SEARCHING MOVIES BASED ON USER DEFINED SEMANTIC EVENTS

Bart Lehane
Centre for Digital Video Processing
Dublin City University

Noel E. O'Connor
Adaptive Information Cluster
Centre for Digital Video Processing
Dublin City University

Hyowon Lee
Centre for Digital Video Processing
Dublin City University

Keywords: Video Retrieval, Event-Based Movie Indexing, Video Searching.

Abstract: The number, and size, of digital video databases is continuously growing. Unfortunately, most, if not all, of the video content in these databases is stored without any sort of indexing or analysis and without any associated metadata. If any of the videos do have metadata, then it is usually the result of some manual annotation process rather than any automatic indexing. Locating clips and browsing content is difficult, time consuming and generally inefficient. The task of managing a set of movies is particularly difficult given their innovative creation process and the individual style of directors. This paper proposes a method of searching video data in order to retrieve semantic events thereby facilitating management of video databases. An interface is created which allows users to perform searching using the proposed method. In order to assess the searching method, this interface is used to conduct a set of experiments in which users are timed completing a set of tasks using both the searching method and an alternate, keyframe based, retrieval method. These experiments evaluate the searching method, and demonstrate it's versatility.

1 INTRODUCTION

Decreasing production costs as well as widespread availability of content creation tools have led to an increase in the amount of digital video being produced. As the video databases which store this content continue to grow, they are becoming increasingly difficult to manage. This difficulty is largely due to the lack of information about the stored video. Manually associating metadata to video is both labor intensive and time consuming, so is impractical in many cases. Ideally, some automatic process is required in order to structure the data so that the video can be efficiently managed. This paper presents an approach which facilitates searching of video data in order to retrieve semantic events.

There are many video content types that are in need of indexing. Sporting programs, news programs, or documentaries are just some of the content genres created each day. However, fictional video content, specifically movies, are a medium particularly difficult to manage for a number of reasons. Firstly, their temporally long nature means that it is difficult to manually locate particular portions of a movie, as opposed to, say, a thirty minute news program. Most movies are at least one and a half hours long, with many as long as three hours. Summarisation of movies is also hindered due to its challenging nature. Each movie is created differently, using a different mix of directors, editors, cast, crew, plots etc., which results in varying styles. Filmmakers are given ample opportunity to be creative in how they shoot each scene, which results in diverse and innovative video styles. This is in direct contrast to the way most news and sports programs are created, where a rigid broadcasting technique must be followed as the program makers have an extremely short time span in which to make their program.

There have been many approaches to movie summarisation, although most focus on segmenting the

Lehane B., E. O'Connor N. and Lee H. (2006).
SEARCHING MOVIES BASED ON USER DEFINED SEMANTIC EVENTS.
In *Proceedings of the International Conference on Signal Processing and Multimedia Applications*, pages 232-239
DOI: 10.5220/0001569802320239

movie or detecting specific occurrences rather than implementing a searching strategy. Many approaches aim to summarise movies by detecting scene boundaries and creating a scene based index. For example, (Yeung and Yeo, 1996; Yeung and Yeo, 1997; Rui et al., 1998; Zhou and Tavanapong, 2002) all aim to cluster similar shots together, and locate areas where one set of clusters does not relate to any previous clusters. Other approaches, such as (Li and Kou, 2003; Rasheed and Shah, 2003; Sundaram and Chan, 2000), use the concept of shot coherence to locate scene boundaries. In general, this involves locating areas where fundamental changes in audiovisual features occur. However, a scene based index does not carry any semantic meaning, and it is difficult to locate a sought part of a movie with scene boundary information alone, unless each individual scene is viewed.

