Automated Video Program Summarization Using Speech Transcripts

Cuneyt M. Taskiran, *Member, IEEE*, Zygmunt Pizlo, Arnon Amir, *Senior Member, IEEE*, Dulce Ponceleon, *Member, IEEE*, and Edward J. Delp, *Fellow, IEEE*

***Abstract*—Compact representations of video data greatly enhances efficient video browsing. Such representations provide the user with information about the content of the particular sequence being examined while preserving the essential message. We propose a method to automatically generate video summaries using transcripts obtained by automatic speech recognition. We divide the full program into segments based on pause detection and derive a score for each segment, based on the frequencies of the words and bigrams it contains. Then, a summary is generated by selecting the segments with the highest score to duration ratios while at the same time maximizing the coverage of the summary over the full program. We developed an experimental design and a user study to judge the quality of the generated video summaries. We compared the informativeness of the proposed algorithm with two other algorithms for three different programs. The results of the user study demonstrate that the proposed algorithm produces more informative summaries than the other two algorithms.**

***Index Terms*—Speech transcripts, summarization evaluation, video summarization.**

I. Introduction

DERIVING compact representations of video sequences that are intuitive for users and let them easily and quickly browse large collections of video data is fast becoming one of the most important topics in content-based video processing. Such representations, which we collectively refer to as *video summaries*, rapidly provide the user with information about the content of the particular sequence being examined, while preserving the essential message. The need for automatic methods for generating video summaries is fueled both from the user and production viewpoints. With the proliferation of personal video recorder devices and hand-held cameras, users can easily generate many times more video footage than they can digest. On the other hand, in today's fast-paced news coverage, programs such as sports and news must be processed quickly for production or their value quickly diminishes. Such time constraints and the increasing number of services being offered places a large burden on production companies to process, edit, and distribute video material as fast as possible.

Summarization, for a video, audio, or text document, is a challenging and ill-defined task, since it requires the processing system to make decisions about high-level notions such as semantic content and relative importance of parts of a document with respect to each other. Objective evaluation of resulting media summaries is also a problem, since it is hard to derive quantitative measures of summary quality.

In this paper, we first present a review of the previous work in video summarization and video summary visualization techniques and clarify some of the terminology that is used in the literature. Second, we propose an automated method to generate video skims for information-rich video programs, such as documentaries, educational videos, and presentations, using statistical analysis based on speech transcripts that are obtained by automatic speech recognition (ASR) from the audio. We show how important phrases can be automatically extracted even from ASR text that has a high word error rate. These detected phrases may be used to augment current summarization methods and visualization schemes. Ideally one would like the generated summaries to be both detailed and covering most of the important points of the full program they were derived from. Clearly, if a media document is to be highly summarized, it is impossible to satisfy both of these constraints. Our summarization approach quantifies these two concepts and maximizes a weighted sum of both detail and coverage functions to obtain a tradeoff between the two. This approach enables the user to change the weights and regenerate the video summary of a program with more detail or more coverage, depending on a particular application.

Our third goal in this paper is to develop objective evaluation methods for video summaries. We evaluate summaries produced by three algorithms using a question and answer evaluation scheme and discuss other methods of summary evaluation.

The outline of the paper is as follows. We first characterize various aspects of video summarization methodology, such as different application domains and summary visualization schemes in Section II. The current state of the art in automatic video summarization is reviewed in Section III. Our proposed summarization algorithm is described in detail in Section IV. In Section V we review some of the video summary evaluation schemes that were used in the literature. We describe the design of our user study, present experimental results, and provide a detailed analysis of evaluation results in Section VI. Finally, conclusions are given in Section VII.

Manuscript received October 5, 2004; revised August 21, 2005; revised June 2005. The associate editor cordinating the review of this manuscript and approving it for publication was Dr. Chitra Dorai.

C. M. Taskiran is with the Multimedia Research Lab, Motorola Labs, Schaumburg, IL 60196 USA (e-mail: cuneyt.taskiran@motorola.com).

Z. Pizlo is with the Department of Psychological Sciences, Purdue University, West Lafayette, IN 47907-1364 USA (e-mail: pizlo@psych.purdue.edu).

A. Amir and D. Ponceleon are with the IBM Almaden Research Center, San Jose, CA 95120 USA (e-mail: arnon@almaden.ibm.com; dulce@almaden.ibm.com).

E. J. Delp is with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-2035 USA (e-mail: ace@ecn.purdue.edu).

Digital Object Identifier 10.1109/TMM.2006.876282

II. Characterization of Video Summaries

A. *Summary Application Domains*

The goal of video summarization is to process video programs, which usually contain semantic redundancy, and make them more interesting or useful for users. The properties of a video summary depends on the application domain, the characteristics of the sequences to be summarized, and the purpose of the summary. Some of the purposes that a video summary might serve are as follows.

- Intrigue the viewer to watch the whole video. Movie trailers, prepared by highly skilled editors and with high budgets, are the best examples of summaries of this type.
- Let the viewer decide if the complete program is worth watching. Summaries of video programs that may be used in personal video recorders are examples of this category of summaries. The user may have watched the episode that was recorded or may have already watched similar content so might not want to watch the program after seeing the summary.
- Help the viewer locate specific segments of interest. For example, in distance learning, students can skip parts of a video lecture that they are familiar with and, instead, concentrate on new material.
- Let users judge if a video clip returned by a video database system in response to their query is relevant. In content-based image database applications the results of a user query are shown as thumbnail images, which can be judged at a glance by the user for relevance to the query. Judging the relevance of query results is time-consuming for video sequences, since search results may contain long sequences containing hundreds of shots. Presenting the summaries of the results would be much more helpful.
- Enable users of pervasive devices, such as personal digital assistants, palm computers, or cellular phones, to view video sequences, which these devices otherwise would not be able to handle due to their low processing power. Using summaries also result in significant downloading cost savings for such devices. An example of such an application is described in [1], which makes use of annotations based on the MPEG-7 standard. With the increased consumption of video on cellular phones, some form of video summarization has become very important.
- Give the viewer all the important information contained in the video. These summaries are intended to replace watching the whole video. Executive summaries of long presentations or videoconferences would be an example of this type of summary.

The above list, while not exhaustive, illustrates the wide range of different types of video summaries one would like to generate. For most applications video summaries mainly serve two functions: the *indicative function*, where the summary is used to indicate what topics of information is contained in the original program; and the *informative function*, where the summaries are used to cover the information in the source program as much as possible, subject to the summary length. Clearly these two summary functionalities are not independent. Video summarization applications often will be designed to achieve a mixture of the two functionalities. The viewer's available time and the environment where the summary will typically be consumed, as well as the display characteristics of the device used, are also important factors in determining the type of summary to generate. For example, a summary of a lecture must cover the main results and conclusions, while a movie trailer must not reveal the punchline. Naturally, there is no single approach, either manual or automatic, that will apply to the generation of all types of video summaries.

B. *Types of Program Content*

An important distinction we make when considering video summarization algorithms is the type of the program content that will be summarized. For the purposes of video summarization we categorize video program content into two broad classes.

- *Event-based content*. Video programs of this type contain easily identifiable story units that form either a sequence of different events or a sequence of events and nonevents. Examples of the first kind of programs are talk shows and news programs where one guest or news story follows another and their boundaries are well-defined. The best example of programs where sequence of events and nonevents occur are sports programs. Here, the events may correspond to important instances in games, such as touchdowns, home runs, or goals.
- *Uniformly informative content*. These are programs which cannot easily be broken down to a series of events as event-based content. For this type of content, many parts of the program may be equally important for the user. Examples of this type of content are sitcoms, presentation videos, documentaries, soap operas, and home movies.

Note that the distinction introduced above is not clear cut. For example, for sitcoms one can define events according to audience laughter in the soundtrack. Movies are another example: most action movies have a clear sequence of action and nonaction segments.

For event-based content, since the types of events of interest are well-defined, one can use knowledge-based video event detection techniques. In this case, the processing is generally domain-specific and a new set of events and event detection rules must be derived for each application domain, which is a disadvantage. However, the summaries produced will be more reliable than those generated using general-purpose summarization algorithms. This class of algorithms is examined in Section III-D.

If domain-based knowledge of events is not available or if one is dealing with uniformly informative content, one has to resort to more general summarization techniques. This is generally done by first dividing the program to be summarized into a number of segments. Then, segments are selected for the summary either by deriving an importance score for each program segment and selecting segments with high scores, or by clustering similar segments together. The scores with high segments are selected for the summary. Another approach is to cluster frames from the video directly. Once the segments to be included in the summary are identified, each segment is represented using either keyframes extracted from the segment or portions of video within the segment.

C. Summary Visualization Methods

After the parts of the video to be included in the summary are determined, they have to be displayed to the user in an intuitive and compact manner. Depending on the desired summary type and length, information from all detected video segments in the full program may be used. Alternatively, importance scores may be assigned to each segment using various combinations of visual, audio, textual, and other features, and only portions or keyframes extracted from the segments with the highest scores may be included in the summary. We will use the term *video summarization* to denote any general method that can be used to derive a compact representation of a video program, and the term *summary visualization*, to refer to the method that is used to present the summary to the user. In this section we review some of the summary visualization approaches that have been proposed. These methods mainly fall into two categories: Video abstracts [2]–[6] based on keyframes extracted from video and video skims [7]–[12] where portions of the source video are concatenated to form a much shorter video clip. For video skims the duration of the summary is generally specified by the user in terms of the *summarization ratio* (SR), which is defined as the ratio of the duration of the video skim to the duration of the source video. For video abstracts, the user may specify the number of keyframes to be displayed.

1) Static Visualizations or Video Abstracts: The simplest static visualization method is to present one frame from each video segment, which may or may not correspond to an actual shot, in a storyboard fashion, sometimes accompanied by additional information such as timestamps and closed caption text. The problems with this method are that all shots appear equally important to the user and the representation becomes unpractically large for long videos.

One way to alleviate this problem is to rank the video segments and to display only the representative keyframes belonging to the segments with highest scores. In order to further reflect the relative scores of the segments the keyframes may be sized according to the score of the segment. This approach have been used in [2] and [5] where keyframes from segments are arranged in a "video poster" using a frame packing algorithm. The PanoramaExcerpts system [3] uses panoramic icons, which are obtained by merging consecutive frames in a shot, in addition to keyframes. In the Informedia project time ordered keyframes, known as filmstrips, were displayed as video abstracts [13].

Although video abstracts are compact, they present fundamental drawbacks, since they do not preserve the time-evolving nature of video programs. They are somewhat unnatural and hard to grasp for nonexperts, especially if the video is complex. Most techniques just present keyframes to the user without any additional metadata, like keywords, which can make the meaning of keyframes ambiguous. Finally, static summaries are not suitable for instructional and presentation videos, as well as teleconferences, where most shots contain a talking head, and most of the relevant information is found in the audio stream. These problems are addressed by dynamic visualization methods.

2) Dynamic Visualizations or Video Skims: In these methods the segments with the highest scores are selected from the source video and concatenated to generate a video skim. While selecting portions of the source video to be included in the video skim, care must be exercised to edit the video on long audio silences, which generally correspond to spoken sentence boundaries. This is due to the experimentally verified fact that users find it annoying when audio segments in a video skim begin in mid-sentence [7], [10].

