A TOOLSET FOR BROADCAST AUTOMATION FOR THE C-SPAN NETWORKS

Cuneyt M. Taskiran†, *Anthony Martone*†, *Robert X. Browning*‡, *and Edward J. Delp*†

†Video and Image Processing Laboratory
School of Electrical and Computer Engineering
Purdue University
West Lafayette, Indiana, U.S.A.

‡Department of Political Science
Purdue University
West Lafayette, Indiana, U.S.A.

ABSTRACT

In this paper we describe systems that automate two important processes for broadcast video, specifically C-SPAN programs. The first system that we describe automates the insertion of on-screen graphics that give information about video content. The second one automatically generates closed-caption text by aligning program transcripts and automatic speech recognition output. We have used these systems in real-world settings and present results on a large set of video sequences.

1. INTRODUCTION

In this paper we describe two systems that we have developed for video indexing and closed-caption generation to be used by C-SPAN. C-SPAN is a private, non-profit company, created in 1979 by the cable television industry as a public service, to provide public access to the political process in the U.S.A. The C-SPAN Video Archives was established to record, index, and archive all C-SPAN programming. As of January 2002 the Archives contained 167,267 hours of C-SPAN programs; every program that aired since 1987 is contained in the Archives database immediately accessible through the database and electronic archival systems developed and maintained by the Archives.

The layout of the paper is as follows: In Section 2 we describe a system to identify unique speakers in a given program. This system extracts features from each shot, which are then used to cluster shots that contain the same person. Section 3 introduces our automated closed-caption generation system, which produces off-line captions using program transcripts and automatic speech recognition. Finally, we illustrate the performance of these systems on a large collection of video data.

2. THE AUTOMATIC GRAPHICS INSERTION SYSTEM

Screen graphics in news broadcasts and documentaries provide information on *where*, *when*, and *who* of the video content. An example of a video frame with an on-screen graphics section providing the name and title of the speaker in the shot is shown in Figure 1. C-SPAN programming exclusively contains news programs, speeches, and meeting proceedings; therefore the graphics insertion requirements are larger compared to typical television programs. Currently on-screen graphics for C-SPAN programs

This work was supported by a grant from the C-SPAN Archives. Address all correspondence to E. J. Delp, ace@ecn.purdue.edu or telephone: +1 765 494 1740.

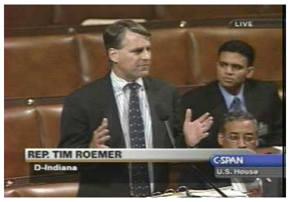


Fig. 1. A C-SPAN frame with on-screen graphics identifying the speaker.

are manually inserted. For programs in which many speakers appear, quickly identifying speakers appearing in each shot becomes a problem. Furthermore, deciding which shots to insert graphics while taking into account factors like shot length, places a large burden on the caption insertion operators.

We have developed a system that automates the above graphics insertion procedure to a large extent. We use a combination of visual and audio features to cluster shots belonging to same persons together. We assume no prior knowledge about the number of persons appearing in sequences. The components of the system are illustrated in Figure 2.

We first segment a given video to be processed into its constituent shots. For this task, only visual features, in the form of color histograms and pixel variances of frames are used. Our shot boundary detection technique is described in detail in [1]. After the video sequence is divided into shots we extract three features from each shot in the sequence.

- RGB color histogram of the middle frame of the shot, whic is selected as the keyframe.
- Average RGB color histogram of face regions detected during the shot. We use the procedure described in [2] to detect the faces.
- Acoustic vectors for the audio segment of each shot consisting of the 20 melfrequency cepstral coefficients obtained every 10 ms using a 20 ms Hamming window.

