The VISION Digital Video Library Project

Susan Gauch, John Gauch and Kok Meng Pua
Information and Telecommunication Technologies Laboratory
Department of Electrical Engineering and Computer Science
University of Kansas

ABSTRACT

The goal of the VISION (Video Indexing for SearchIng Over Networks) project is to demonstrate the technology necessary for a comprehensive, on-line digital video library. We have developed real-time algorithms to create a searchable and browsable video archive. Our approach is based on the integrated application of mature image or video processing, information retrieval, and text classification technologies for efficient creation and exploration of the library materials. In order to provide access to video footage within seconds of broadcast, we have developed a new pipelined digital video processing architecture which is capable of digitizing, processing, indexing, and compressing video in real time on an inexpensive general purpose computer. These videos are automatically partitioned into short scenes using video, audio and closed-caption information. The resulting scenes are indexed based on their captions and stored in a multimedia database. A client-server-based graphical user interface was developed to enable users to remotely search this archive and view selected video segments over networks of different bandwidths. Additionally, VISION classifies the incoming videos with respect to a taxonomy of categories and will selectively send users videos which match their individual profiles. The archive can also be explored by browsing through the taxonomy.

1. INTRODUCTION

As a result of the rapid development of multimedia computing technologies and high-speed network systems, vast amounts of multimedia information are becoming available over the Internet. Organizing such a tremendous amount of data to provide intelligent access and effective use is a major topic of digital library research. One problem which limits the use of this material is the lack of effective video indexing methods. Users of video archives often rely on title and keyword information to identify videos of interest. Valuable time is then wasted manually scanning the video to locate the portions which are most relevant.

The VISION (Video Indexing for SearchIng Over Networks) digital video library prototype is being developed at the Information and Telecommunication Technologies Laboratory of the University of Kansas as a testbed for evaluating automatic and comprehensive mechanisms for library creation and content-based search, retrieval, filtering and browsing of video across networks with a wide range of bandwidths (Gauch, S., 1994). Our pilot system was populated with a collection of nature, science, and news videos from WGBH and CNN (Gauch, S., 1995, 1997).

These videos were automatically partitioned into short segments based on their content, and stored in a multimedia database. A client-server based graphical user interface was developed to enable users to remotely search this library and view selected video segments over networks of different bandwidths. Recent advances enable us to classify video scenes into an appropriate category in a pre-defined taxonomy, which supports content-based browsing.

The value of any library is related to both the volume and timeliness of the information it contains. Digital video libraries are no exception. Ideally, we would like to have video footage added to the library within seconds of broadcast. In order to achieve this goal, it is necessary to perform video digitization, segmentation and compression in real time. To address this problem, we have developed a new pipelined digital video processing architecture which is capable of digitizing, processing, indexing, and compressing video in real time on an inexpensive general purpose computer. VISION has also been extended to operate as an information filtering system, classifying video and sending it to users whose profiles contain the matching categories (Gauch, J., 1998). Preliminary work has been done on keyframe extraction and feature analysis to support archive browsing image based querying.

2. RELATED WORK

Unlike "video-on-demand" services, digital video libraries integrate image and video processing and understanding, speech recognition, distributed data systems, networks, and human-computer interactions in a comprehensive system. A key component of this difference is the use of content-based indexing and retrieval algorithms to enable users to interact with the video library rather than simply playing back entire movies or broadcasts. As a consequence, there has been considerable activity developing improved tools for video processing and content analysis. Systems which share features and goals with the VISION system are described below.

Several approaches have been proposed to decompose raw video into *shots* (a continuous roll of a camera) and *scenes* (collections of shots which occur in a single location or are temporally unified). It is important to note that the above definitions follow usage defined by (Hampapur 1994) whereas some authors use the word *scene* to refer to a sequence of video representing continuous action and *story* to refer to a sequence of scenes. The problem of identifying *cuts* (sharp transitions between shots) has been typically approached from a bottom up perspective, looking for rapid changes in color histogram or image intensity (Arman, 1993; Nagasaka, 1992; Zhang, 1993). Model-based algorithms have also been developed to successfully detect fades, dissolves, and page translate edits (Hampapur, 1994). Once shots have been identified, keys frames which characterize the shot can be selected by considering the motion of objects within the shot. Here, we can either select frames which are as still as possible (Wolf, 1996) or identify the background and moving objects explicitly and select an image which focuses on one or the other (Sawheney, 1996).

Although shot boundaries can now be accurately detected, in general, detecting scene boundaries accurately is still an open problem. The Princeton Deployable Video Library (PDVL) (Wolf, 1995; Yeo & Yeung, 1997) is identifies key frames in each shot and uses image-based clustering to construct a scene transition graph to visually present the relationships among shots. By browsing through a collection of graphs, users can locate scenes of interest (e.g., two person interviews). The scene transition graph can then be used to navigate through the video. The Algebraic Video System (Weiss, 1995) uses an alternative technique where shots are organized in a hierarchical structure which allows nested stratification (subtrees may refer to overlapping portions of the raw video). This system uses the VuSystem (Lindblad, 1994) for recording and processing video but hierarchy construction is currently performed manually. A model-based approach has been proposed to parse video by an *a priori* model of the video structure (Zhang, 1995; Zhang, 1997). Such a model represents a strong spatial order within the individual frames of shots and/or strong temporal order across a sequence of shots. For many tasks, however, it will be difficult or impossible to define models for the video.