3) Other Types of Visualizations: There are also some video browsing approaches which may be used to visualize video content compactly and hence may be considered a form of video summarization.

As part of the Informedia Project, Wactlar [14] proposes video collages, which are rich representations that display video data along with related keyframes, maps, and chronological information in response to a user query. In their BMOVIES system Vasconcelos and Lippman [15] use a Bayesian network to classify shots as action, close-up, crowd, or setting based on motion, skin tone, and texture features. The system generates a timeline that displays the evolution of the state of the semantic attributes throughout the sequence. Taskiran *et al.* [16] cluster keyframes extracted from shots using color, edge, and texture features and present them in a hierarchical fashion using a similarity pyramid. In the CueVideo system, Amir *et al.* [17] provide a video browser with multiple synchronized views. It allows switching between different views, such as storyboards, salient animations, slide shows with fast or slow audio, content-based accelerating fast playback, and full video while preserving the corresponding point within the video between all different views. Ponceleon and Dieberger [18] propose a grid, which they call the movieDNA, whose cells indicate the presence or absence of a feature of interest in a particular video segment. When the user moves the mouse over a cell, a window shows a representative frame and other metadata about that particular cluster. A system to build a hierarchical representation of video content is discussed in Huang *et al.* [19], where audio, video, and text content are fused to obtain an index table for broadcast news. Aner *et al.* [20] compose mosaics of scene backgrounds in sitcom programs. The mosaic images provide a compact static visual summary of the physical settings of scenes. By matching keyframes with mosaics, shots are clustered into scenes based on their physical locations, resulting in a compact scene-based representation of the program and an automatic reference across multiple episodes of the same sitcom.

III. Previous Approaches to Video Summary Generation

In this section we investigate various algorithms that have been proposed to generate video summaries.

A. Speedup of Playback

A simple way to compactly display a video sequence is to present the complete sequence to the user but increase the playback speed. A technique known as time scale modification can be used to process the audio signal so that the speedup can be made with little audio distortion [21]. The compression allowed by this approach, however, is limited to a summarization ratio (SR) of 0.4–0.7, depending on the particular program genre [22]. Based on a comprehensive user study, Amir *et al.* report that for most program genres and for novice users, a SR of

0.59 can be achieved without significant loss in comprehension. However, this SR value is not adequate for most summarization applications, which often require a SR between 0.1 and 0.2.

B. Techniques Based on Frame Clustering

Dividing a video sequence into segments and extracting one or more keyframes from each segment was long recognized as one of the simplest and most compact way of representing video sequences. For a survey of keyframe extraction techniques the reader is referred to [23]. These techniques generally focus on the image data stream only. Color histograms, because of their robustness, have generally been used as the features for clustering.

One of the earliest work in this area is by Yeung and Yeo [2], which uses time-constrained clustering of shots. Each shot is labeled according to the cluster it belongs to and three types events, dialog, action, and other, are detected based on these labels. Representative frames from each event are then selected for the summary. The Video Manga system by Uchihashi *et al.* [5] clusters individual video frames using YUV color histograms. Iacob *et al.* [24] propose a similar technique. However, in their approach video frames are first divided into rectangles, whose sizes depend on the local structure, and YUV histograms are extracted from these rectangles. Ratakonda *et al.* [25] extract keyframes for summaries based on the area under the cumulative action curve within a shot, where the action between two frames is defined to be the absolute histogram difference between color histograms of the frames. They then cluster these keyframes into a hierarchical structure to generate a summary of the program.

Ferman and Tekalp [26] select keyframes from each shot using the fuzzy c-clustering algorithm, which is a variation of the k-means clustering method, based on alpha-trimmed average histograms extracted from frames. Cluster validity analysis is performed to automatically determine the optimal number of keyframes from each shot to be included in the summary. This summary may then be processed based on user preferences, such as the maximum number of keyframes to view, and cluster merging may be performed if there are too many keyframes in the original summary.

The approaches proposed in [23] and [27] both contain a two-stage clustering structure, which is very similar to the method used in [26] but instead of performing shot detection segments are identified by clustering of video frames. Hanjalic and Zhang use features and cluster validity analysis techniques that are similar to those in [26]. Farin *et al.* [27] propose a two-stage clustering technique based on luminance histograms extracted from each frame in the sequence. First, in an approach similar to time-constrained clustering, segments in the video sequence are located by minimizing segment inhomogeneity, which is defined as the sum of the distances of all frames within a segment to the mean feature vector of the segment. Then, the segments obtained are clustered using the Earth-Mover's distance [28]. Yahiaoui *et al.* [29] first cluster frames based on the L_1 distance between their color histograms using a procedure similar to k-means clustering. Then, a set of clusters is chosen as to maximize the coverage over the source video sequence.

An application domain which poses unique challenges for summarization is home videos. Since home videos generally have no plot and contain little or no editing, the editing patterns and high level video structure, which offer strong cues in the summarization of broadcast programs, are absent. However, home videos are inherently time-stamped during recording. Lienhart [9] proposes a summarization algorithm where shots are clustered at four time resolutions using different thresholds for each level of resolution using frame time stamps. Very long shots, which are common in home videos, are shortened by using the heuristic that during important events audio is clearly audible over a longer period of time than less important content.

C. Techniques Based on Frame Clustering by Dimensionality Reduction

These techniques perform a bottom-up clustering of the video frames selected at fixed intervals. A high dimensional feature vector is extracted from each frame and this dimensionality is then reduced either by projecting the vectors to a much lower dimensional space [30], [31] or by using local approximations to high dimensional trajectories [4], [6]. Finally, clustering of frames is performed in this lower dimensional space.

DeMenthon *et al.* [4] extract a 37-dimensional feature vector from each frame by considering a time coordinate together with the three coordinates of the largest blobs in four intervals for each luminance and chrominance channel. They then apply a curve splitting algorithm to the trajectory of these feature vectors to segment the video sequence. A keyframe is extracted from each segment. Stefanidis *et al.* [6] propose a similar system; however, they split the three-dimensional (3-D) trajectories of video objects instead of feature trajectories.

Gong and Liu [30] use singular value decomposition (SVD) to cluster frames evenly spaced in the video sequence. Each frame is initially represented using 3-D RGB histograms, which results in 1125-dimensional frame feature vectors. Then, SVD is performed on these vectors to reduce the dimensionality to 150 and clustering is performed in this space. Portions of shots from each cluster are selected for the summary. Cooper and Foote [31] sample the given video sequence at a rate of 1 frame/s and extract a color feature vector from each extracted frame. The cosine of the angle between feature vectors is taken to be the similarity measure between them and a non-negative similarity matrix is formed between all pairs of frames. Non-negative matrix factorization (NMF) [32], is used to reduce the dimensionality of the similarity matrix.

D. Techniques Using Domain Knowledge

As discussed in Section II-B, if the application domain of the summarization algorithm is restricted to event-based content, it becomes possible to enhance summarization algorithms by exploiting domain-specific knowledge about important events. Summarization of sports video has been the main application for such approaches. Sports programs lend themselves well for automatic summarization for a number of reasons. First, the interesting segments of a program occupy a small portion of the whole content; second, the broadcast value of a program falls off rapidly after the event so the processing must be performed in near real-time; third, compact representations of sports programs have a large potential audience; finally, there are often clear markers, such as cheering crowds, stopped games, and replays that signify important events.

Summarization of soccer programs has received a large amount of attention recently, see [33] for a survey of work in this area. Li *et al.* [34] develop a general model for sports programs where events are defined to be the actions in a program that are replayed by the broadcaster. The replay is often preceded by a close-up shot of the key players or the audience. They apply their approach to soccer videos where they detect close-up shots by determining if the dominant color of the shot is close to that of the soccer field. Ekin and Tekalp [33] divide each keyframe of a soccer program into nine regions and use features based on color content to classify shots into long, medium, and close-up shots. They also detect shots containing the referee and the penalty box. Goal detection is performed similar to [34] by detecting close-up shots followed by a replay. Cabasson and Divakaran [35] use audio peaks and a motion activity measure to detect exciting events in soccer programs. Based on the heuristic that the game generally stops after an exciting event, they search the program for sequences of high motion followed by low motion. If an audio peak is detected near such a sequence it is marked as an event and included in the summary.

Incorporating domain knowledge can be very helpful even when summarizing uniformly informative content. For example, He *et al.* [10] have proposed algorithms based on heuristics about slide transitions and speaker pitch information to summarize presentation videos.

E. Techniques Using Closed-Captions or Speech Transcripts

For some types of programs a large portion of the informational content is carried in the audio. News programs, presentation videos, documentaries, teleconferences, and instructional videos are some examples of such content. Using the spoken text to generate video summaries becomes a powerful approach for these types of sequences. Content text is readily available for most broadcast programs in the form of closed captions. For sequences, like presentations and instructional programs, where this information is not available, speech recognition may be performed to obtain the speech transcript. Once the text corresponding to a video sequence is available, one can use methods of text summarization to obtain a text summary. The portions of the video corresponding to the selected text may then be concatenated to generate the video skim. Processing text also provides a high level of access to the semantic content of a program that is hard to achieve using image content only.

Agnihotri *et al.* [36] search the closed-caption text for cue words to generate summaries for talk shows. Cues such as "please welcome" and "when we come back" in addition to domain knowledge about program structure are used to segment programs into parts containing individual guests and commercial breaks. Keywords are then used to categorize the conversation with each guest into a number of predetermined classes such as *movie* or *music*. In their ANSES system, Pickering *et al.* [37] use key entity detection to identify important keywords in closed-caption text. Working under the assumption that story boundaries always fall on shot boundaries, they perform shot detection followed by the merging of similar shots based on the similarity of words they contain. They then detect the nouns in text using a part of speech tagger and use lexical chains [38] to rank the sentences in each story. The highest scoring sentences are then used to summarize each news story.

An example of a technique that uses automatic speech recognition (ASR) is the one proposed by Taskiran *et al.* [7]. The usage of ASR makes their method applicable to cases where the closed-caption text is not available, such as presentations or instructional videos. In their approach the video is first divided into segments at the pause boundaries. Then, each segment is assigned a score using term frequencies within segments. Using statistical text analysis, dominant word pairs are identified in the program and the scores of segments containing these pairs are increased. The segments with highest scores are selected for the summary.

F. Approaches Using Multiple Information Streams

Most current summarization techniques focus on processing one data stream, which is generally image data. However, multimodal data fusion approaches, where data from images, audio, and closed-caption text are combined, offer the possibility to greatly increase the quality of the video summaries produced. In this section, we look at a few systems that incorporate features derived from multiple data streams.

The MoCA project [12], one of the earliest systems for video summarization, uses color and action content of shots, among other heuristics, to obtain trailers for feature films. The Informedia project constitutes a pioneer and one of the largest efforts in creating a large video database with search and browse capabilities. It uses integrated speech recognition, image processing, and natural language processing techniques for the analysis of video data [11], [14]. Video segments with significant camera motion, and those showing people or a text caption are given a higher score. Audio analysis includes detection of names in the speech transcript. Audio and video segments selected for summary are then merged while trying to maintain audio/video synchronicity.

