

Wilkens, N.: Detektion von Videoframes mit Texteinblendungen in Echtzeit. Master's thesis, Universität Bremen, 2003.

Yambor, W. S.: Analyis of PCA-based and Fisher Discriminant-based Image Recognition Algorithms. Master's thesis, Colorado State University, 2000.

Yeung, M. & Yeo, B.: Video visualization for compact presentation and fast browsing of pictorial content. *IEEE Trans CSVT,* 1997 7:771–785.

Zhang, H. & Kankanhalli, A. & Smoliar, S. W.: Automatic partitioning of full-motion video. *Multimedia Syst.*, 1993 1(1):10–28.

Zhu, X. & Fan, J. & Elmagamid, A.K. & Wu. X.: Hierarchical video content description and summarization using unified semantic and visual similarity. *Multimedia Syst.*, 2003 9(1):31–53.