Many other movie summarisation approaches focus on detecting individual event types from the video. (Leinhart et al., 1999) detect dialogues in video based on the common shot/reverse shot shooting technique, which results in detectable repeating shots. This approach however, is only applicable to dialogues involving two people, since if three or more people are involved the shooting structure will become unpredictable. Also, there are many other event types in a movie or television program apart from dialogues. (Li and Kou, 2003; Li and Kou, 2001) expand on this idea to detect three types of events, 2-person dialogues, multi-person dialogues and hybrid events (where a hybrid event is everything that isn't a dialogue). However, only dialogues are treated as meaningful events and everything else is declared as a hybrid event. (Chen et al., 2003) aim to detect both dialogue and action events in a movie, however the same approach is used to detect both types of events, and the type of action events that are detected is restricted. (Nam et al., 1998) detect violent events in a movie by searching for visual cues such as flames or blood pixels, or audio cues such as explosions or screaming. This approach, however, is quite restricted as there may be violent events that do not contain their chosen features. (Kang, 2003) extract a set of colour and motion features from a video, and then use relevance feedback from individual users to class events into 'fear', 'sadness', or 'joy'. (Zhai et al., 2004) generate colour, motion and audio features for a video, and then use finite state machines to class scenes into either conversation scenes, suspense scenes or action scenes. However, this approach relies on the presence of known scene breaks, and classifies a whole scene into one of the categories, while in reality an entire scene may contain a number of important events. Previous work by the authors (Lehane and O'Connor, 2006), created an event based index of an entire movie by devising a set of event classes that cover all of the meaningful events in a movie. By detecting each of the events in an event class, the entire movie is indexed.

The object of the searching method proposed in this paper is to retrieve sought *events* a movie. An event is something which progresses the story onward. Events are the portions of a movie which viewers remember as a semantic unit after the movie has finished. A conversation between a group of characters, for example, would be remembered as a semantic unit ahead of a single shot of a person talking in the conversation. Similarly, a car chase would be remembered as 'a car chase', not as 50 single shots of moving cars. A single shot of a car chase carries little meaning when viewed independently, it may not even be possible to deduce that a car chase is taking place from a single shot, however, when viewed in the context of the surrounding shots in the event, its meaning becomes apparent. Events are components of a single scene, and a scene may contain a number of different events. For example, a scene may contain a conversation, followed by a fight, which are two distinct events. Similarly, there may be three different conversations (between three sets of people) in the same scene, corresponding to three different events. The searching system in this paper aims to return events to a user, rather than specific shots. This allows for efficient retrieval.

Previous work by the authors focused on the detection of specific events in movies. (Lehane et al., 2005) proposed a method of detecting dialogue events, while (Lehane et al., 2004a) proposed an exciting event detection method. The approach in (Lehane and O'Connor, 2006) built upon previous work in order to generate a complete event-based summary of a movie. In each of the approaches above, a set of audiovisual features were generated, and finite state machines were utilised in order to detect the relevant events. Each of the approaches examined film grammar principles in order to assist in the event detection process. The work presented in this paper extends upon the event detection structure in order to allow user specified searching. This allows a human browser to retrieve events based on their requirements, rather than on a predefined structure. As this paper presents an extension of previous work, the explanation of portions of the system design elements is reduced, in particular the feature generation process.

The presented approach only utilises audiovisual searching. No textual information is used, and the results are solely based on the extracted features. Section 2 describes the feature generation part of the search system, which includes the feature selection process, and the creation of a shot-level feature vector. Section 3 describes the actual search method. This is a two step process. Firstly, finite state machines (FSMs) are utilised in order to generate a set of sequences, and secondly, a layer of filtering is undertaken and a set

of events is returned. A user interface which allows searching using the proposed method is presented in Section 4. Following this, a set of experiments and results in which the search based system is compared to a shot-based retrieval system are presented in Section 5. Finally, Section 6 contains a set of conclusions and future work.

2 FEATURE GENERATION

This section describes the features extracted in order to retrieve events. As the system is used for retrieval of events within movies, the features were chosen based on the amount of information they convey regarding the occurrences in movies. By examining film making principles it is possible to select a set of features that indicate a film makers intent, and therefore contain the most information as to the occurrences in the movie. These features are extracted and combined into a *shot-level feature vector*. All of the analysis in this paper is undertaken on MPEG-1 video and PCM encoded WAV audio, but could be applied to other formats.