Ma *et al.* [39] propose a generic user attention model by integrating a set of low-level features extracted from video. This model incorporates features based on camera and object motion, face detection, and audio. An attention value curve is obtained for a given video sequence using the model and portions near the crests of this curve are deemed to be interesting events. Then, heuristic rules are employed, based on pause and shot boundaries, and the SR value, to select portions of the video for the summary. Another model-based approach is the computable scene model proposed by Chang and Sundaram [40], which uses the rules of film-making and experimental observations in the psychology of audition.

IV. Summary Generation Using Speech Transcripts

As can be seen from the survey of video summarization approaches given in Section III, most methods for video summarization do not make use of one of the most important sources of information in a video sequence, the spoken text or the natural-language content; Informedia, CueVideo and the system proposed in [19] being some exceptions. Speech transcripts are readily available for TV programs in the form of closed captions. For video programs like seminars and instructional programs, where this information is not available, speech recognition may be performed on audio to obtain a transcript. Once the text corresponding to a video sequence is available, one can use methods of text summarization to obtain a text summary. The portions of the video corresponding to the selected text are then concatenated to generate the video skim.

The techniques used in text summarization may be roughly divided into two groups.

- *Statistical analysis based on information-retrieval techniques*. In this approach, the problem of summarization is reduced to the problem of ranking sentences or paragraphs in the given text according to their likelihood of being included in the final summary. This likelihood is a high level semantic quality of a sentence, compromising notions of meaning, relative importance of sentences, style, and the like. In these techniques, instead of employing natural language understanding methods, various features are extracted from the text which are correlated with the "abstract-worthiness" of a sentence, and the ranking is done using a combination of these features.
- *Natural Language Processing (NLP) analysis based on information-extraction techniques*. This paradigm, making use of techniques from artificial intelligence, entails performing a detailed semantic analysis of the source text to build a source representation designed for a particular application. Then a summary representation is formed using this source representation and the output summary text is synthesized [41].

Methods using statistical processing to extract sentences for the summary often generate summaries that lack coherence. These methods also suffer from the *dangling anaphora problem*. Anaphoras are pronouns, demonstratives, and comparatives like "he," "this," or "more," which can only be understood by referring to an antecedent clause appearing before the sentence in which these words occur. If the antecedent clause has not been selected for the summary, anaphoras may be confusing for the user. Although techniques based on NLP generate better summaries, the knowledge base required for such systems is generally large and complex. Such systems are specific to a narrow domain of application and are hard to generalize to other domains. Furthermore, attempts to directly apply NLP methods on text decoded by speech recognition usually fail because of the poor quality of the detected text, which is reflected in a high word error rate and lack of punctuation. We use the statistical analysis approach in processing the speech transcripts. A block diagram of the proposed system is shown in Fig. 1.

A. Program Segmentation Using Audio and Speech

The first step in our video summarization system is the segmentation of the given program into a number of segments. One approach to segmentation is to use shots as segments, thereby dividing the video into visually coherent groups. However, shot boundaries are not aligned with audio boundaries in most types of video programs, that is, a shot boundary may occur in the middle of a sentence in the audio. In user studies of summarization algorithms, it has been found that users find it annoying

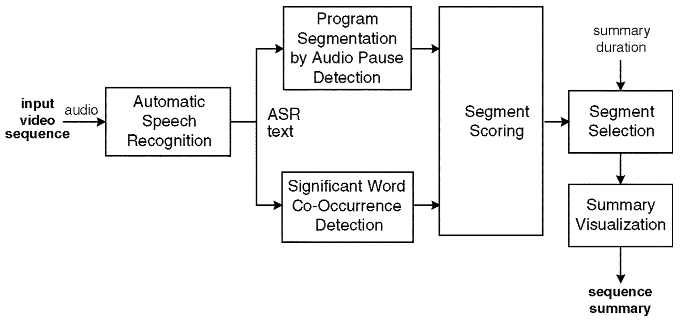

Fig. 1. Block diagram of the proposed video summarization system.

when audio segments in the summary begin in the middle of a sentence [10]. Therefore, instead of using shot boundaries to segment video into segments, we use long interword pauses in audio, which generally coincide with sentence boundaries. We define the duration of video between two consecutive long pauses to be a *segment*, which form the smallest processing unit for our summarization system. An advantage of this approach to video segmentation is that it avoids having very long segments which could occur if shots are used as segments, due to long shots that commonly occur in videos of presentations and meetings.

There are many techniques available to detect prolonged silence in speech. However, we use a simple yet efficient heuristic using the speech recognition text generated by the large vocabulary IBM ViaVoice speech recognition system [42]. This text contains the time stamp for each word detected by the speech recognition system together with its estimated duration. Based on these timestamps we calculate the duration between words w_i and w_{i+1} as $p_i = t_{i+1} - (t_i + l(w_i))$, where t_i and t_{i+1} are the start timestamps of words w_i and w_{i+1} in the video, and $l(w_i)$ is the estimated duration of word w_i. The measured interword durations of two video sequences, a wildlife documentary and a seminar, are shown in Fig. 2. The durations between words in the documentary are generally larger since this is an educational program with a well-defined script and the speaker speaks at a relaxed pace for the audience to easily follow. On the other hand, the pauses in the seminar video are generally quite short, corresponding to free-flowing natural speech, with vocalized pauses, such as "ah" or "um," when the speaker is thinking about how to phrase a thought. The variability between these plots imply that a global threshold cannot be used to detect segment boundaries.

We use a sliding window scheme to robustly detect long pauses in audio that is similar to the cut detection method given in [43]. Let $\mathbf{W} = \{p_{i-m}, \ldots, p_i, \ldots, p_{i+m}\}$ be a sliding window of size $2m+1$ centered at the inter-word duration that is being currently considered for marking as a long pause. We declare the pause p_i to be a segment boundary if the following conditions are met.

1) The value p_i is the maximum within $\mathbf{W}$.
2) The value p_i is also n times larger than the mean value of $\mathbf{W}$, not including the value p_i, that is

$$p_i \geq n \left(\frac{\sum_{j=i-m, j \neq i}^{i+m} p_j}{2m} \right).$$

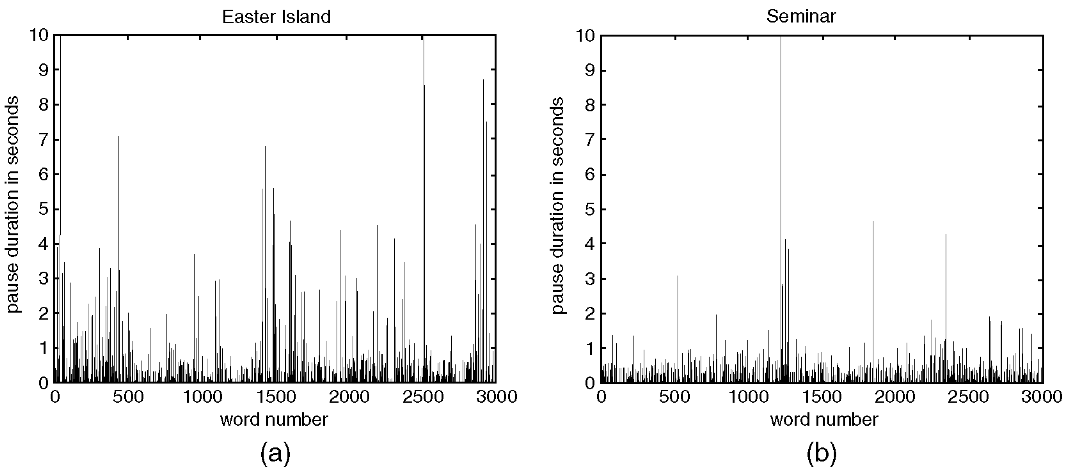

Fig. 2. The measured interword durations for portions of two programs with different types of content. (a) Documentary; (b) seminar.

The parameter n controls detection sensitivity, increasing it will decrease the detection rate but also decrease the false alarm rate. The value of m controls the granularity of the resulting segments by setting a lower limit on the number of words a segment can have. A small value of m causes the video to be divided into a large number of small segments, which may make the resulting video skim seem "choppy" and hard to follow. On the other hand, using too large a value for m will cause segments to be too long. We have used a window size of 21, i.e., $m = 10$, and $n = 2$ in our experiments. The value $m = 10$ was selected based on pilot studies that we have performed. This value worked well for a wide variety of documentary programs. However, a different value may be necessary for different types of content, such as news or sports.

After the long pauses in the program are detected using the above procedure the video sequence is divided into segments using these pauses as boundaries. Each word detected by the ASR system in the program is assigned to a segment based on the time of occurrence. In our system we use what is referred to as the "bag of words" approach, where all sentence structure and word ordering is ignored.

B. Calculation of Segment Scores

1) Segment Scoring Using Speech Recognition Text: After the video is segmented using pause boundaries, as described in Section IV-A, we calculate a score for each segment based on the words they contain. Not all the words belonging to a segment are used in deriving the segment score; some words, called *stop-words*, which have grammatical function but contribute little to the information content of the segment are ignored. Generally, stopwords include articles, pronouns, adjectives, adverbs, and prepositions. We have used a list of 371 stopwords in our system.

A score is calculated for each of the words that remain in a segment after the stopwords are eliminated. We use a scoring method that is related to a family of word scoring schemes commonly used information retrieval, referred to as term frequency—inverse document frequency (tf.idf) methods. In these scoring schemes, the score of a word is calculated as a function of *term frequency*, $n_{i,w}$, the number of times a word has appeared in a video program, reflecting how salient the word is within the program and *document frequency*, the number of segments that a word appears in, indicating how semantically focused the word is. Based on these ideas, for each word w in segment i we compute the score, $s_{i,w}$, which measures the statistical importance of w using the formula [44]

$$s_{i,w} = \frac{(k_1+1)n_{i,w}}{k_1\left[(1-k_2)+k_2\frac{L_i}{AL}\right]+n_{i,w}} \log\frac{N}{n_w} \tag{1}$$

where

$n_{i,w}$	number of occurrences of word w in segment i;
n_w	total number of occurrences of word w in the video sequence;
L_i	number of words in segment i;
AL	average number of words per segment in the video sequence;
N	total number of segments in the video sequence;
k_1, k_2	tuning parameters.

The constant k_1 determines the sensitivity of the first term to changes in the value of the term frequency, $n_{i,w}$. If $k_1 = 0$ this term reduces to the counting function which is 1 if word w occurs in segment i and 0 otherwise; if k_1 is large, this term becomes nearly linear in $n_{i,w}$. Since not all segments have the same number of words, the segment score has to be normalized by the number of words in the segment. A simple normalization would be to divide $n_{i,w}$ by L_i, which corresponds to the case of $k_2 = 1$ in (1). This normalization is based on the assumption that if two segments are about the same topic but of different lengths, this is just because the longer is more wordy. Extensive experiments reported in [44] suggest the values $k_1 = 2$ and $k_2 = 0.75$, which we have used in our experiments.