After these features are extracted from each shot, we calculate pairwise shot distances between all the shots in the sequence. We then use agglomerative clustering to group shots containing the same person together. Finally, the clusters obtained are presented to an operator who uses a graphical user interface to label each cluster with the name of the person appearing in the cluster and corrects errors in clustering, if any. Further details of our unique person detection system are given in [3]. Given the labels for shots, the

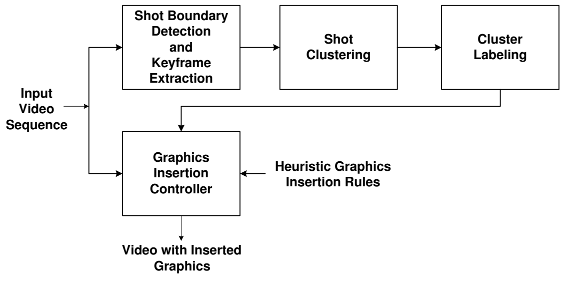

Shot Boundary Detection and Keyframe Extraction
Shot Clustering
Cluster Labeling
Input Video Sequence
Graphics Insertion Controller
Heuristic Graphics Insertion Rules
Video with Inserted Graphics

Fig. 2. Block diagram of the proposed automated program graphics generator.

system decides, for each shot, if a graphic needs to be inserted and what graphic to insert using heuristic graphics insertion rules.

3. AUTOMATED GENERATION OF CLOSED CAPTIONS

In the U.S. text information is transmitted in most broadcast video signals (both analog and digital) that corresponds to the audio information in the program. This information, known as *closed captioning*, is different from teletex information available in Europe in that the closed captioning text is displayed in real-time and approximately synchronized to the audio channel [4]. Closed captions are very similar to subtitles used in foreign films. Currently the Federal Communications Commission (FCC) in the US requires that 900 programming hours per channel per quarter be closed captioned for every television station; by 2006 all television programs, with few exceptions, must be captioned [5]. This places a large cost on television networks, which has spurred interest on fast and cost-effective methods for producing captions. Current closed-captioning technology consists of two approaches: online and off-line. While online closed-captions are generated in real time by an operator watching the program, Off-line captions can take several hours to generate. The accuracy of off-line captions is higher then online captions, but they are also more expensive to produce.

We will refer to three types of texts for a given TV program: *program transcripts*, which are accurately generated transcripts of the program prepared by a human transcriber that do not contain time code and are not prepared in real-time; *closed-captioned text* or simply *captioned text*, which are texts generated by a human captioner that contain a time code for every group of 3-5 words, and may or not be prepared in real-time; and *automatic speech recognition (ASR) output*, which denotes text generated by an ASR system that has time code associated with each word.

In this paper we present an alternative approach to off-line captioning [6]. Our goal is to automatically generate closed-caption text by aligning a program transcript with the ASR output. The program transcripts are highly accurate but lack time code information that is necessary to synchronize the text with the speech in a video program. The ASR output contains time code for each word uttered and can be used to synchronize the transcript. The accuracy of the text produced by current ASR systems is less than that produced by human captioner. In general, for a wide variety of real-world speech that includes combinations of speech with background noise, degraded acoustics, and non-native speakers, the ASR word error rate varies between 35% and 65% [7, 8, 9]. By aligning a program transcript with the ASR output we are able to generate a highly accurate and time-coded transcript, which can then be used as closed-caption text for the video program. A block diagram of the system is shown in Figure 3. The major processing steps are described below.

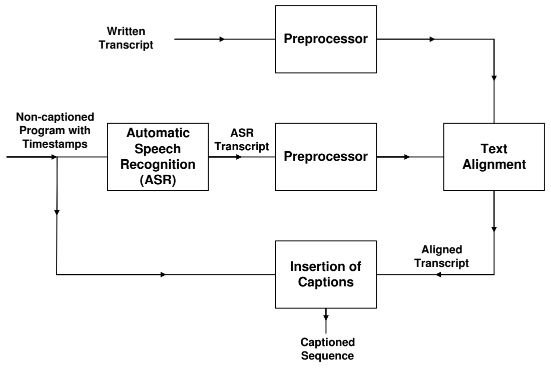


Fig. 3. Components of the proposed automated off-line closed-caption text generation system.