2.1 Feature Selection

The techniques used by directors and editors to create films were examined, and the resultant observations were utilised in order to aid in the design of the searching system. Thus, the audiovisual features which can be used to garner the most information from a movie are extracted and incorporated into the system. For example, the use of camera movement is often used in order to show excitement during an event. If an event such as a fight is taking place, typically the camera moves rapidly to follow the action. Contrarily, a lack of camera movement is used in order to show relaxation, and to allow users to focus on the occurrences on screen (Bordwell and Thompson, 1997). Therefore, knowledge of the amount of camera motion present in a given shot, or sequence of shots, is important.

The type of audio present is also useful for searching as it allows a user to infer activities of characters. For example, if a user is searching for a conversation between characters, then knowing the areas of a movie where the audio type is 'speech' is useful. Similarly, if a montage[1] event is sought, then areas with musical shots will be of interest to a searcher. Thus, a set of audio features were chosen that can be used to classify the audio into a number of classes that describe the different types of audio present in a movie. These classes are: Speech, Music, Silence, Quiet Music and Other.

Colour information can also be used to infer semantics from a movie. Firstly, colour information can be used in order to detect shot boundaries and select keyframes for each shot, which is a first step for most video indexing systems. The frequency of shot cuts can also be detected using the shot boundary information. Increased editing pace is often used by film makers to create excitemtent. When shooting a particular event, lighting and colour typically remain consistent throughout. This can be utilised to locate boundaries between events, as when the overall colour changes significantly it is a strong indication that an new event has begun (Bordwell and Thompson, 1997). Thus, similar shots can be clustered together in order to find locations in a movie where one event ends and another begins. The clustering information can also be used to determine how much shot repetition is taking place in a given sequence of shots. This may indicate interactions between characters.

2.2 Feature Extraction

The aim of the feature extraction process is to generate a set of features for each shot in a movie. This set of features is termed the *shot-level feature vector*.

2.2.1 Colour Features

The aim of colour analysis is to generate a feature that represents a frame of video so that it can be compared to other frames, both for frame matching and for shot boundary detection. In order to achieve this, colour histograms are utilised. A 64-bin colour histogram is extracted for each frame of video. In order to detect shot boundaries, the inter frame histogram difference is calculated. If it is greater than a predefined threshold, a shot boundary is declared (Lehane et al., 2004b). One keyframe is selected per shot by examining the average values of the frames in a shot and locating the frame which is closest to the average. Using the shot boundary information, the shot lengths are found and inserted into the shot-level feature vector.

Once the shot boundaries are located, keyframes that have similar colour are clustered together. There are two reasons for clustering shots together, the first is to locate boundaries between events. Using a technique adapted from (Yeung and Yeo, 1996) (explained more thoroughly in (Lehane et al., 2005)), it is possible to locate areas in the movie where the camera moves from shooting in one location to another, signifying an event boundary. The second reason for clustering shots is so that shots filmed with the same camera angle (for example, repeating shots in a conversation) can be identified. This gives information about

[1] A montage is a collection of shots that span space and/or time. Montages usually have a strong musical background

the type of occurrence on screen. For example, an area populated with repeating shots is more likely to involve interaction between characters than one with many different shots. A measure of shot repetition can be generated for a sequence of shots by examining the cluster to shot ratio (C:S ratio), first presented in (Lehane et al., 2004b). If there are many repeating shots (and therefore many clusters), the C:S ratio will be quite low. If, however, most of the shots in a sequence are visually different, there will be many clusters, giving a higher C:S ratio.

2.2.2 Motion Feature

As outlined previously, MPEG-1 video is used for the analysis, thus the motion vectors created during encoding can be used. Due to the fact that motion vectors are compressed in the MPEG-1 data stream, a full decode of the video is not required, which means that analysis can be extremely fast. The motion information from each P-frame is extracted.