After the scores for individual words are computed using (1), the score for segment i is computed as the sum of the scores for the words it contains as

$$S_i = \sum_{w \in \text{segment } i} s_{i,w}. \tag{2}$$

2) Detection of Dominant Word Co-Occurrences: One problem with using the words detected by a speech recognition

TABLE I
TWENTY CO-OCCURRENCES WITH THE HIGHEST LOG-LIKELIHOOD VALUES FOR THREE DOCUMENTARY PROGRAMS

MarkTwain	$-2\log\lambda$	*SeaHorse*	$-2\log\lambda$	*BrooklynBridge*	$-2\log\lambda$
mark–twain	26.64	gets–pregnant	17.52	air–compressed	28.64
history–shakespeare	15.78	setting–term	15.72	new–york	18.86
forced–times	15.78	lives–tiny	15.72	favorite–institute	16.06
ball–put	15.78	change–color	15.72	cure–decided	16.06
got–talked	15.78	shallows–shore	15.72	favorite–polytechnic	16.06
twelve–miles	15.78	female–takes	13.14	polytechnic–institute	16.06
boat–brothers	15.78	animal–kingdom	13.14	greatest–existence	16.06
ambition–steamboat	15.78	unfold–wild	12.40	brooklyn–york	13.88
aside–spring	15.78	care–pregnancy	12.40	top–chamber	12.74
american–bowl	15.66	half–human	12.40	size–sons	12.74
people–understood	14.28	breeding–season	12.40	compressed–water	12.44
sam–clemens	13.60	watch–wild	12.40	civil–washington	12.00
father– nancial	12.46	carefully–know	12.40	people–views	11.26
dollars–jones	12.46	carries–eggs	12.40	inside–required	11.26
long–want	12.46	cash–takes	12.40	city–half	11.26
go–literature	12.46	arrested–need	12.40	back–feet	10.22
look–night	12.46	arrested–think	12.40	feet–level	10.22
race–understood	12.46	females–males	11.74	existence–structure	10.22
nearby–summer	12.46	sea–horses	11.68	greatest–structure	10.22
friends–nearby	12.46	consider–long	10.90	structure–plans	10.22

system is that, the detection accuracy may be low, especially for video programs containing rare words and phrases. In this study we used the IBM ViaVoice large vocabulary continuous speech recognition (LVCSR) real-time system with the Broadcast News language model of 60 000 words [45]. The word error rate of LVCSR systems may vary substantially depending on the content, from 19% on prepared speech read by an anchor in a studio as reported by [45], to up to 35–65% for a wide variety of real-world spontaneous speech data that may include combinations of speech with background music, noise, degraded acoustics, and non-native speakers. The difficulty is that speech recognition programs are generally trained using a large, well-balanced collection of content in which phrases like "animal kingdom," "breeding season," "compressed air," or "Moore's Law" are rare. However, such terms and proper names play an important role in understanding a program and should be taken into account by the video summarization algorithm when selecting segments for the summary. Such groups of words are referred to as *collocations*, which are defined as sequences of two or more consecutive words that have characteristics of a syntactic or semantic unit [46]. Since we ignore the ordering of words in segments, we generalize the definition of a collocation and include pairs of words that are strongly associated together but do not necessarily occur in consecutive order. We refer to such pairs of words as *co-occurrences*. We measure the association of words by deriving a likelihood measure that measures how dependent the use of one word in a segment is on the presence on the other word.

Detecting co-occurrences in the speech transcript of a video program may be viewed as a hypothesis test: given the occurrence distribution of the words w_1 and w_2 in the segments detected in the program, the null hypothesis is that they occur independently and the alternate hypothesis is that the occurrences are dependent. The likelihood ratio for this hypothesis test may be written as

$$\lambda = \frac{\max_p P(w_1, w_2; p|H_0)}{\max_{p_1,p_2} P(w_1, w_2; p_1, p_2|H_1)} \tag{3}$$

where H_0 and H_1 are the null and alternate hypotheses, respectively, and p_1 and p_2 are the estimates of the occurrences of w_1 and w_2. The word counts that summarize the occurrence distribution of w_1 and w_2 are given by

$a = n(w_1, w_2)$	number of segments where w_1 and w_2 both appear;
$b = n(w_1, \neg w_2)$	number of segments where w_1 appears but not w_2;
$c = n(\neg w_1, w_2)$	number of segments where w_2 appears but not w_1;
$d = n(\neg w_1, \neg w_2)$	number of segments where neither w_1 nor w_2 appear.

The maximum likelihood estimates for the probabilities are then given by $p = (a+b)/N$, $p_1 = a/(a+c)$, and $p_2 = b/(b+d)$. Substituting these values in (3) and assuming that the probability that a word will appear in a segment is modeled using a binomial distribution, the log-likelihood ratio becomes [47]

$$\begin{aligned}\log\lambda = {} & (a+b)\log(a+b) + (a+c)\log(a+c) \\ & + (b+d)\log(b+d) + (c+d)\log(c+d) - a\log a \\ & - b\log b - c\log c - d\log d - N\log N. \end{aligned} \tag{4}$$

Using the likelihood ratio in (4) to find significant word co-occurrences in speech transcripts has many advantages over other methods [46]. It is more accurate for sparse data than other methods and, since the quantity $-2\log\lambda$ is asymptotically χ^2 distributed, the critical value to reject the null hypothesis of independency for a given confidence level can easily be calculated.[1]

In Table I, we list the 20 co-occurring word pairs that have the highest $-2\log\lambda$ values for three documentary programs we

[1]It should be pointed out that the log-likelihood may not indicate if the co-occurrence words are frequent or not. For example, for the extreme case of w_1 and w_2 always occurring together, we have $c = d = 0$ and $\log\lambda = a\log(a/(a+d)) + d\log(d/(a+d))$. In this case we will have similar log-likelihood values both when the words are common $(a \gg d)$ and when they are rare $(a \ll d)$.

have used in our experiments. *MarkTwain* is documentary detailing Mark Twain's biography, *SeaHorse* includes information about the life and mating habits of sea horses, and *BrooklynBridge* is about the building of the Brooklyn Bridge in New York City. From this table we observe that the word pairs give valuable information about the contents of the programs. For a confidence level of $\alpha = 0.05$, the critical value of the χ^2 distribution with one degree of freedom is 7.88 so for all these word pairs the null hypothesis can be rejected and there is strong association between the words in the pair.

In our system the log-likelihood ration in (4) is calculated for all possible nonstopword pairs in the speech transcript. Then 30 pairs with the highest $-2\log\lambda$ values are chosen to be the significant co-occurrences for the program, which are used to update segment scores according to the equation

$$S'_i = S_i C_w^{L_i^{sig}} \tag{5}$$

where L_i^{sig} is the number of significant co-occurrences that segment i contains and C_w is a constant. We have used the value $C_w = 1.2$ in our system. This score increase is based on the assumption that the segments including a larger number of significant co-occurrences are more important.

C. *Segment Selection for the Skim Using a Greedy Algorithm*

After the segment scores are calculated using constituent nonstopwords and updated using the significant word co-occurrences detected in the transcript, they are used to select the segments that will form the video skim. The duration of the skim is specified by the user in terms of the *summarization ratio*, f, which is defined as the ratio of the duration of the video skim to the duration of the source video. We then have

$$T_{\text{summary}} = fT_{\text{full}}. \tag{6}$$

Since the score of a segment is proportional to its "semantic importance," our goal in generating the skim is to select those segments that maximize the cumulative score for the resulting summary while not exceeding T_{summary}. This may be viewed as an instance of the 0-1 knapsack problem (KP) which is defined as follows [48]

$$\begin{aligned} &\text{maximize} \quad \sum_{i=1}^{N} S'_i x_i \\ &\text{subject to} \quad \sum_{i=1}^{N} T_i x_i \le T_{\text{summary}}, \end{aligned} \tag{7}$$

where T_i is the duration of the segment i, N is the number of segments in the program, and the binary variable x_i is defined as

$$x_i = \begin{cases} 1, & \text{if segment } i \text{ is selected for the skim} \\ 0, & \text{otherwise.} \end{cases}$$

Although efficient branch-and-bound algorithms exist [48] that can be used to solve this problem, in our system we have chosen to implement a greedy algorithm to solve the 0–1 KP due to its fast execution speed. Although the solution found by the greedy algorithm will be suboptimal, the difference between the greedy solution and the optimal solution has been shown to be small in many cases [49].

We define the *efficiency* of segment i to be the ratio of its score to its duration, i.e., $e_i = S'_i/T_i$. Clearly, we would like to include segments with high efficiency in the summary. Therefore, the first step in solving the constrained maximization problem given in (7) is to sort the segments in a program according to their efficiencies to have a list with the property $e_i \ge e_j$ when $i < j$. Assuming that k segments with the largest efficiency values are selected for the summary, the duration of the summary will be

$$\overline{T}_k = \sum_{i=1}^{k} T_i.$$

The greedy selection process proceeds as follows. Starting from $k = 1$, segments are added to the summary as long as $T_k \le T_{\text{summary}} - \overline{T}_{k-1}$, that is, as long as there is still time left in the summary to accommodate the next segment. Let the index of the first segment that cannot be included in the summary be k^*. Then, we have

$$\overline{T}_{k^*-1} \le T_{\text{summary}} < \overline{T}_{k^*}.$$

The greedy algorithm then selects the segments $\{1, \ldots, k^* - 1\}$ for the summary. Note that this solution leaves an unused capacity $T_{\text{summary}} - \overline{T}_{k^*-1}$. In order to include as many segments in the summary as possible, we have modified the basic greedy algorithm so that once it encounters a segment that is too long to fit in the unused time for the summary, it continues to search the list of segments to find a shorter segment that will fit. Finally, the selected segments are sorted according to their start times and concatenated to generate the video skim.

D. *Increasing Summary Coverage Using a Dispersion Measure*

Our previous summary evaluation results [7] suggest that the summary segment selection algorithm described in Section IV-C generates summaries that tend to be focused on a few important points in the full program. Such summaries are detailed but have low overall program coverage. When the goal of video summarization is to preserve as much information about the original program as possible in the video summary, the advantage of this algorithm over mechanical summarization procedures, such as randomly selecting the segments to be included in the summary, may be limited. This is the coverage—detail dilemma of summarization discussed in Section I, i.e., it is impossible to maximize both the coverage and detail of the summary of a program.

One way to alleviate this problem is to add another term to the maximization problem in (7) that is a function of the coverage of a summary and maximize both the cumulative score and coverage for the generated summary. In order to formalize the notion of coverage we introduce a measure of *dispersion*, $d(\mathcal{S})$, for a summary, $\mathcal{S}$. We want the quantity $d(\mathcal{S})$ to be small when $\mathcal{S}$ contains segments that are clustered together in the full program, and to be large when they are distributed uniformly across the full program.