There are three basic approaches to automatically identifying the content of a video segment: image understanding, speech recognition, and caption processing. Although the human visual system is very effective, research in computer vision over the past 20 years has had success in only limited domains (Haralick, 1992). For this reason, many approaches for image-based content identification have focused on feature-based classification schemes. For example, images can be indexed using color histograms (Swain, 1991) or combinations of shape and color features (Smoliar, 1994). The QBIC (Query By Image Content) project (Faloutsos, 1994) investigated methods to query large on-line image databases using the image contents, such as color, texture, shape, and size. Although feature-based classification is quite fast, one drawback is that very different objects may have the same features (e.g., a red car and a red apple). Similarity-based image retrieval can be also accomplished using Hidden Markov Models (HMM) which have been trained with representative images (Yu, 1995), multiresolution wavelet decompositions (Jacobs, 1995), and combinations of features such as texture, shape and appearance (Picard, 1994; Pentland, 1996). Although these systems may have excellent success within a restricted domain of images (e.g., textures and faces), the computational expense may limit their use for identifying video content.

Given the difficulty of image-based content analysis, processing the audio track and closed caption information is an attractive alternative. In one study (Turner, 1997), the words in closed captions were compared to index terms manually assigned by professionals. There was a strong overlap in the two sets (over 80% agreement), suggesting that the closed captions are a reasonably accurate indexing resource. However, alignment of the captions with shots must be done carefully. In the sample of Nova evaluated in the study, 33% of the captions appear partially or wholly in adjacent shot(s). This problem is likely to be even larger in video which has been captioned in real-time where delays of three to five seconds are common.

The Informedia Digital Video Library project uses video, audio and caption information to create a large, on-line library (Christel, 1994; Christel, 1995; Wactlar, 1996). Informedia's News-On-Demand system (Hauptmann, 1997) uses entirely automatic segmentation and indexing to provide efficient access to news videos. News-On-Demand uses the video, audio and closed captions for segmentation and also includes a large-vocabulary, speaker-independent, continuous speech recognizer. Speech recognition allows them to align the closed captions with the video and to provide words for indexing where closed captions are unavailable. Their archive is built using MPEG-I for the video, with one Pentium PC used for the digitization and a supplementary platform for the SPHINX-II speech recognition system.

The VISION system (Gauch S., 1994, 1995, 1997) shares many of the goals of the News-On-Demand project, but we further constrain the problem by requiring a system which is capable of making all news stories available within seconds of their broadcast. In addition, we have designed the system to run continuously on a dual-processor Pentium PC. Thus, one can monitor multiple broadcast channels cost-effectively by devoting one commodity computer per channel, all of which feed the indexing information to a central database. Because of these constraints, we perform limited processing of the audio track during automated scene segmentation and content analysis. In particular, audio information is used only to combine shots. Closed captions are required in almost all television broadcasts, and they are a valuable source of information for video segmentation. The text associated with each scene is also used as input to our full-text retrieval engine to search the video library for material of interest. The text is also used to classify the scene into a category in our taxonomy which allows it to be browsed or sent to users who have indicated an interest in that category. Our major contribution is an exploration of what can be achieved in entirely automatic archive construction running real-time on commodity hardware.

3. ARCHITECTURE OF VISION

VISION is constructed in a client-server architecture in which each subsystem provides a major function to other subsystems through network protocols. It is logically divided into the following subsystems:

- *Video Processing System (VPS)* which captures video/audio and closed-captions and produces compressed and segmented video scenes, keyframes, keyframe features and keywords for each scene.

- *Client* the graphical user interface of the system, which transmits queries and browsing requests to the VISION server and plays back video/audio/captions.

- *Server* which interacts with the Client to receive browsing and searching requests and display keyframes, browsing hierarchies, and video streams.
- *Video Streamer* which streams a selected video scene to the client over a network connection.
- *Search Engine* which indexes the video scenes by their associated keywords and provides full-text search of the video archive.
- *Archive Browser* which allows the user to browse the archive via the taxonomy of categories or the feature-based clusters.