The amount of camera movement in each shot is detected. This method was previously presented in (Lehane et al., 2005). This approach involves examining the motion vectors across the entire frame. A threshold is used to distinguish between P-frames with/without camera movement. In order to generate a shot-based value for camera movement, the percentage of P-frames with camera movement in each shot is calculated.

2.2.3 Audio Features

Each audio feature is generated at a rate of one per second. In total, four audio features are extracted. The *High Zero Crossing Rate Ratio*, the *Silence Ratio*, the *Short Term Energy*, and the *Short-Term Energy Variation*. The effectiveness of these features in distinguishing between speech and music has previously been demonstrated (Chen et al., 2003; Lehane et al., 2005). In order to classify each second of audio, a set of support vector machines (SVMs) are utilised. Details are provided in (Lehane et al., 2005). In summary, the SVMs classify each second of audio into an audio class. These values are then up-sampled in order to create a value for each shot. At the end of this process, for each shot of a movie, there is a value for the percentage of speech, music, silence, quiet music, and other sound present.

3 SEARCHING TECHNIQUE

After feature generation, the shot-level feature vector contains the following values for every shot in a movie: [shot length, % static camera frames, % non-static camera frames,% speech, % music, % silence, % quiet music, % other audio]. This feature vector is used in order to conduct searching. There are two steps involved in the search process. Firstly, an array of finite state machines is created which allow a user to locate areas where particular features dominate. So for example, a music FSM locates the areas where a large number of music shots occur in succession (termed a music sequence). Following this, an optional filtering step reduces the set of sequences returned by the the FSM. So, for example, a user might be interested in locating all of the areas in a movie where there is a musical track combined with high amounts of camera movement. In this case, the filtering would involve removing all of the returned music sequences from the music FSM that contain *low* amounts of camera motion. The searching process is depicted graphically in Figure 1.

3.1 Sequence Generation

Finite state machines are used in order to generate sequences of shots in which a particular feature dominates. As can be seen from Figure 1, six FSMs are created, corresponding to the features in the feature vector. There is a speech FSM, a music FSM, a non-speech FSM, a static camera FSM, a non-static camera FSM and a high motion/short shot (HSMM) FSM. The speech, music, static camera, and non-static camera FSMs all locate areas where that particular feature dominates. The non-speech FSM locates areas where there is no speech present (i.e. the audio type is either silence, music or other). The HMSS FSM was designed due to the strong link between increased editing pace and camera movement when film makers want to depict excitement, and therefore it locates areas where both of these features are present. Each FSM produces a list of start and end times (i.e. sequences) throughout the movie where the input feature(s) are dominant. The presence of an event boundary resets the FSM, and signals the end of a particular sequence. For further explanation of the FSMs, interested readers are advised to consult (Lehane et al., 2005; Lehane et al., 2004a).

3.2 Filtering

The second step in the searching process involves filtering the set of sequences generated by a FSM. For example, the speech FSM may locate forty sequences in a movie where speech shots are dominant (i.e. there is somebody talking). The filtering step allows users to reduce this by rejecting sequences which do not contain a certain feature. For example, filtering allows a user to select the maximum or minimum percentage of static camera shots that must be present in a sequence. Thus, a user can place a constraint in

which at least, say, 60% of the shots in a generated speech sequence must contain a static camera for it to be retained. In total there are seven filtering options, which allow users to set boundaries for: the percentage of static shots, the percentage of non-static shots, the percentage of speech shots, the percentage of music shots, the percentage of non-speech shots, the percentage of short shots, and the amount of shot repetition. Any sequences that are retained after filtering are termed *events*.

After the two-step search, a list of start and end times of events is generated. Each of these events corresponds to an area in which the selected features dominate. In order to prepare these events for presentation, a set of five keyframes are chosen to represent the entire event. These keyframes are chosen at equal time increments throughout the event. Section 4 illustrates how these events are presented to a user for browsing.