Let the original video be divided into P equal temporal intervals. Given a set of segments, $\mathcal{S}$, we estimate their temporal distribution over the original program by counting the number of segments contained in each temporal piece:

$$h(l) = \#\text{segments with } b_i \in [l, l+1), \quad l = 0, \ldots, P-1$$

where b_i is the start time of segment i in $\mathcal{S}$. Then, we define the dispersion of the set $\mathcal{S}$ to be

$$d(\mathcal{S}) = -\sum_{l=1}^{P} h(l) \log h(l). \tag{8}$$

Several other measures of dispersion of elements of a set can be defined, such as the Gini or the misclassification measures, which are commonly used to measure the impurity of a node in decision tree growing [50]. The particular dispersion function selected is not vital to the algorithm and any of the other choices can also be used.

At each iteration k of the segment selection algorithm, we want to add the segment that leads to the greatest increase in the dispersion value of the current summary, $\mathcal{S}^k$. To this end, we define the increase in dispersion caused by adding segment i to $\mathcal{S}^k$ as

$$\Delta d_i^k = d\left(\mathcal{S}^k \cup \{i\}\right) - d(\mathcal{S}^k). \tag{9}$$

Then, the dispersion score of segment i at iteration k is defined as

$$d_i^k = e^{-C_d \Delta d_i^k}, \tag{10}$$

where C_d is a constant used to accentuate the difference in dispersion between segments. The algorithm is not very sensitive to the particular value of C_d selected, we have used the value $C_d = 20$ in our experiments.

In order to be able to compare the word-based score for a segment and its dispersion score, at each iteration we normalize each of these scores by the score of the segment that has the highest value for that score. We then employ a greedy algorithm, selecting the segment Σ^* that has the highest combined score using

$$\Sigma^* = \arg\max_i \left(\alpha_{\text{word}} \tilde{S}'_i + (1 - \alpha_{\text{word}}) \tilde{d}_i^k\right), \tag{11}$$

where $\tilde{S}'$ and $\tilde{d}_i^k$ are the normalized word-based and dispersion scores, respectively, and $w_{\text{word}} \in [0, 1]$, is the weight for the word-based score. By changing the value of w_{word} one can change the relative importance of detail versus coverage for the generated summaries. Based on our pilot studies, we used the value $w_{\text{word}} = 0.7$, which was found to constitute a good tradeoff between detail and coverage. In a summarization system these values would be user adjustable to balance coverage and detail.

V. Evaluation of Video Summaries

Since automatic video summarization is still an emerging field, serious questions remain concerning the appropriate methodology in evaluating the quality of the generated summaries. Most video summarization studies do not include any form of quantitative summary evaluation. Evaluation of the quality of automatically generated video summaries is a complicated task because it is difficult to derive objective quantitative measures for summary quality. In order to be able to measure the effectiveness of a video summarization algorithm one first needs to define features that characterize a good summary, given the specific application domain being studied. As discussed in Section II, summaries for different applications will have different sets of desirable attributes. Hence, the criteria to judge summary quality will be different for different application domains.

Automated text summarization dates back at least to Luhn's work at IBM in the 1950s [51], which makes it the most mature area of media summarization. We will apply the terminology developed for text summary evaluation to evaluation of video summaries. Methods for the evaluation of summaries can be broadly classified into two categories: *intrinsic* and *extrinsic* evaluation methods [52], [53]. In intrinsic evaluation methods, the quality of the generated summaries is judged directly based on the analysis of summary. The criteria used may be user judgment of fluency of the summary, coverage of key ideas of the source material, or similarity to an "ideal" summary prepared by humans. On the other hand, in extrinsic methods the summary is evaluated with respect to its impact on the performance for a specific information retrieval task.

For event-based content, e.g., a sports program, where interesting events are unambiguous, summaries might be judged on their coverage of these events in the source video. Two such evaluations are given in [26] and [33]. Ekin and Tekalp [33] give precision and recall values for goal, referee, and penalty box detection, which are important events in soccer games. In Ferman and Tekalp's study [26], the video summary was examined to determine, for each shot, the number of redundant or missing keyframes in the summary. For example, if the observer thought that an important object in a shot was important but no keyframe contained that object, this resulted in a missed keyframe. Although they serve as a form of quantitative summary quality measure, the event detection precision and recall values given in these studies do not reflect the quality of the summaries from the user's point of view.

For uniformly informative content, where events may be harder to identify, different evaluation techniques have been proposed. He *et al.* [10] determine the coverage of summaries of key ideas from presentation videos by giving users a multiple choice quiz derived from the full program video before and after watching a video skim extracted from it. The quizzes consist of questions prepared by the presentation speakers and are assumed to reflect the key ideas of the presentation. The quality of the video skims were judged by the increase in quiz scores. A similar technique was used by Taskiran *et al.* [17] in evaluating video skims extracted from documentaries. The quiz method has some drawbacks: First, it was found that this approach may have difficulty differentiating between different summarization

TABLE II
INFORMATION ABOUT THE FULL-LENGTH DOCUMENTARY PROGRAMS USED IN THE EXPERIMENT

Sequence name	**length (min.)**	**Content description**
Seahorse	22.10	Life and mating habits of seahorses [54]
MarkTwain	21.19	Detailed biography of MarkTwain [55]
WhyDogsSmile	22.53	Investigation of mother-child bond and other emotions in animals [56]

algorithms depending on program content [7], [10]. Second, it is not clear how quiz questions can be prepared in an objective manner, except, perhaps, by authors of presentations who are usually not available. Finally, the concept of a "key idea" in a video program is ambiguous and may depend on the particular viewer watching the skim. An interesting extrinsic evaluation method was used by Christel *et al.* [11]. In this study video skims extracted from documentaries were judged based on the performance of users on two tasks: fact-finding, where users used the video skims to locate video segments that answered specific questions; and gisting, where users matched video skims with representative text phrases and frames extracted from source video.

In intrinsic evaluation of text summaries, generally summaries created by experts are used [52]. Using a similar approach would be much more costly and time consuming for video sequences. Another scheme for evaluation may be to present the segments from the original program to a large number of viewers and let them select the segments they think should be included in the summary, thereby generating a ground truth. This seems to be a promising approach although agreement among human subjects becomes an issue for this scheme. A commonly used intrinsic evaluation method is to have users rate the skims based on subjective questions, e.g., "Was the summary useful?" and "Was the summary coherent?" Such surveys were used in [9]–[11].

VI. EXPERIMENTS AND RESULTS

The experiments we have performed to detect the differences in the quality of summaries generated by different algorithms are described in detail in the following sections. We conducted experiments based on both intrinsic and extrinsic evaluation methods that were defined in Section V. The summary quality measure we used was the ratio of information contained about the full program in the video summary, i.e., the more information a summary contains about the full program, the higher its quality was deemed to be. Given this summary quality criterion, a natural method to test the effectiveness of video summarization algorithms is the *question and answer method*, where viewers answer questions derived from the full program just by watching the summary and the number of correctly answered questions is accepted as a quantitative measure of summary quality. We used this task as the extrinsic summary evaluation method.

As the intrinsic evaluation scheme, we tested how many of the key points of the full program were covered by the summaries by determining how many of the questions extracted from the full program were in the generated summaries. We also asked the subjects who participated in our experiment their assessment of the summaries they watched.

A. Sequences and Algorithms Used in the Experiments

We chose three documentary programs to use in our study. The original lengths of these programs were 60 min, 210 min, and 93 min for the *Seahorse*, *MarkTwain*, and *WhyDogsSmile* documentaries, respectively. Using a summarization ration of 0.1 would have resulted in long summaries which would have limited the number of summaries that could be presented to subjects in our experiments. Because of this reason, and to have all full programs be approximately of the same length, we have selected a 20 min portion of each program near the beginning and used these as our full-length programs from which summaries were extracted. In broadcast documentaries a summary of the whole documentary is sometimes presented at the beginning of the program. Since this would have interfered with our evaluation, we have taken care not to include such programs in the selected documentaries.

We selected program content to minimize the chance that the subjects participating in the study have prior knowledge about the programs. Information about the three documentary programs used in our experiments is given in Table II. All sequences were processed using the CueVideo system to obtain speech transcripts.

Three algorithms, abbreviated as FREQ, RAND, and DEFT, were used on the full programs to generate three different skims for each full program. We denote the summary of the Seahorse program using the RAND algorithm by $\mathrm{Seahorse}+\mathrm{RAND}$, and similarly for the other summaries. The algorithm FREQ is the summary generation algorithm based on word-frequency and dispersion scores derived from program segments, as described in Section IV. The RAND algorithm detects the segments in the program, using the same pause detection algorithm as was used for FREQ, but does not calculate any segment scores and randomly selects the segments to be included in the summary. The DEFT, or default, algorithm is the simplest video summarization algorithm possible, consisting of subsampling the video program temporally at fixed intervals. The DEFT algorithm divides the full program into $\lfloor T_{\mathrm{summary}}/T_s \rfloor$ temporal intervals, and selects the first T_s s of each interval for the summary. Our preliminary tests suggest a value of $T_s = 5$ s to be the minimum value that viewers feel comfortable with; values smaller than this lead to "choppy" summaries that are hard to understand. DEFT is the simplest possible summarization algorithm, consisting of uniform temporal sampling of the program, and has no dependency on program content. Nevertheless it is a popular strategy used by viewers who want to quickly get the gist of a program, e.g., using $8\times$ or $16\times$ skips in DVD viewing. The only difference between RAND and DEFT is breaking on audio pauses. In [11] pilot testing of the video summaries was used to derive the conclusion that users preferred summary segments based on audio pauses. However, this factor was not tested in

TABLE III
THE ORDERING OF THE THREE VIDEO SKIMS WATCHED BY SUBJECTS IN EACH GROUP IN THE EXPERIMENT

Group	**£rst summary**	**second summary**	**third summary**
Group 1	MarkTwain+FREQ	Seahorse+RAND	WhyDogsSmile+DEFT
Group 2	WhyDogsSmile+RAND	MarkTwain+DEFT	Seahorse+FREQ
Group 3	Seahorse+DEFT	WhyDogsSmile+FREQ	MarkTwain+RAND

isolation to determine its effect on an extrinsic task. We included the RAND algorithm to study this factor.

A factor that can influence evaluation results is the value of the summarization ratio used to obtain the video skims. The evaluation results of the same summarization system can be significantly different when it is used to generate summaries of different lengths [52]. In our experiments we have kept the summarization ratio constant at $f = 0.1$, that is, the skims generated are approximately one-tenth the duration of the original programs. This value was also used in other video summarization studies and represents a reasonable amount of compaction for practical applications.

B. Experimental Design

The 48 subjects participating in the experiment were randomly divided into three groups of 16. Each subject watched three video skims. In order to minimize learning and other unforeseen cross-over effects, both the order of the content and the algorithms were different for the three groups, as shown in Table III. Therefore, each subject watched skims generated from all programs by all three algorithms. Subjects were Purdue University students and employees. Each subject received $10 for participation in the experiment and participated in the experiment individually. Subjects used a personal computer on which the graphical user interface for the experiment was run. After watching each video skim, the subjects first answered three questions about the quality of the summary they just watched. Then, they answered ten multiple choice questions derived from the full program and containing important facts about the content presented in the skim. Each question offered four answer choices. Only one choice was correct for each question. While watching the summaries the users were not able to pause the video, jump to a specific point in the video, or take notes. No time limit was imposed for completing the experiment.