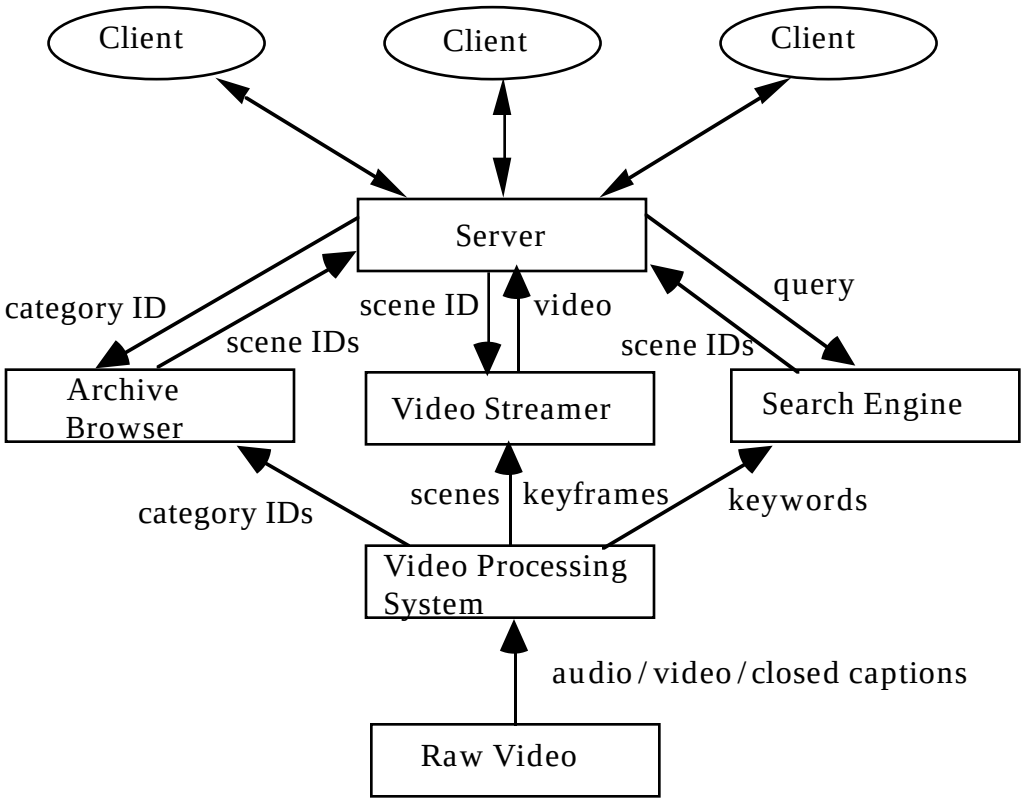

Figure 1. The architecture of the VISION system.

The architecture of VISION is summarized in Figure 1. Although we originally developed our own client, server and video streamer, the VISION system now uses a World Wide Web server and an Internet browser for the Client and Server, and the RealMedia Server and Client for video streaming. The Video Processing System, Archive Browser and Search Engine are all written in C/C++ and run on a dual-processor Pentium II. The Search Engine is an implementation of the vector space model (Salton, 1983) and the Archive Browser is a cgi program which presents a browsable hierarchy of concepts (the taxonomy) or a clustered set of keyframes to

the user. The input to all the aforementioned subsystems is produced by the Video Processing System, which we will discuss in more detail in the following sections.

4. VIDEO PROCESSING SYSTEM (VPS)

The Video Processing System (VPS) can continuously capture, segment, compress, classify, extract keyframes (and their features) and store and index video clips from a live broadcast feed in real-time. To achieve this, we extract video, audio and closed caption features as the video is digitized and use this information to segment the video into meaningful scenes in real time. The digitized audio and video frames associated with each scene are then compressed and placed in the multimedia archive within seconds of their broadcast. Given the real time constraint, and the computational platform we were targeting for this system, special care was taken to balance the computational load of feature extraction, segmentation, compression and content classification. The remainder of this section describes our software design and implementation in detail.

Our new video processing system consists of five main components: (1) the video capture and feature extraction (VCFE) module, (2) the video segmentation and compression (VSC) module, (3) the keyframe extraction module (KFE), (4) the video classification module (VC), and (5) the keyframe clustering module (KC). These modules execute in separate threads on a dual processor PC running Windows NT and communicate through shared memory data buffers.

4.1 The Video Capture and Feature Extraction (VCFE) module

The VCFE module digitizes the audio and video frames and closed captions. In order to ensure that no audio or video frames are lost (which would cause the audio and video tracks to become mis-aligned over time), we do not perform any disk I/O, but rather we store the results of audio/video feature extraction in a large circular buffer pool in shared memory. After experimenting with a variety of image features (Bouix, 1998), we chose to extract invariant moments of the color distribution and the image. These seemed to provide the best balance between efficient extraction and acceptable image similarity metrics. The captions are read two characters per frame, and broken in to tokens (words) which are reduced to their word stems and stop words are removed. The remaining keywords are saved in a shared memory buffer together with timing and word frequency data.

4.2 The Video Segmentation and Compression (VSC) module

The VSC module does real time video segmentation and controls the audio/video compression. Before segmentation begins, we buffer several seconds of audio/video and closed captions. We then apply the shot and scene detection algorithms described in Section 5. The data provided by the VCFE module is used to detect shot boundaries and perform audio- based shot merging. Text analysis functions are invoked at the remaining shot boundaries to determine if caption-based merging is needed. This generates starting and ending frame numbers of scenes which in turn

are used to control the software video compression process. Simultaneously, the video features are used to identify keyframes. While most features are discarded once segmentation is completed, the features extracted from the keyframes are permanently stored for use by the clustering algorithm.

We perform software-based audio/video compression using a SDK produced by Real Networks which outputs digital video in RealMedia format. This obviates the need for special purpose MPEG/JPEG video compression hardware, while also providing low bit rate digital video compression suitable for Internet broadcast. Each video scene is stored in a separate RealMedia file on disk. When the end of a scene is detected, the current file is closed and copied to the video server, and a new output file is created. A separate video library client which uses a RealMedia decoder can then be used to retrieve and play back video clips as soon as they become available. Because compression requires file I/O and a variable amount of computational time per frame, it was important to perform this operation in a separate thread from the VCFE operations which are time critical (frames are lost if audio/video callbacks take too much time). On a dual processor system, this also enables our system to distribute the work load between the two processors. The segmentation process is described in detail in Section 5.