3.3 Discussion

Typically, when shooting an event, a film maker will remain consistent in shooting style for the duration of an event. Thus, the use of FSMs to locate areas where particular features are dominant is central to the event retrieval process. For example, a conversation will typically contain speech throughout. When the speech ends, it indicates that the conversation itself is finished. So in order to locate a particular conversation, a searcher can use the speech FSM, which returns all of the areas with high amounts of speech, and possibly filter the returned vales to only retain sequences of shots with a high amount of repetition. This would remove many areas of speech that are not conversations (voiceovers etc.).

Each time the search system is used, the user must first decide on which FSM to use. This involves selecting a feature that occurs throughout the sought event. So, for example if a searcher wants to find a particular event, say a conversation that takes place in a moving car, he/she could use the non-static camera FSM to find all the non-static camera sequences (as there will be camera movement due to the moving car), and then filter the results by only accepting the sequences with a high amount of speech shots. In this way, a number of events will be returned, all of which contain high amounts of moving camera shots and high amounts of speech. The user can then browse the returned events and view the desired conversation. Note that another way of retrieving the same event would be to use the speech FSM and then filter the returned sequences by only allowing the ones with high amounts of non-static shots through.

Similarly, a searcher could be looking for an event in a nightclub, where two people's eyes meet and they stare at each other across the dance floor. To find this event, the sequences generated by the music FSM could be used, and then filtered by only accepting the sequences with a high amount of shot repetition, or by only accepting sequences with a high amount of static camera shots. Figure 1 illustrates this two-step approach. In the first step, the set of music potential sequences is selected. Secondly, these potential sequences are filtered by only retaining potential sequences with a user defined amount of static camera shots. This results in a retrieved event list as indicated in the figure.

In some cases no filtering may be desired, and the sequences returned by the FSM may be the desired events. For example, a film student may be interested in analysing how music is used in a particular film. The student could use the music FSM to find all of the areas where music is present, with no filtering required.

4 MOVIEBROWSER SYSTEM

A system, called the *MovieBrowser*, that allows searching movies as described above was developed. A sample screen from this system can be seen in Figure 2. On the top of the screen, the search feature selection interface can be seen. Using this panel, it is possible to select the FSM and filtering options as outlined in Section 3. Figure 2 shows the system presenting a set of search results. In this figure, the music FSM is chosen, and the filtering only retains sequences with high amounts of static camera shots (as can be seen by the checkbox in the top right). Five keyframes from each returned event are displayed. Some other information about the event (timing, number of shots etc.) is also displayed. It is possible to play the retrieved events in an external video player by clicking on the 'Play' button.

The MovieBrowser system also allows users to browse every shot in a particular movie. This simply displays the keyframe from each shot to the browser. Thus, if a user knows that a particular event occurs at a certain point in the movie, he/she can browse all shots in the movie in order to locate the event.

A number of movies were used as input to the MovieBrowser system. The movies were chosen to represent a broad range of origins, and span different geographical locations including America, Australia, Japan, England and Mexico. The test data in total consists of ten movies corresponding to over eighteen hours of video.

Table 1: Average time in seconds taken for each browsing method.

Average time for all tasks =	132.4 S
Average time for keyframe based method =	155.7 S
Average time for search based method =	98.9 S

5 EXPERIMENTAL ANALYSIS

5.1 Experimental Setup

In order to assess the effectiveness of searching for user defined events in a movie and presenting them to a user as an indexing solution, a set of experiments using the MovieBrowser were devised. The experimental process involves a number of users completing a set of tasks. Each task asks the user to retrieve a particular clip using either the proposed searching method, or the keyframe based browsing method. Each task was completed by four volunteers, two users using the keyframe based method and two users using the searching method. The time taken to complete each task was noted, so the results are directly comparable. In total, thirty tasks were created. For example, one task was "In the film High Fidelity, Find the part where Barry sings 'Lets get it on' with his band", while another was "In the film Reservoir Dogs, Find the getaway scene when Mr. Pink is running away from the cops".