The questions that were presented to the subjects were determined by two judges. One judge was the first author and the other judge was naive about the algorithms used. Each judge independently marked the parts of the closed-caption transcripts of the programs that they deemed were the important points of the programs without watching any of the summaries. The intersection of these two lists of marked locations were used to generate the questions with the constraint that the questions be as uniformly distributed over the full program as possible. Ten questions were generated for each program, together with one correct and three incorrect multiple choice answers. The questions were kept simple to ensure that if the answer to a question appears in a summary, then a viewer who watches that summary can easily answer the question correctly. All the questions used in the experiments and source program transcripts are available from the first author.

TABLE IV
STATISTICS FOR THE NUMBER OF CORRECT ANSWERS FOR TEN QUESTIONS AND $n = 48$ SUBJECTS FOR THE QUESTION AND ANSWER TASK. SCORES WERE AGGREGATED FOR THE THREE EXPERIMENTAL GROUPS

	FREQ	RAND	DEFT
estimated mean, $\hat{\mu}$	6.542	5.313	4.604
estimated standard deviation, $\hat{\sigma}$	1.650	1.652	1.865
estimatedstandarderror, $\hat{\sigma}_M = \hat{\sigma}/\sqrt{n}$	0.238	0.238	0.269

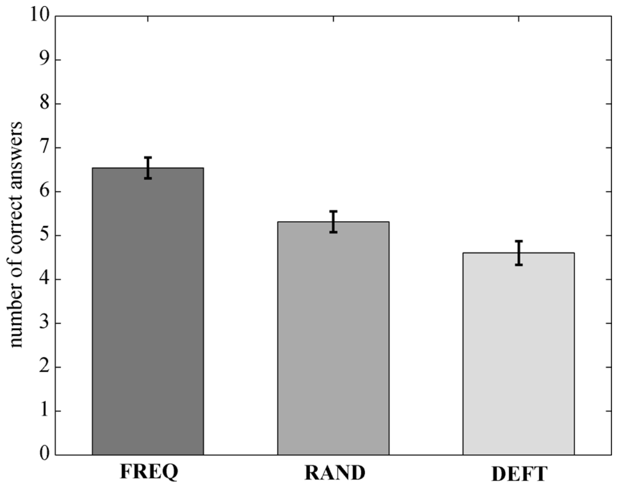

Fig. 3. Graphical representation of the data in Table IV.

TABLE V
PAIRWISE p-VALUES USING THE TWO-SIDED t-TEST TO TEST THE STATISTICAL SIGNIFICANCE OF THE DIFFERENCE IN SCORES BETWEEN ALGORITHMS

	FREQ	RAND
RAND	$4{:}348 \times 10^{-4}$	
DEFT	$5{:}189 \times 10^{-7}$	$5{:}183 \times 10^{-2}$

C. Results for the Questions and Answer Task

Statistics for the performance of the subjects in the question and answer task are shown in Table IV and represented graphically in Fig. 3. The number of correct answers out of ten questions that the subjects scored after watching the summaries generated by the same algorithm were summed for the three experimental groups to obtain these total results. In Fig. 3, the error bars represent an interval of $\hat{\mu} \pm \hat{\sigma}_M$, where $\hat{\mu}$ is the estimated mean number of correct answers for the subjects, the quantity $\hat{\sigma}_M = \hat{\sigma}/\sqrt{n}$ is the standard error for the mean, which gives a measure how much sampling fluctuation the mean will show, $\hat{\sigma}$ is the estimated standard deviation of subject scores, and $n = 48$ is the number of subjects.

From these results we can see that the FREQ algorithm performed better than RAND and DEFT. The performance of RAND and DEFT were comparable, with RAND outperforming DEFT slightly. This result was expected since neither of these algorithms took program content into account while

TABLE VI
STATISTICS FOR THE NUMBER OF CORRECT ANSWERS FOR TEN QUESTIONS AND $n = 16$ SUBJECTS IN EACH GROUP FOR EACH PROGRAM

	Seahorse			**MarkTwain**			**WhyDogsSmile**		
	FREQ	RAND	DEFT	FREQ	RAND	DEFT	FREQ	RAND	DEFT
estimated mean, $\hat{\mu}$	5.563	5.313	4.750	7.063	5.000	5.375	7.000	5.625	3.688
estimated standard deviation, $\hat{\sigma}$	1.896	1.922	1.915	1.181	1.461	1.857	1.414	1.586	1.493
estimated standard error, $\hat{\sigma}_M = \hat{\sigma}/\sqrt{n}$	0.474	0.481	0.479	0.295	0.365	0.464	0.354	0.397	0.373

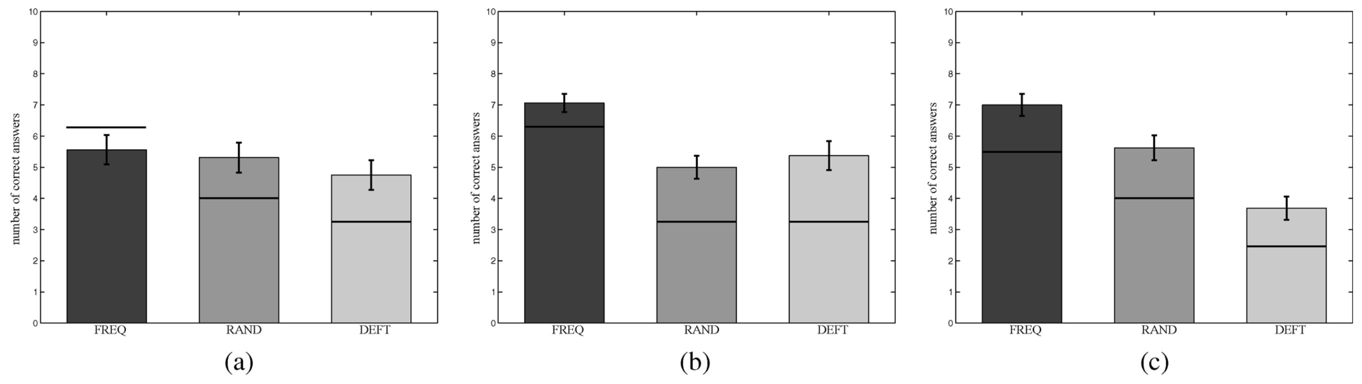

Fig. 4. Graphical representation of the data in Table VI. (a) Seahorse results; (b) MarkTwain results; (c) DogSmile.

generating the summary. In order to assess the statistical significance of the results in Table IV, we list pairwise comparison p-values in Table V obtained using a two-sided t-test. The t-values were calculated with the assumption of equal variances for the three populations. The p-value refers to the probability of incorrectly rejecting the hypothesis that two populations have identical mean values. From Table V, we see that one can safely conclude that FREQ and DEFT represent different populations, i.e., these two algorithms produce summaries of different quality, since the probability that this conclusion is wrong is less than 10^{-6}. The true p-value is larger, however. The reason is that Table IV shows the results of testing three hypotheses, not one, and the probability of incorrectly rejecting at least one hypothesis out of three is higher than the probability of rejecting one out of one. In particular, if one takes $p = 0.05$ as the cutoff value below which the difference between two populations is considered significant, then each pair of hypotheses in Table IV should be evaluated by using a cutoff value equal to $p = 0.05/3 = 0.0167$. We see from the Table V that using this criterion the differences in scores between FREQ and RAND, and FREQ and DEFT are statistically significant, while the difference between RAND and DEFT is not statistically significant.

In order to check the experimental assumption that subjects had little prior knowledge about the topics in the programs, we asked them to rate their prior knowledge of the program content after watching each summary on a scale of 1 to 5, where 1 indicates that the subject was not familiar with the subject at all while 5 indicates that the subject was very familiar with the subject. The average ratings for the three movies were found to be 1.813, 1.583, and 1.792 for *Seahorse*, *MarkTwain* and *DogSmile*, respectively. These figures agree well with the experimental assumption that the subjects had little prior knowledge about the program content that they were tested on.

In our previous study with a smaller group of subjects using the question and answer method [7], we found that the results for this evaluation method were correlated with the particular documentary program used to extract the summaries. In order to investigate the effect of program content on the performance of the algorithms, the mean correct scores that the subjects obtained from the multiple choice questions for the three documentary programs are compared in Table VI. The graphical representation of the means and standard errors for the data in Table VI is given in Fig. 4. From this figure we see that the performance of the subjects who watched the summaries produced by the FREQ algorithm are consistently larger than the scores for subjects who watched summaries from the other two algorithms, implying that our algorithm produced more informative summaries. Specifically, assuming the value $p = 0.05$ as the cutoff value for deciding statistical significance, and correcting for the fact that we are performing nine tests, statistically significant differences were found between FREQ—RAND and FREQ—DEFT for MarkTwain, and among all three algorithms for DogSmile.

The theoretical maximum scores, $c_{\max}$, that the subjects can obtain for the three programs are shown as horizontal lines on the bars in Fig. 4. These values were calculated using the relation

$$c_{\max} = c_{\text{inc}} + 0.25(10 - c_{\text{inc}}),$$

where c_{inc} is the number of answers determined by the judges included in the summary of each program, which are listed in Table VIII, and 0.25 is the probability of getting a correct answer by guessing. This simple model assumes that the subjects have perfect memory and full attention, therefore always correctly answering the questions that were covered by the summary. As seen in Fig. 4, in all cases but one the mean score of the subjects is higher than the theoretical maximum. These differences are

TABLE VII
NUMBER OF CORRECT ANSWERS FOR EACH ALGORITHM GROUPED ACCORDING TO WHETHER A SUBJECT WAS NATIVE SPEAKER OF ENGLISH (NS) OR NOT (NNS)

	FREQ		RAND		DEFT	
	NS	NNS	NS	NNS	NS	NNS
estimated mean, $\hat{\mu}$	6.778	6.400	5.611	5.133	5.056	4.333
estimated standard deviation, $\hat{\sigma}$	2.074	1.354	1.685	1.634	1.893	1.826
estimated standard error, $\hat{\sigma}_M = \hat{\sigma}/\sqrt{n}$	0.489	0.247	0.397	0.298	0.446	0.333

TABLE VIII
THE NUMBER OF IMPORTANT ITEMS COVERED BY EACH SUMMARY, OUT OF TEN ITEMS EXTRACTED FROM THE FULL PROGRAMS

	FREQ	RAND	DEFT
Seahorse	5	2	1
MarkTwain	5	1	1
DogSmile	4	2	0

TABLE IX
RESULTS FOR SUMMARY ASSESSMENT QUESTIONS AGGREGATED OVER THREE SEQUENCES FOR $n = 48$ SUBJECTS. THE QUESTIONS WERE 1) "I FOUND THE SUMMARY TO BE CLEAR AND EASY TO UNDERSTAND" AND 2) "I FEEL THAT I CAN SKIP WATCHING THE WHOLE PROGRAM BECAUSE I WATCHED THIS SUMMARY." SUBJECTS RATED ON A SCALE OF 1 (STRONGLY DISAGREE) TO 5 (STRONGLY AGREE)

	FREQ	RAND	DEFT
Question I median response	3	3	2
Question II median response	2	2	1

mainly due to two factors: First, while we tried to minimize previous knowledge about program content in the experiment, it is impossible to totally eliminate it. For some questions, subjects have used their previous knowledge to determine the correct answer even though it was not included in the summary. Second, although the answer to a particular question may not be included in a summary, it is still possible for subjects to use visual cues and inference to "guesstimate" the correct answer.