4.3 The Keyframe Extraction (KFE) module

The KFE module uses the features extracted by the VCFE module to identify a keyframe for each shot detected by the VSC module. Due to efficiency constraints, we concentrated on identifying a single representative keyframe for each shot rather than attempting to combine frames to produce a shot summary. Most keyframe extraction techniques rely on motion estimation. Since motion information is stored as part of the MPEG format, this information is readily available. However, we use a RealMedia or AVI format for our video and must calculate our motion estimates, which can be computationally expensive. To address this, we defined a new *stillness* criteria based on a local energy function which computes the sum of pixel intensity differences over a sequence of n frames around a specified frame t. For each shot, we first identify the frame t_0 which minimizes the energy function for the shot. Then, to identify the most representative frame within a window of 5-10 frames of t_0, we select the frame whose brightness is closest to the average brightness for frames in the window. We evaluated our keyframe selection algorithm by having users view 80 minutes of video which contained roughly 1500 shots. For each shot, they were asked to rate quality of the chosen keyframe. They rated the keyframes as Very Satisfactory (60%), Satisfactory (35%), or Irrelevant (5%), indicating a high-level of satisfaction with the keyframe selections (Bouix, 1998).

4.4 The Video Classification (VC) module

Our video categorization and filtering system has the following components:

- a taxonomy of categories which will be used for video classification
- training data for each category
- a classification algorithm
- the user profile (a selection of categories from the above taxonomy)

- the text associated with the video being classified

Any hand-built taxonomy of categories which matches the domain of the expected video clips would suffice. Since we are currently working with news videos, we have chosen a taxonomy designed to classify news stories on the Web as the basis for our classification system. This taxonomy contains approximately 2,000 categories arranged in a hierarchy of height three. For training data, we spidered the site over several days to download not just the category names, but also 2 - 4 Web pages hand-attached to the leaf nodes. These Web pages form the basis of our training data. Note that having accumulated a taxonomy and sufficient training data, we are unaffected by future changes to the taxonomy structure (which has, indeed, undergone a significant rearrangement). However, if new stories occur which use previously unseen words as major clues about the correct category, we will need to update the documents used as training data for the category. This could be done by adding the closed captions from correctly classified clips to the training data for the appropriate category. We then built a Java-applet which displays the top three levels of the taxonomy and allows users to select one or more leaf categories as being of interest. This list of selected categories forms their user profile.

The final component of the filtering system associates an incoming video clip with the appropriate category(s) so that it can be automatically sent to interested users. Our classification system makes use of the software built for searching the archive. Traditional text-based searching takes in a query and returns the top matches from an indexed collection of documents. Similarly, our classifications system takes in the closed-caption information for a new video clip and returns the top matching categories from an indexed collection of categories. To accomplish this, the closed-caption information is initially processed to identify the highest weighted N words (usually 5 - 10 words per clip), where the weight for term w in video clip v is calculated as follows:

$$Wt(w, v) = \sum tf_{wv} * idf_w$$

where

tf_{wv} is the frequency of term w in clip v

$idf_w = \log_2 (freq_{max} / freq_w)$

$freq_{max}$ = frequency of most frequent term in sample text collection[1]

$freq_w$ = frequency of term w in sample text collection[1]

These highly weighted words and their weights are then used as a query against the index of categories. The categories themselves are indexed by treating all of the Web pages associated with each category as a single, large document. Hauptmann and Lee (1998) also use this to classify broadcast news stories. We are currently doing extensive experimentation of our classification approach, comparing it to Bayesian approaches and studying the effect of varying amounts of training data on the quality of the results. The user can use their Web browser to explore the taxonomy

[1]The word frequency statistics from 3 years of the Wall Street Journal are used.

and see, for each category, the keyframes of the associated video clips. Figure 2 shows the architecture of the VC module.

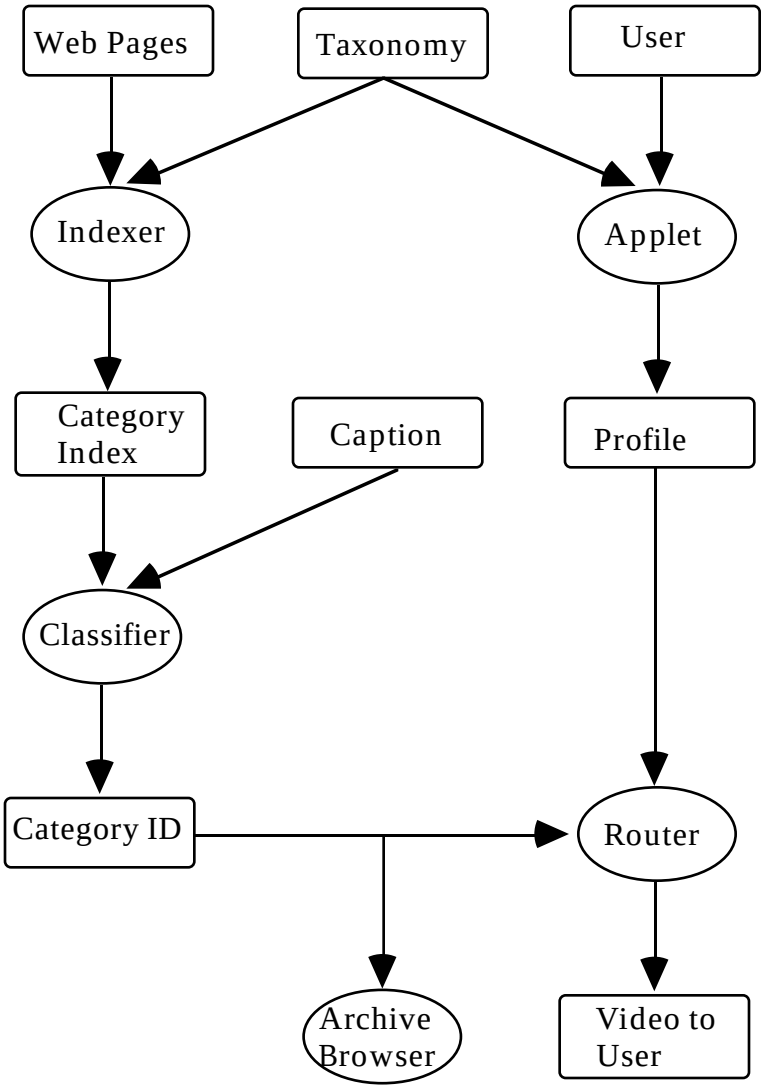

Figure 2. Data flow diagram for video scene classification.