An automatic timing program was implemented that recorded how long it takes a user to complete each task, and also to check whether users have located the correct event. Once a user has located a clip in the movie that he/she considers to be correct, they enter the time of the event into the system (which compares this time with the manually annotated start and end times of the tasks). If the supplied time is correct (i.e. between the start and end time of the relevant part of the movie) the time taken to complete the task is automatically recorded. If a user supplies an incorrect time, he/she is instructed to continue browsing in order to find the correct time of the event. If a user cannot complete a task, there is an option to give up searching. If this happens, an arbitrarily large completion time (five minutes) is assigned for the task. This heavily penalises non-completion of tasks. In order to compare results for different users, a pre-test questionnaire was created in which the volunteers were required to state which films they had seen before, and how long ago. Also, for equality, each user was supplied with thumbnail pictures of the main characters in each movie. Each user was briefly trained on the two retrieval methods, and then asked to begin their tasks.

Table 2: Average task completion times by browsing method, where the average results are shown for users that had, and had not seen the film previously.

Method Used	**Average Time For Unseen Movies (s)**	**Average Time For Seen Movies (s)**
Keyframe method	187.5	145.11
Search method	124.3	92.7

5.2 Results

The average task completion times are presented in Table 1. As can be seen, the average task completion time for the search based system is significantly lower than the average completion time for the keyframe based retrieval method. On two occasions users gave up whilst using the search based method, thus the maximum time was allocated (five minutes). These were the only two tasks that were not completed. In both cases, the users were not familiar with the movie, and although the results returned from searching contained the sought event, these were not recognised. Table 2 presents the average task completion times for users that had, and had not, viewed the movie previously. Predictably, when the movie had not been viewed previously, the retrieval times are longer. In both cases the search based method results in superior performance. The slower retrieval time for unseen movies is largely due to the aforementioned non-completion of two tasks, as, if the five minute time was not allocated, then the average time would have been 89.2 Seconds.

The search-based method performed well in most cases. When the users chose features appropriately it provided for efficient retrieval. The user selection of features is central to the success of each search. It was observed that, in general, users had no difficulty in selecting appropriate features.

The users that performed best using the search based method were the ones that did low amounts of filtering, and relied heavily on the FSMs. This usually meant a slightly larger amount of events to browse through, but, as the results show, the returned events can be navigated quickly. In some cases the best search method involved selecting a FSM with a feature that the user knows is prevalent in the desired event, and then browsing all results. The results for the search based method may improve if the user interface is altered, as some volunteers noted that it could be made more user friendly. Also, as the search based system requires the most user input, a larger amount of training time may help users familiarise themselves with the system and result in better search queries.

6 CONCLUSIONS

This paper presented a method of searching movies based on user defined events. A set of audiovisual features were chosen and extracted from movie files. A searching method was developed that facilitates retrieval of events in a movie using a combination of finite state machines and filtering. A system, called the MovieBrowser, implemented this searching method in the form of a graphical user interface. In order to assess the searching method, a set of user experiments were conducted which timed users completing a task using both the searching method and a keyframe based retrieval method. These experiments shows the effectiveness of this method of searching.

Future work will involve refining the filtering process in order to allow for different combinations of features. Also, the search interface will be altered to make it more user friendly. Although this paper focused on investigating audiovisual retrieval, the addition of textual data (obtained from movie subtitles) would significantly aid the retrieval process and result in improved performance.

ACKNOWLEDGEMENTS

The support of the Irish Research Council for Science, Engineering and Technology is gratefully acknowledged. This material is based on works supported by Science Foundation Ireland under Grant No. 03/IN.3/I361.

REFERENCES

Bordwell, D. and Thompson, K. (1997). *Film Art: An Introduction*. McGraw-Hill.

Chen, L., Rizvi, S. J., and Ötzu, M. (2003). Incorporating audio cues into dialog and action scene detection. In *Proceedings of SPIE Conference on Storage and Retrieval for Media Databases*, pages 252–264.

Kang, H.-B. (2003). Emotional event detection using relevance feedback. In *Proceedings of the International Conference on Image Processing*.

Lehane, B. and O'Connor, N. (2006). Workshop on image analysis for multimedia interactive services (wiamis), incheon, korea. In *Movie Indexing via Event Detection*.