We should note that our question preparation strategy of distributing the questions as uniformly as possible over the total duration of the original program introduced a bias toward algorithms that maximize coverage rather than detail, i.e., toward RAND and DEFT. Since the value of the coverage weight used in FREQ, $1 - \alpha_{\text{word}} = 0.3$, is less than half of the detail weight value, $\alpha_{\text{word}} = 0.7$, FREQ maximizes detail much more than coverage. This means that the performance of the subjects for RAND and DEFT actually overestimates the quality of these algorithms.

Finally, we noticed some systematic differences in scores between subjects who were native speakers of English and those who were not. Statistics for the scores, separated according to native speaker status, are listed in Table VII Although not found to be statistically significant, these results nevertheless suggest that there may be differences in performance for native and non-native speakers for the questions and answer task. This factor should be taken into account when designing video summarization evaluations using this task. From Table VII we can see that the largest difference between the scores occurs for the DEFT algorithm. Due to the poor quality of the video summaries produced by DEFT, information gathering is the most challenging for this algorithm. The language understanding capabilities of viewers need to be used to the maximum for DEFT summaries, which accounts for the relatively large difference between native and non-native speaker performances for this algorithm.

D. Results for the Intrinsic Evaluation

We examined each of the nine summaries for each documentary—algorithm pair and determined how many answers each summary contains. These numbers are tabulated in Table VIII. As can be seen from this table, the information covered by summaries generated using FREQ contain significantly more information about the full programs compared to the other two algorithms. Since the summarization ratio was $f = 0.1$ we would expect the random summaries to contain one answer on the average. The results for RAND and FREQ in Table VIII agree well with this prediction.

As the second intrinsic evaluation scheme, after watching each summary we asked the subjects to answer two assessment questions. The questions were "I found the summary to be clear and easy to understand" and "I feel that I can skip watching the whole program because I watched this summary." Subjects rated both of these statements on a scale of 1 to 5, 1 being "strongly disagree" and 5 being "strongly agree". The medians of the subjects ratings for the three algorithms, aggregated over the three programs, are shown in Table IX. We believe that the interpretation of such subjective quality assessments may be difficult and error-prone. A controlled study focusing on intrinsic tasks is required to study such assessments in-depth. However, from the data in Table IX we can make the following observation: Although the informativeness of the summaries generated by FREQ was much better than both RAND and DEFT, as evidenced by the results in Table VIII, we do not observe such a large difference in the subjective quality assessment between these algorithms. This result implies that the correlation between the extrinsic and subjective evaluation of the summaries is not very strong.

VII. CONCLUSIONS

In this paper, we proposed an algorithm to automatically summarize video programs. We use concepts from text summarization, applied to transcripts obtained using automatic speech recognition. The log-likelihood ratio criterion was used for detecting significant co-occurring words and was found to be a powerful tool in identifying important phrases in video programs. A measure of summary dispersion over the full program was derived and was shown to be effective in managing the tradeoff between detail and coverage of the generated video summaries.

In the second part of this work we developed an experimental design and a user study to judge the quality of the generated video summaries. We used both extrinsic and intrinsic schemes to evaluate the summaries generated using a summarization factor of 0.1 by our algorithm and two random segment selection algorithms. For extrinsic evaluation, we used a question answering strategy, where the questions are derived from the full programs while the subjects who participated in our experiment watched only the summaries generated from those programs before answering the questions. For the intrinsic evaluation of the generated summaries we counted the number of important points about the programs each summary contains. We also asked the subjects to assess the quality of the summaries.

For the extrinsic evaluation user experiment, the mean number of correct answers out of ten questions for 48 subjects were 6.542, 5.313, and 4.604 for our algorithm and the two random algorithms, respectively. The differences in these mean scores were found to be statistically significant, implying that our algorithm produced more informative summaries.

For the intrinsic evaluation scheme, the number of important items that were contained in the summaries produced by our algorithm were significantly higher than those produced by the random algorithms, which had performance at chance level, as expected. We found that the subjective quality assessment was comparable for all three algorithms, although it was slightly better for FREQ and RAND that both take pauses in audio into consideration while segmenting the full program. The results imply that the extrinsic evaluation results and subjective quality assessment is not strongly correlated. Users also have a slight preference to algorithms that have segment boundaries on audio pauses, which is similar to the results obtained in [11]

The algorithm that we proposed only uses the text content of the programs in generating the summaries. In order to improve the quality of summaries, the algorithms need to take into account other modalities, e.g., the image content and audio features, such as prosody. Currently we are investigating methods to incorporate such modalities in our summarization scheme, such as clustering the keyframes extracted from a program and maximizing coverage intelligently by maximizing the number of clusters that have segments in the summary.

Acknowledgment

The authors would like to thank Dr. E. Ekmekçi for her help in conducting the user study for this paper.

References

[1] B. L. Tseng, C.-Y. Lin, and J. R. Smith, "Video summarization and personalization for pervasive mobile devices," in *Proc. SPIE Conf. Storage and Retrieval for Media Databases 2002*, San Jose, CA, Jan. 23–25, 2002, vol. 4676, pp. 359–370.

[2] M. M. Yeung and B.-L. Yeo, "Video visualization for compact presentation and fast browsing of pictorial content," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 5, pp. 771–785, Oct. 1997.

[3] Y. Taniguchi, A. Akutsu, and Y. Tonomura, "Panorama excerpts: extracting and packing panoramas for video browsing," in *Proc. ACM Multimedia*, Seattle, WA, Nov. 9–13, 1997, pp. 427–436.

[4] D. DeMenthon, V. Kobla, and D. Doerman, "Video summarization by curve simplification," in *Proc. ACM Multimedia Conf.*, Bristol, England, Sep. 12–16, 1998, pp. 211–218.

[5] S. Uchihashi, J. Foote, A. Girgenson, and J. Boreczky, "Video manga: generating semantically meaningful video summaries," in *Proc. ACM Multimedia'99*, Orlando, FL, Oct. 30–Nov. 5 1999, pp. 383–392.

[6] A. Stefanidis, P. Partsinevelos, P. Agouris, and P. Doucette, "Summarizing video datasets in the spatiotemporal domain," in *Proc. Int. Workshop on Advanced Spatial Data Management (ASDM'2000)*, Greenwich, U.K., Sep. 6–7, 2000.

[7] C. M. Taskiran, A. Amir, D. Ponceleon, and E. J. Delp, "Automated video summarization using speech transcripts," in *Proc. SPIE Conf. Storage and Retrieval form Media Databases 2002*, San Jose, CA, Jan. 20–25, 2002, vol. 4676, pp. 371–382.

[8] J. Oh and K. A. Hua, "An efficient technique for summarizing videos using visual contents," in *Proc. IEEE Int. Conf. Multimedia and Expo (ICME'2000)*, New York, Jul. 30–Aug. 2 2000.

[9] R. Lienhan, "Dynamic video summarization of home video," in *Proc. SPIE Conf. Storage and Retrieval for Media Databases 2000*, San Jose, CA, Jan. 2000, vol. 3972, pp. 378–389.

[10] L. He, E. Sanocki, A. Gupta, and J. Grudin, "Auto-summarization of audio-video presentations," in *Proc. 7th ACM Int. Multimedia Conf.*, Orlando, FL, Oct. 30–5 Nov. 1999, pp. 489–498.

[11] M. G. Christel, M. A. Smith, R. Taylor, and D. B. Winker, "Evolving video skims into useful multimedia abstractions," in *Proc. ACM Computer–Human Interface Conference (CHI'98)*, Los Angeles, CA, Apr. 18–23, 1998, pp. 171–178.

[12] S. Pfeiffer, R. Lienhart, S. Fischer, and W. Effelsberg, "Abstracting digital movies automatically," *J. Vis. Commun. Image Represent.*, vol. 7, no. 4, pp. 345–353, Dec. 1996.

[13] M. Christel, A. G. Hauptman, A. S. Warmack, and S. A. Crosby, "Adjustable filmstrips and skims as abstractions for a digital video library," in *Proc. IEEE Conf. Advances in Digital Libraries*, Baltimore, MD, May 19–21, 1999.

[14] H. D. Wactlar, "Informedia—search and summarization in the video medium," in *Proc. IMAGlNA 2000 Conf.*, Monaco, Jan. 31–Feb. 2 2000.

[15] N. Vasconcelos and A. Lippman, "Bayesian modeling of video editing and structure: semantic features for video summarization and browsing," in *Proc. IEEE Int. Conf. Image Processing*, Chicago, IL, Oct. 4–7, 1998.

[16] C. Taskiran, J.-Y. Chen, A. Albiol, L. Torres, C. A. Bouman, and E. J. Delp, "Vibe: a compressed video database structured for active browsing and search," *IEEE Trans. Multimedia*, vol. 6, no. 1, pp. 103–118, Feb. 2004.

[17] A. Amir, S. Srinivasan, and D. Ponceleon, , A. Rosenfeld, D. Doermann, and D. DeMenthon, Eds., "Efficient video browsing using multiple synchronized views," in *Video Mining*. Boston, MA: Kluwer, Aug. 2003.

[18] D. Ponceleon and A. Dieberger, "Hierachical brushing in a collection of video data," in *Proc. 34th Hawaii Int. Conf. System Sciences (HICSS'34)*, Maui, HI, Jan. 3–6, 2001.

[19] Q. Huang, Z. Liu, A. Rosenberg, D. Gibbon, and B. Shahraray, "Automated generation of news content hierarchy by integrating audio, video, and text information," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Phoenix, AZ, Mar. 15–19, 1999, pp. 3025–3028.

[20] A. Aner, L. Tang, and J. R. Kender, "A method and browser for cross-referenced video summaries," in *Proc. IEEE Int. Conf. Multimedia and Expo (ICME)*, Lausanne, Switzerland, Aug. 26–29, 2002.

[21] A. Amir, D. B. Ponceleon, B. Blanchard, D. Petkovic, S. Srinivasan, and G. Cohen, "Using audio lime scale modification for video browsing," in *Proc. 33rd Annual Hawaii Int. Conference on System Sciences (HICSS-33)*, Maui, HI, Jan. 4–7, 2000.

[22] B. Arons, "Speechskimmer: a system for interactively skimming recorded speech," *ACM Trans. Computer Human Interact.*, vol. 4, no. 1, pp. 3–38, 1997.

[23] A. Hanjalic and H. Zhang, "An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 8, pp. 1280–1289, 1999.

[24] S. M. Iacob, R. L. Lagendijk, and M. E. Iacob, "Video abstraction based on asymmetric similarity values," in *Proc. SPIE Conf. Multimedia Storage and Archiving Systems IV*, Boston, MA, Sep. 1999, vol. 3846, pp. 181–191.