4.5 The Keyframe Clustering (KC) module

The keyframes for each shot are clustered based on the features extracted by the VCE module (invariant moments of color distribution and image). The results of the clustering are presented via a Web page. Current work is concentrating on extending the capabilities of this module to include keyword based clustering, identifying a single representative keyframe per scene, and allowing users to control the clustering process by setting the relative importance of the various features.

5. VIDEO SEGMENTATION ALGORITHM

The goals of video segmentation are: (1) to locate the start and end of each camera shot, and (2) to combine camera shots based on content to obtain the start and end points of each scene.

5.1 Shot Detection

The detection of shot transitions can be trivial or complex depending on the video content being combined and the type of transition used. For example, when video from two very different sources are spliced together with zero frames of transition it is easy to detect the scene change. On the other hand, if two very similar shots are combined with a gradual cross fade, the visual changes may be much smaller than we might expect in a video with moderate object motion. Thus, it is very likely that any automated image-based shot detection algorithm will miss some fraction of the shot boundaries. Fortunately, this does not impact the quality of the scene detection greatly because shot transitions which are this gradual are often chosen by producers when the two shots are actually related and should remain in the same scene. Shot detection in the VISION system is performed by combining three image cues: (1) the average brightness of each video frame, (2) the change in pixel values from frame to frame, and (3) the change in color distribution from video frame to frame. These three quantities are compared to thresholds to identify potential shot boundaries. The selection of the thresholds is challenging because no set of thresholds will be effective for all sources of production video. We have addressed this problem in two ways. First, we use *a priori* knowledge of each video producer (e.g., WGBH, CNN, CNBC) to select initial values for these thresholds based on which video source is being processed. Then, we gather statistical information about during video processing to update the thresholds dynamically every few minutes.

Our process of shot detection is illustrated in Figure 3. In this data flow diagram, raw NTSC video is input. The Boolean output function shot(t) is true if a shot boundary has been detected at frame t, and false otherwise.

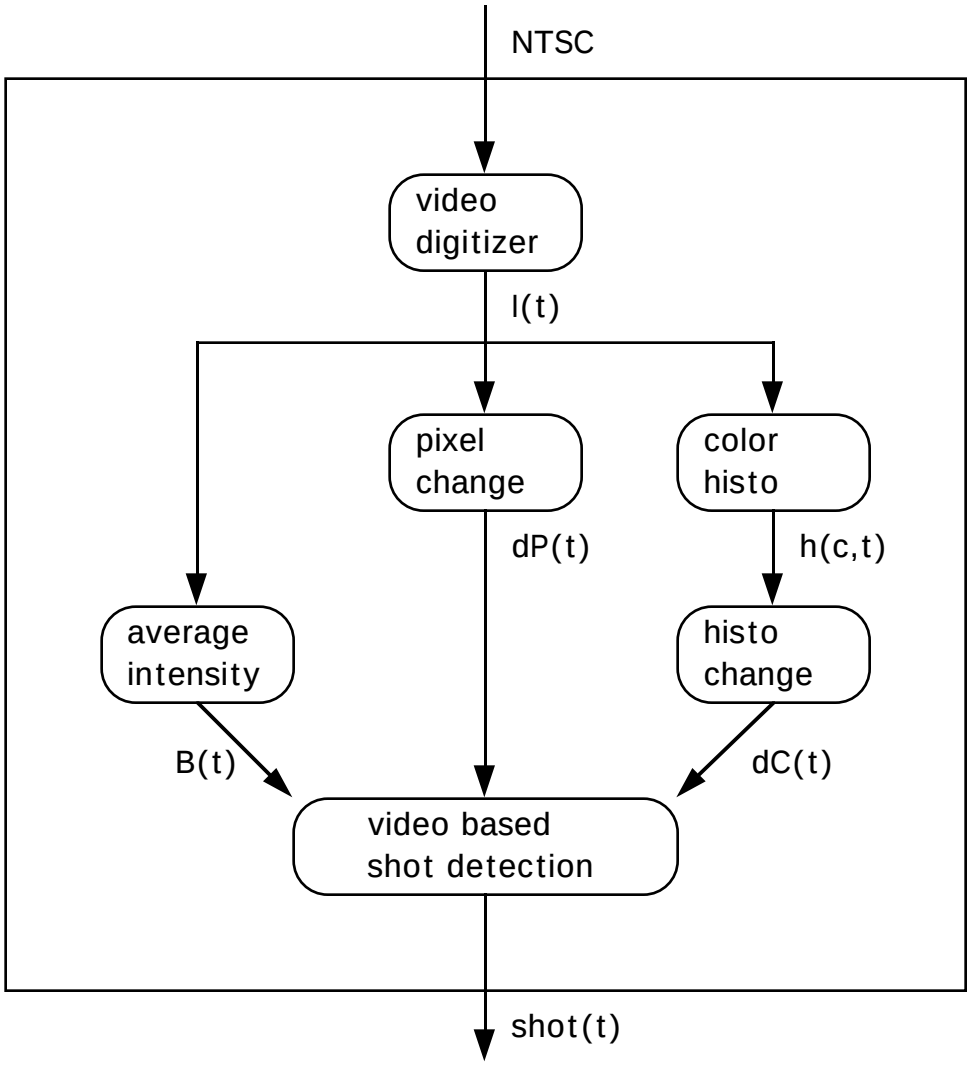

Figure 3. Data flow diagram for shot detection.