Lehane, B., O'Connor, N., and Murphy, N. (2004a). Action sequence detection in motion pictures. In *The international Workshop on Multidisciplinary Image, Video, and Audio Retrieval and Mining*.

Lehane, B., O'Connor, N., and Murphy, N. (2004b). Dialogue scene detection in movies using low and mid-level visual features. In *International Workshop on Image, Video, and Audio Retrieval and Mining*.

Lehane, B., O'Connor, N., and Murphy, N. (2005). Dialogue scene detection in movies. In *International Conference on Image and Video Retrieval (CIVR), Singapore, 20-22 July 2005*, pages 286–296.

Leinhart, R., Pfeiffer, S., and Effelsberg, W. (1999). Scene determination based on video and audio features. In *In proceedings of IEEE Conference on Multimedia Computing and Systems*, pages 685–690.

Li, Y. and Kou, C.-C. J. (2001). Movie event detection by using audiovisual information. In *Proceedings of the Second IEEE Pacific Rim Conferences on Multimedia: Advances in Multimedia Information PRocessing*.

Li, Y. and Kou, C.-C. J. (2003). *Video Content Analysis using Multimodal Information*. Kluwer Academic Publishers.

Nam, J., Alghoniemy, M., and Tewfik, A. H. (1998). Audiovisual content-based violent scene characterization. In *Proceedings of International Conference on Image Processing (ICIP)*, volume 1, pages 351–357.

Rasheed, Z. and Shah, M. (2003). Scene detection in hollywood movies and tv shows. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Rui, Y., Huang, T. S., and Mehrotra, S. (1998). Constructing table-of-content for video. In *ACM Journal of Multimedia Systema*, pages 359–368.

Sundaram, H. and Chan, S.-F. (2000). Determining computable scenes in films and their structures using audio-visual memory models. In *ACM Multimedia 2000*.

Yeung, M. and Yeo, B.-L. (1996). Time constrained clustering for segmentation of video into story units. In *Proceedings of International Conference on Pattern Recognition*.

Yeung, M. and Yeo, B.-L. (1997). Video visualisation for compact presentation and fast browsing of pictorial content. In *IEEE Transactions on Circuits and Systems for Video Technology*, pages 771–785.

Zhai, Y., Rasheed, Z., and Shah, M. (2004). A framework for semantic classification of scenes using finite state machines. In *International Converence on Image and Video Retrieval*.

Zhou, J. and Tavanapong, W. (2002). Shotweave: A shot clustering technique for story browsing for large video databases. In *Proceedings of the Workshops XMLDM, MDDE, and YRWS on XML-Based Date Management and Multimedia Engineering-Revised Papers*.