3.1. Preprocessing of Text

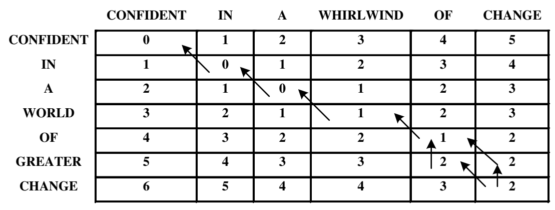
Given two input text files to be aligned, we first perform preprocessing on the files [6]. The goal of this step is to convert both texts into a standard form in order to simplify the core alignment procedure [10]. First, we divide the input text into units known as *tokens*, where each token is delimited by white space. Words within parentheses are ignored, since these are usually added by captioners in closed-caption text to provide extra information about the program. Then, each token is processed to remove all punctuation and non-alphanumeric characters. Uppercase characters are also converted to lowercase. For the ASR text, each token is labelled with a time code obtained from the ASR output.

3.2. Text Alignment

After the preprocessing step, the input texts are represented as two ordered sequences of tokens, S_1 and S_2. By alignment of these sequences we mean the following: We find a correspondence between tokens in S_1 and S_2 as to minimize some distance metric. This may require the insertion of spaces either into or at the ends of S_1 and S_2, so that every token in either sequence is matched with a token in the other sequence or a space.

The objective of aligning a program transcript with ASR output is to determine, for each token in the ASR output, the corresponding token in the program transcript. Once this is achieved, we can determine the time code of the tokens in the program transcript from the corresponding matching token in the ASR output. If no match for a token in the program transcript can be found in the ASR output, then its time code is estimated from the time code of its neighboring tokens [6].

In order to perform the alignment, a distance metric must be defined between token sequences that measures the quality of a particular alignment, i.e., if the distance is large, then the alignment is poor and vice versa. We have used the string edit distance [11] as our metric. The edit distance, $D(i, j)$, for two sequences, S_1 and S_2, is defined as the minimum number of edit operations needed to transform the first i tokens of S_1 into the first



	CONFIDENT	IN	A	WHIRLWIND	OF	CHANGE
CONFIDENT	0	1	2	3	4	5
IN	1	0	1	2	3	4
A	2	1	0	1	2	3
WORLD	3	2	1	1	2	3
OF	4	3	2	2	1	2
GREATER	5	4	3	3	2	2
CHANGE	6	5	4	4	3	2

Fig. 4. The dynamic programming table for the alignment of two sequences. The values in the cells are the edit distances and the arrows indicate possible paths.

j tokens of S_2. The allowed edit operations are insertion, deletion, and substitution.

An efficient method to calculate the total string edit distance, $D(M, N)$, between two sequences is to use a dynamic programming method [12]. In other words, in order to obtain $D(M, N)$ all the possible edit distances are obtained. This is done in an efficient manner using a recurrence relation that is used to obtain $D(i, j)$ based on previously obtained values. Starting from the initial conditions $D(i, 0) = i$ and $D(0, j) = j$, the distance values, $D(i, j)$, are obtained using the following relationship [11], for $i > 0$ and $j > 0$

$$D(i, j) = \min\left[D(i-1, j) + 1,\ D(i, j-1) + 1,\ t(i, j)\right] \quad (1)$$

where the token match function, $t(i, j)$, is defined as

$$t(i, j) = \begin{cases} 1 & \text{if } S_1(i) \neq S_2(j) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where $S_1(i)$ is the i^{th} token in sequence S_1 and $S_2(i)$ is similarly defined.

Once the edit distances are placed in a table format, as shown in Figure 4, a traceback procedure is used to extract the optimal alignment between the two sequences. For each cell in the table, an arrow is placed from that cell to the adjacent cell with the minimum value of the edit distance. Note that, there can be multiple arrows leading away from a cell. The optimal alignment is then found by tracing back the arrows from cell (M, N) to cell $(1, 1)$. If there is more than one arrow from a cell, the arrow to be followed is chosen randomly; therefore the final alignment may not be unique. The alignment is then recovered from the path by interpreting a horizontal step as an insertion and a vertical step as a deletion. A diagonal step from (i, j) to $(i-1, j-1)$ is interpreted as a match if $S_1(i) = S_2(j)$ and as a substitution otherwise.