5.2 Merging Shots to Obtain Scenes

Since many video producers use motion and shot transitions to attract and retain viewer interest, it is common to have numerous shots per scene. Merging related shots back together to identify scenes is very important to avoid excessive fragmentation of information in the library. There are two sources of information which can be exploited for this purpose: audio cues, and closed caption cues. Given our requirement for real time video segmentation, we focus on low level audio properties. If someone is talking while there is a shot transition, it is an indication that the two shots are related and should be merged. To determine if this situation occurs, we perform endpoints detection on the audio signal to identify the start and end of each utterance. We merge adjacent shots together if the audio energy is above a thresholds. Again, the threshold value is chosen using *a priori* information about the video source and updated dynamically using audio statistics.

The second source of information is actually the most important for the VISION system. One problem with some video sources is that the closed captions are entered as the show is broadcast. This introduces a 2-3 second time delay between when words are spoken and when the transcription appears. Hence, it is necessary to estimate the time delay and realign the closed captions with the audio and video. Incorporating speech recognition could be used to align the captions with the spoken words and thus get a more accurate estimate of the delay (Hauptmann, 1997), but this would require a second computer per broadcast channel and may not operate in real-time.

Once caption alignment has been performed, it is possible to consider the topics being discussed on either side of a shot boundary. If they are similar, although there is a change of shot, there is no change of scene and the shots should be merged. For each shot boundary, we consider the words used within a window of a given number of frames on either side of the boundary (adjusted by the delay factor). We tokenize the closed captions to identify words, then use the Porter stemmer (Frakes, 1992) to remove prefixes and suffixes, and finally delete the most frequent English words (*stopwords*) from the captions in each window. The remaining terms are then weighted using the standard *tf*idf* formula. We then use the cosine similarity measure (Salton, 1983) to calculate the vocabulary overlap between the windows to measure content similarity. We merge adjacent shots when the similarity is above a threshold. To perform this text analysis in real time, we make extensive use of special purpose hash tables for fast lookup into the stopwords list and our lexicon of 40,000 words and their frequencies in a news related corpus (three years of the Wall Street Journal).

Finally, there is an additional closed caption cue which is helpful in certain situations. A change in speaker is often marked by the symbol ">>" in the closed captions. Similarly, a change in topic may be indicated by the symbol ">>>". Thus, we apply the heuristic that if there is a change in topic symbol which is close to the shot transition, then we override any audio cues or word similarity cues and prevent potential shot mergers. As the caption text is processed, we calculate the distance in frames to the nearest ">>>" symbol. We do not merge adjacent shots if distance to the ">>>" symbol is above a threshold.