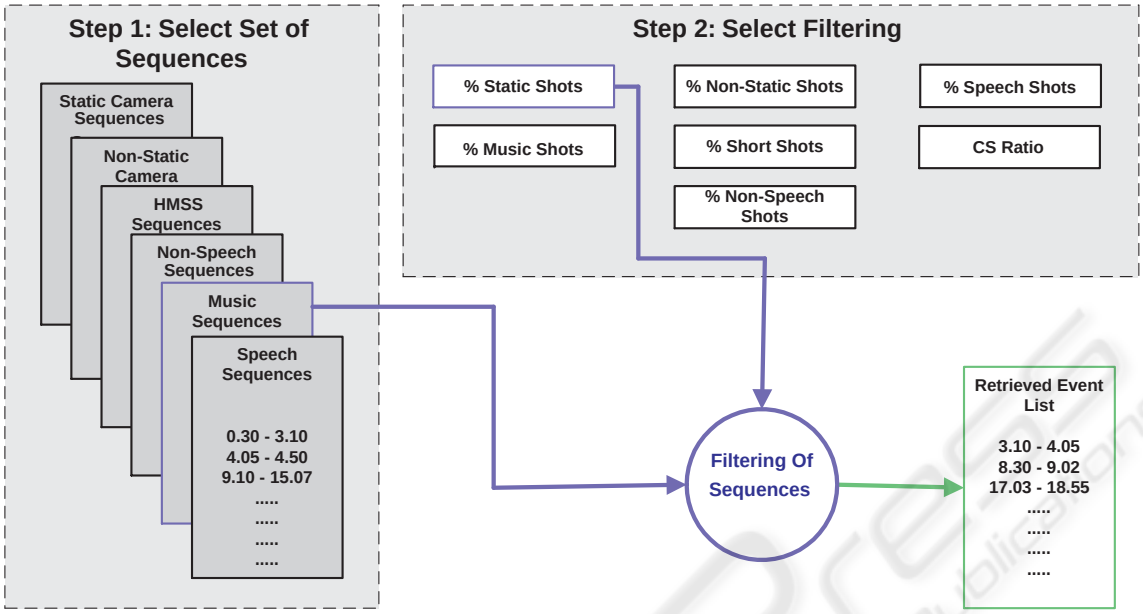

Figure 1: The user defined searching process.

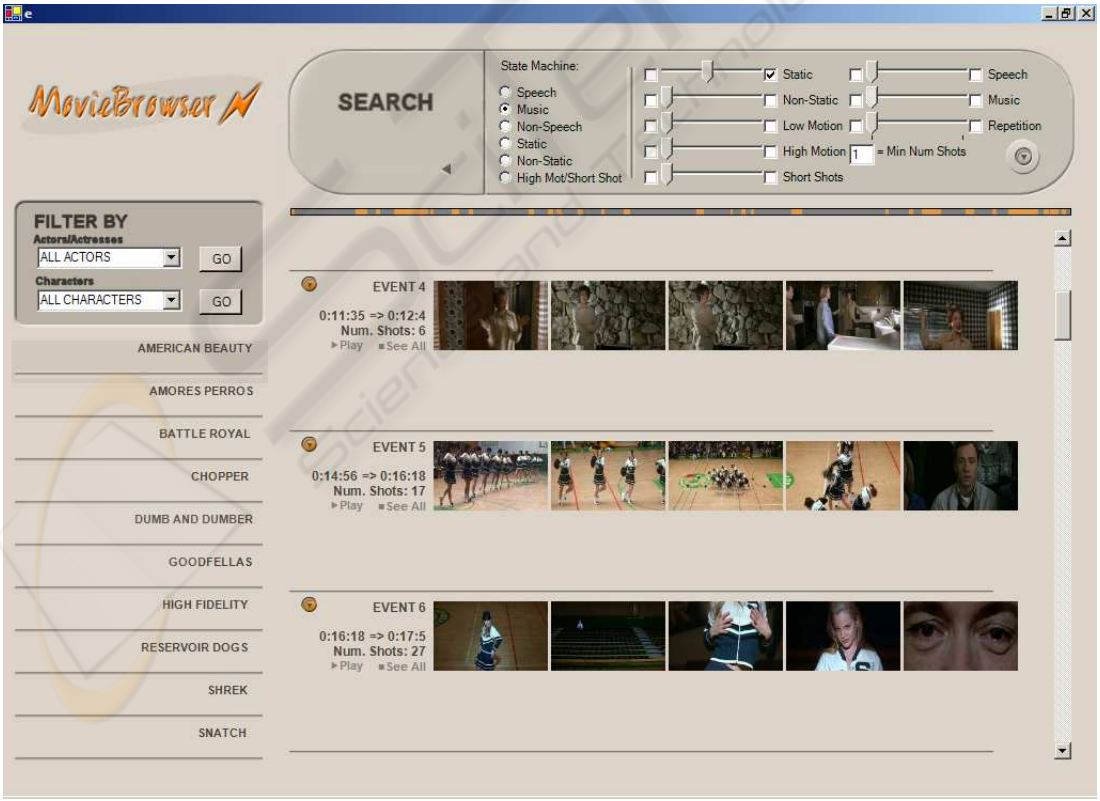

Figure 2: The search panel used to find events with a set of specified features.