[25] K. Ratakonda, I. M. Sezan, and R. J. Crinon, "Hierarchical video summarization," in *Proc. SPIE Conf. Visual Communications and Image Processing*, San Jose, CA, Jan. 1999, vol. 3653, pp. 1531–1541.

[26] A. M. Ferman and A. M. Tekalp, "Two-stage hierarchical video summary extraction to match low-level user browsing preferences," *IEEE Trans. Multimedia*, vol. 5, no. 2, pp. 244–256, Jun. 2003.

[27] D. Farm, W. Effelsberg, and P. H. N. de With, "Robust clustering-based video-summarization with integration of domain-knowledge," in *Proc. IEEE Int. Conf. Multimedia and Expo 2002 (ICME'2002)*, Lausanne, Switzerland, Aug. 26–29, 2002, pp. 89–92.
[28] Y. Rubner, L. Guibas, and C. Tomasi, "The earth mover's distance, multi-dimensional scaling, and color-based image retrieval," in *Proc. ARPA Image Understanding Workshop*, May 1997.
[29] I. Yahiaoui, B. Merialdo, and B. Huet, "Comparison of multi-episode video summarization algorithms," *EURASIP J. Appl. Signal Process.*, vol. 3, no. 1, pp. 48–55, 2003.
[30] Y. Gong and X. Liu, "Video summarization using singular value decomposition," in *Proc .IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 13–15, 2000, pp. 174–180.
[31] M. Cooper and J. Foote, "Summarizing video using non-negative similarity matrix factorization," in *Int. Workshop on Multimedia Signal Processing*, St. Thomas, U.S. Virgin Islands, December 9–11, 2002 [Online]. Available: http://citeseer.nj.nec.conV56859i.html
[32] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," *Adv. Neural Inform. Process. Syst.*, vol. 13, pp. 556–562, 2001.
[33] A. Ekin and A. M. Tekalp, "Automatic soccer video analysis and summarization," in *Proc. SPIE Conf. Storage and Retrieval for Media Databases 2003*, Santa Clara, CA, Jan. 20–24, 2003, vol. 5021, pp. 339–350.
[34] B. Li, H. Pan, and I. Sezan, "A general framework for sports video summarization with its application to soccer," in *Proc. IEEE Int. Conf. Acoustic, Speech and Signal Processing*, Hong Kong, Apr. 6–10, 2003, pp. 169–172.
[35] R. Cabasson and A. Divakaran, "Automatic extraction of soccer video highlights using a combination of motion and audio features," in *Proceedings of SPIE Conference on Storage and Retrieval for Media Databases 2003*, Santa Clara, CA, Jan. 20–24, 2003, vol. 5021, pp. 272–276.
[36] L. Agnihotri, K. V. Devera, T. McGee, and N. Dimitrova, "Summarization of video programs based on closed captions," in *Proc. SPIE Conf. Storage and Retrieval for Media Databases 2001*, San Jose, CA, Jan. 2001, vol. 4315, pp. 599–607.
[37] M. J. Pickering, L. Wong, and S. M. Rueger, "ANSES: summarization of news video," in *Proc. Int. Conf. Image and Video Retrieval*, Urbana, IL, July 24–25, 2003, vol. LNCS 2728, pp. 425–434.
[38] R. Barzilay and M. Elhadad, "Using lexical chains for text summarization," in *Proc. Intelligent Scalable Text Summarization Workshop (ISTS'97)*, Madrid, Spain, July 1997 [Online]. Available: http://citeseer.nj.nec.com/barzilay97using.html
[39] Y.-F. Ma, L. Lu, H.-J. Zhang, and M. Li, "A user attention model for video summarization," in *Proc. ACM Multimedia'02*, Juans Les Pins, France, Dec. 1–6, 2002, pp. 533–542.
[40] S.-F. Chang and H. Sundaram, "Structural and semantic analysis of video," in *Proc. IEEE Int. Conf. Multimedia and Expo (ICME'2000)*, New York, Jul. 30–Aug. 2 2000.
[41] K. S. Jones, , Knorz, Krause, and Womser-Hacker, Eds., "What might be in a summary?," in *Information Retrieval 93: Von der Modellierung zur Anwendung*. Konstanz, Germany: Univ. Konstanz, Sep. 1993, pp. 9–26 [Online]. Available: http://citeseer.nj.nec.com/jones93what.htm
[42] L. R. Bahl, S. Balakrishnan-Aiyer, J. R. Bellegarda, M. Franz, P. S. Gopalakrisnan, D. Nahamoo, M. Novak, M. Padmanabhan, M. A. Picheny, and S. Roukos, "Performance of the IBM large vocabulary continuous speech recognition system on the ARPA wall street journal task," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, Detroit, MI, May 8–12, 1995, pp. 41–44.
[43] B. T. Truong, C. Dorai, and S. Venkatesh, "New enhancements to cut, fade, and dissolve detection processes in video segmentation," in *Proc. 8th ACM Multimedia Conf.*, Los Angeles, CA, Oct. 30–Nov. 4 2000, pp. 290–301.
[44] K. S. Jones, S. Walker, and S. E. Robertson, "A probabilistic model of information retrieval: development and status," *Inform. Process. Manag.*, vol. 36, no. 6, pp. 779–840, 2000.
[45] S. S. Chen, E. M. Eide, M. J. F. Gales, R. A. Gopinath, D. Kanevsky, and P. Olsen, "Automatic transcription of broadcast news," *Speech Commun.*, vol. 37, no. 1–2, pp. 69–87, 2002.
[46] C. D. Manning and H. Schutze, *Foundations of Statistical Natural Language Processing*. Cambridge, MA: MIT Press, 1999.
[47] T. E. Dunning, "Accurate methods for the statistics of surprise and coincidence," *Comput. Linguist.*, vol. 19, no. 1, pp. 61–74, 1993 [Online]. Available: http://citeseer.nj.nec.com/dunning93accurate.html
[48] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. New York: Wiley, 1990.
[49] D. Pisinger, "An expanding-core algorithm for the exact 0-1 knapsack problem," *Eur. J. Oper. Res.*, vol. 87, pp. 175–187, 1995.
[50] R. O. Duda, P. E. Han, and D. G. Stork, *Pattern Classification*. New York: Wiley, 2001.
[51] P.-H. Luhn, "Automatic creation of literature abstracts," *IBM J.*, vol. 2, no. 2, pp. 159–165, 1958.
[52] H. Jing, R. Barzilay, K. McKeown, and M. Elhadad, "Summarization evaluation methods: experiments and analysis," in *Proc. AAAI Symp. Intelligent Summarization*, Palo Alto, CA, March 23–25, 1998 [Online]. Available: http://citeseer.nj.nec.com/jing98summarization.htmi
[53] I. Mani, D. House, G. Klein, L. Hirschman, L. Obrst, T. Firmin, M. Chrzanowski, and B. Sundheim, The TIPSTER SVMMAC Text Summarization Evaluation Oct. 1998, NIST, Tech. Rep..
[54] Kingdom of the Seahorse, PBS NOVA 1997.
[55] Mark Twain, PBS Home Video 2002.
[56] Why Dogs Smile and Chimpanzees Cry, Discovery Home Video 1999.

Cuneyt M. Taskiran (M'04) was born in Istanbul, Turkey. He received his B.S. and M.S. degrees in electrical engineering from Bogazici University, Istanbul, in 1990 and 1993, respectively, and the Ph.D degree in electrical and computer engineering and the M.A. degree in linguistics, both from Purdue University, West Lafateyette, IN.

He joined Motorola Labs, Schaumburg, IL, in 2004. His research interests include media analysis and understanding for content-based applications, media summarization, and natural language watermarking.

Zygmunt Pizlo received the Ph.D. degree in electrical and computer engineering in 1982 from the Institute of Electron Technology, Warsaw, Poland and the Ph.D. degree in psychology in 1991 from the University of Maryland, College Park.

He is a professor of psychology at Purdue University, West Lafayette, IN. His research interests include all aspects of visual perception, motor control and problem solving.

Arnon Amir (SM'05) received the D.Sc. degree in computer science from the Technion—Israel Institute of Technology, Haifa, Israel, in 1997.

He is a Research Staff Member at the IBM Almaden Research Center, San Jose, CA. His main research is in multimedia information retrieval, including speech, image and video analysis, computer vision, image and video segmentation, indexing, searching and browsing. He is mostly known for his contributions to the IBM CueVideo project. He has authored and co-authored many technical papers, holds eight U.S. patents, and served on many program committees of major conferences.

Dr. Amir is a member of the IBM team for the NIST TRECVID video retrieval benchmark since it started in 2001. He is a member of the ACM.

Dulce Ponceleon (M'03) received the M.S. and Ph.D. degrees in computer science from Stanford University, Stanford, CA.

She worked in the Advanced Technology Group at Apple Computer, Inc., where she worked on information retrieval, video and audio compression technologies for QuickTime, and computer graphics. She was a key contributor to the first software-only videoconferencing system. In 1997, she joined the IBM Almaden Research Center, San Jose, CA where she has worked on multimedia content analysis and indexing, video summarization, applications of speech recognition, storage systems, and content protection. She holds several patents and numerous publications in video and audio compression, multimedia information retrieval, numerical linear algebra and nonlinear programming.

Dr. Ponceleon contributed to the ISO MPEG-7 standardization efforts, specifically in Multimedia Description Schemes. She is currently an IBM representative in two standards bodies: Advanced Access Content System (AACS) and 4C Entity. AACS is a content protection standard for the next generation of prerecorded and recorded optimal media for consumer use with PCs and CE devices. She is the chair of the 4C Technical Group. She is in the program committee for ACM Multimedia, SPIE and several multimedia workshops.

Edward J. Delp (S'70–M'79–SM'86–F'97) was born in Cincinnati, OH. He received the B.S.E.E. (cum laude) and M.S. degrees from the University of Cincinnati, and the Ph.D. degree from Purdue University, West Lafayette, IN. In May 2002, he received an Honorary Doctor of Technology degree from the Tampere University of Technology, Tampere, Finland.

From 1980 to 1984, he was with the Department of Electrical and Computer Engineering, The University of Michigan, Ann Arbor.. Since August 1984, he has been with the School of Electrical and Computer Engineering and the Department of Biomedical Engineering at Purdue University. In 2002, he received a chaired professorship and currently is The Silicon Valley Professor of Electrical and Computer Engineering and Professor of Biomedical Engineering. His research interests include image and video compression, multimedia security, medical imaging, multimedia systems, communication and information theory.

Dr. Delp is a Fellow of SPIE, a Fellow of the Society for Imaging Science and Technology (IS&T), and a Fellow of the American Institute of Medical and Biological Engineering. In 2004, he received the Technical Achievement Award from the IEEE Signal Processing Society for his work in image and video compression and multimedia security. In 1990, he received the Honeywell Award and in 1992 the D. D. Ewing Award, both for excellence in teaching. In 2001, he received the Raymond C. Bowman Award for fostering education in imaging science from the Society for Imaging Science and Technology (IS&T). In 2004, he received the Wilfred Hesselberth Award for Teaching Excellence. In 2000, he was selected a Distinguished Lecturer of the IEEE Signal Processing Society and in 2002 he was awarded a Nokia Fellowship.