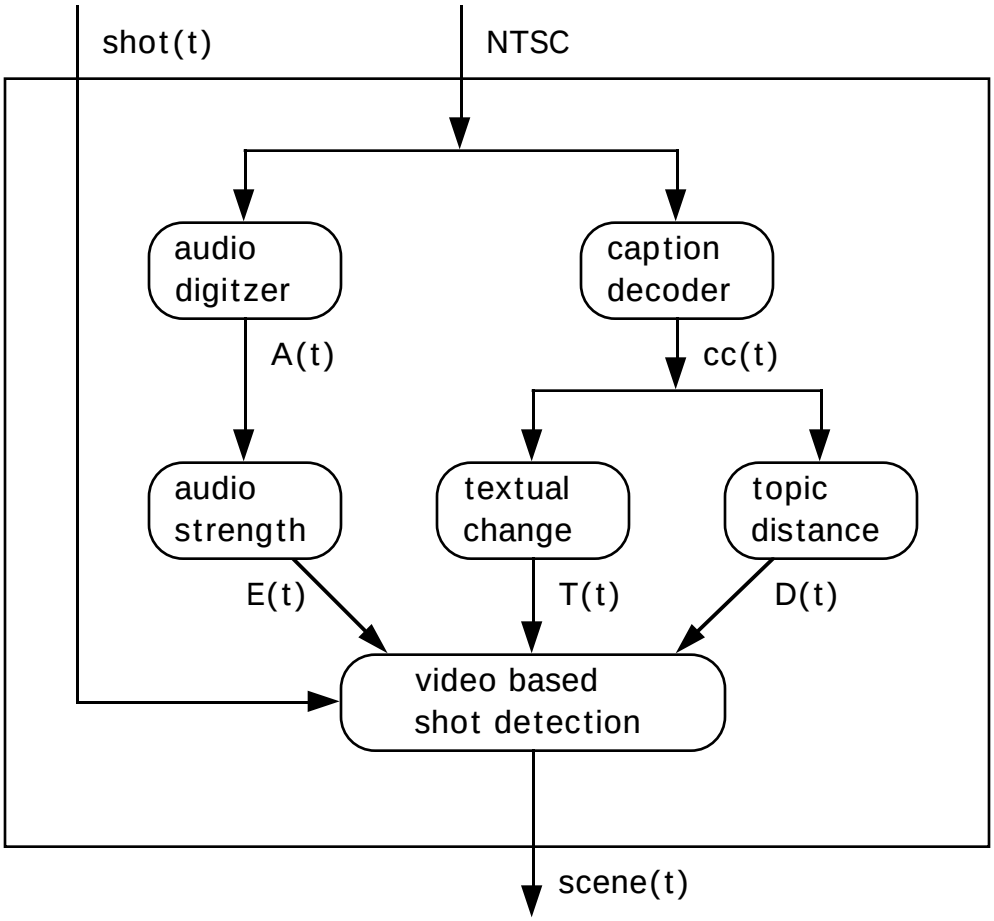

Figure 4. Data flow diagram for scene detection.

The VISION scene merger process is illustrated in Figure 4. The raw NTSC signal and the shot(t) function are inputs. The Boolean output function scene(t) is true if a scene boundary has been detected at frame t, and false otherwise.

5.3 Segmentation Examples

The identification of shot boundaries using video information is relatively straightforward if there is a single frame transition between one shot and the next. Gradual shot transitions and wipes are more challenging to detect. This is where the combination of pixel differences and color histogram differences assist in distinguishing shot boundaries from locations where rapid motion in the video sequence occurs. Figures 3-5 illustrate sequences of four frames from CNN Headline News where our system has correctly identified shot boundaries.

INSERT FIGURE 5 HERE

Figure 5. Example 1 of shot boundary detection. There is a topic change symbol, ">>>" nearby, so this becomes a scene boundary.

INSERT FIGURE 6 HERE

Figure 6. Example 2 of shot boundary detection. The closed captions are similar, so no scene boundary is detected and the shots are merged.

The use of audio and closed caption information to detect scene boundaries is also demonstrated in these examples. The audio energy at the shot boundary in Figure 5 were low, indicating a possible scene boundary, but the words in the closed captions were similar so the two shots shown in Figure 5 were merged.

5.4 Segmentation Evaluation

To evaluate the accuracy of our video segmentation results, we conducted a number of experiments processing videos which were hand segmented to identify the "true" scene locations. Overall, the accuracy of our results are quite good, although there is a tendency to over-segment the input video, breaking news stories and commercials into several parts. To better understand the interactions of the video, audio, and closed caption features when partitioning the video into scenes, we conducted three experiments: (1) video only segmentation, (2) video segmentation with audio merging, and (3) video segmentation with audio and caption based merging. Although many values for the various thresholds were evaluated, for brevity, only the results with the best threshold settings are shown here.

From the results shown in Table 1, we can see that most scene boundaries are detected by the video segmentation (over 94%). However, only 11% of the shot boundaries detected actually correspond to scene boundaries. In other words, the average scene is chopped into ten pieces, clearly demonstrating the need for other techniques merge some shots together. By adding audio information, the percentage of boundaries produces that correctly correspond to scene changes has more than doubled, from roughly 11% to 24.5%. At the same time, the number of scene boundaries found has decreased approximately 13.5% from 94.5% to 81.5%.

Segmentation Technique	Recall	Precision
Video Only	0.9425	0.1087
Video and Audio	0.8150	0.2450
Video/Audio/Captions	0.9200	0.2313

Table 1 Recall and precision values for various scene segmentation techniques evaluated on two hours of CNN Headline News. $P_{threshold} = 70$, $C_{threshold}=70$, automatic audio thresholding, $T_{threshold}=7$, CC-delay=100 and CC-windowsize=4,000.

When we add closed caption information during segmentation, we achieve precision percentages comparable to those resulting from video and audio

segmentation (23% versus 24.5%) while greatly improving recall (92% versus 81.5%). In other words, with closed caption information we remove almost as many false boundaries and far fewer true ones. In fact, very few true boundaries were removed since we started with 94% recall after the video phase which decreased only 2% while precision more than doubled. On average, 92% of all scene boundaries are detected and the average scene is split into four pieces rather than ten pieces which was the case before the merging process began. Although there is obviously work remaining to further increase precision, the results are quite encouraging.

6. CONCLUSIONS

Our real time video segmentation and classification system is now fully operational. It can continuously capture, segment, compress, classify, index and store video clips from a live broadcast feed in real-time. In addition to the new pipeline architecture, we also presented our segmentation algorithm which fuses three sources of information: video, audio and closed-captions. This provides much higher scene detection accuracy than that realized with just video alone or video plus audio. Finally, we describe an information filtering application based upon VISION which matches incoming video to user profiles based upon a taxonomy of categories. The classification results are also used to provide taxonomy-based browsing of the video archive. After segmentation, we locate keyframes within each of the shots in our video library by examining the same video information used for segmentation. We cluster the keyframes to provide image-based browsing of video archives. Since all processing is completely automatic, multiple installations of the VISION system can be used to monitor multiple video feeds simultaneously with little or no burden on the archive staff. It is currently in around-the-clock commercial operation, indexing CSPAN and CSPAN-2 for FASTV (www.fastv.com).

ACKNOWLEDGMENTS

This work was supported in part by the University of Kansas Research Development Fund, the Information and Telecommunication Technologies Laboratory (ITTC), and the National Science Foundation Award CDA-9401021. Video sequences from CNN Headline News are courtesy of Turner Broadcasting.

REFERENCES

Arman, F., Hsu, A., et al, (1993). Image Processing on Compressed Data for Large Video Databases, *ACM Multimedia '93*, California, USA, 267-272.

Bouix, S., (1998). VISION: Segmentation, Indexing and Retrieval of Digital Videos, *Master's Thesis*, EECS Department, University of Kansas.

Christel, M., et al, (1994). Informedia Digital Video Library, *ACM Multimedia '94*,, 480-481.

Christel, M., et al, (1995). Informedia Digital Video Library, *Communications of ACM*, Vol. 38, No. 4, 57-58.

Faloutsos, C., et al., (1994). Efficient and Effective Querying by Image Content, *Journal of Intelligent Information Systems*, Vol. 3, 231-262.

Frakes, W. B., Baeza-Yates, R. (1992) in *Information Retrieval, Data Structures & Algorithms* (Verde, K., Goodwin, B., Doench, G., and Papanikolaou, S. eds), Prentice-Hall International (UK) Limited, London.

Gauch, J.M., Gauch, S., Bouis, S., Zhu, X., (1998). Real Time Video Scene Detection and Classification, *Information Processing & Management*. (to appear)

Gauch, S., et al, (1994). The Digital Video Library System: Vision and Design, *Digital Libraries '94* , College Station, Texas, 47-52.

Gauch, S., Gauch, J., Pua, K.M., (1996). VISION: A Digital Video Library, *ACM Digital Libraries '96*, Bethesda, MD, 19-27.

Gauch, S., Li, W., Gauch, J., (1997). The VISION Digital Video Library System, *Information Processing & Management*, **33**(4), 413-426.

Hampapur, A., Jain, R., Weymouth, T., (1994). Digital Video Segmentation, *ACM Multimedia '94*, San Francisco, 357-364.

Haralick, R.M., and Shapiro, L.G., (1992). *Computer and Robot Vision*, Addison Wesley.

Hauptmann, A.G. and Lee, D. (1998). Topic Labeling of Broadcast News Stories in the Informedia Digital Video Library", *ACM Digital Libraries '98*, Pittsburgh, PA, 287-288.

Hauptmann, A.G. and Witbrock, M.J. (1997). Informedia: News-on-Demand Multimedia Information Acquisition and Retrieval, in *Intelligent Multimedia Information Retrieval*, Mark T. Maybury (ed.), MIT Press, 215-239.

Jacobs, C.E., Finkelstein, A., Salesin, D.H., (1995). Fast Multiresolution Image Querying, ACM Computer Graphics (SIGGRAPH '95), 277-286.

Lindblad, C.J., et al, (1994). The VuSystem: A Programming System for Visual Processing for Digital Video, *ACM Multimedia '94*, San Francisco, 307-314.

Pentland, A., Picard, R.W., Sclaroff, S., (1996). Photobook: Content-Based Manipulation of Image Databases, *International Journal of Computer Vision*, 18(3), 233-254.

Picard, R.W., Liu, F., (1994). A new Wold ordering for image similarity, *Proc. ICASSP*, Adelaide, Australia.

Nagasaka, A., Tanaka, T., (1992). Automatic Video Indexing and Full-Video Search for Object Appearances, *Visual Database Systems, II*, E. Knuth and L.M. Wegner, Editors, North-Holland, 119-133.

Salton, G., & McGill, M. (1983). *Introduction to modern information retrieval.* New York: McGraw-Hill.

Sawhney, H.S., and Ayer, S., (1996). Compact Representations of Videos Through Dominant and Multiple Motion Estimation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18, No. 8, 814-830.

Swain, M., and Ballard, D., (1991). Color Indexing, *International Journal of Computer Vision*, 7(1), 11-32.

Smoliar, S., and Zhang, H., (1994). Content-based video indexing and retrieval, *IEEE Multimedia Magazine*, 1(2), 62-72.

Turner, J.M., (1997). Deriving Shot-Level Indexing from Audio Description Texts, *Association of Moving Image Archivists Annual Conference (AMIA '97)*, November 17-22, Bethesda, MD, http://esi25.ESI.UMontreal.CA:80/~turner/english/texts/amia97.html, visited: 9/16/98.

Wactlar, H.D., et al, (1995). Intelligent Access to Digital Video: Informedia Project, *IEEE Computer*, Vol. 29, No. 5, 46-52.

Weiss, R., Duda, A., Gifford, D.K., (1995). Composition and Search with a Video Algebra, IEEE Multimedia, 12-25.

Wolf, W., Liu, B., Wolf, W., (1995). A Digital Video Library for Classroom Use, *Proceedings of the International Symposium on Digital Libraries*.

Wolf, W., (1996). Key Frame Selection by Motion Analysis, *Proc. ICASSP*.

Yu, H.H., and Wolf, W., (1995). Scenic Classification Methods for Image and Video Databases, *Digital Image Storage and Archiving Systems*, SPIE 2606, 363-371.

Zhang, H.J., et al, (1993). Automatic Partitioning of Video, *Multimedia Systems*, Vol. 1, 10-28.

Zhang, H.J., et al, (1995). Automatic Parsing and Indexing of News Video, *Multimedia Systems*, Springer-Verlag, No. 2, 256-266.

Zhang, H.J., Low, C.Y., Smoliar, S.W., and Wu, J.H. (1997). Video Parsing, Retrieval and Browsing: An Integrated and Content-Based Solution, in *Intelligent Multimedia Information Retrieval*, Mark T. Maybury (ed.), MIT Press, 139-158.