Direct use of ASR technology could be used to generate both real time and off-line captions [6]. However, the accuracy for current ASR systems may not be able to achieve the acceptable accuracy of closed-caption text. We have also used our alignment system to obtain the accuracy of both ASR text and closed captioning text. It will be shown that the accuracy of current ASR technology falls below the accuracy of closed-caption text; therefore an ASR system can not be used as a stand-alone system to produce closed-caption text.

sequence name	number of speakers	FS	WG
cspan3	3	0	0
cspan4	4	0	0
cspan7	3	2	1
cspan11	4	0	0
cspan12	4	0	0
cspan15	9	0	1
cspan17	12	0	1
cspan18	6	4	2
cspan20	6	0	0
cspan21	5	1	0
cspan22	11	4	2
cspan25	8	2	3
cspan25	8	0	0
TOTAL	**82**	**13**	**10**

Table 1. Unique person detection results.

4. RESULTS

4.1. Unique Person Detection

For our speaker detection experiments we have selected 13 C-SPAN sequences that contain more than two persons. In evaluating the accuracy of our person detection system we have used the following rules

1. Only keyframes containing speakers were considered in our error analysis, that is, key frames containing the audience, wide shots, etc. were ignored.
2. A *false split* was declared if the keyframes belonging to the same speaker are split into two different clusters. For example, if the same person is split into three clusters, we count this as two false splits. However, shots of the same person obtained using different camera angles were not considered as false splits if they are grouped in different clusters.
3. A *wrong grouping* was declared if a keyframe for a person is grouped with that of another one in the same cluster. For example, if a cluster contains keyframes from three different people, we count this as two wrong groupings.

The speaker detection results for our data set are given in Table 4.1. From the results we observe that our system is accurately able to detect unique people appearing in a wide variety of programs. The errors are mostly localized to three sequences: $cspan18$, $cspan22$, and $cspan25$. The main source of errors for these sequences is what we call "voice overs," which occur when a shot of a person contains the audio for another person. Currently we are working on an updated version of our system that uses audio continuity to get rid of these errors.

4.2. Caption and ASR Accuracy

The text alignment algorithm was used to test the real-time captioning accuracy of nine C-SPAN programs. The accuracy of the captions is determined by first measuring the word error rate (WER), which is defined as

$$\text{WER} = \frac{\#\text{ tokens in error}}{\#\text{ tokens in program transcript}} \times 100 \quad (3)$$

	Program transcript	Closed-caption text
Spelling Errors	yasin	yassin
	pleuger	ploiga
	salahuddin	london
Interpretation Errors	we have	we've
	may 1	may 1st
	it's	it is

Table 2. Examples of different types of errors in the closed-caption texts.

	Program transcript	ASR output
Interpretation Errors	Dr	Doctor
	npc	n.p.c.
Homophone Substitution Errors	Ba'ath Party	path party
	Hiroshi Yoshisuga	hiroshi you she sued to
	the al Qaeda	be out candy
	outside pressure	all side pressure
	they are	there

Table 3. Examples of different types of errors in the automatic speech recognition output.

The accuracy is then defined as

$$\text{accuracy} = 100 - \text{WER} \quad (4)$$

We have classified errors found within the caption text into two major categories: spelling and repetition errors. Examples of these errors are illustrated in Table 2. We have also classified errors within the ASR output into two major categories, misinterpretation and homophone errors, which are illustrated in Table 3.

The accuracies of closed-caption text and text produced by the ASR system are compared in Table 4. The performance of the ASR system can show large deviations depending on the particular program, audio quality, and speaker. The errors produced by captioners generally reflect the same context as the correct text, albeit with spelling errors. On the other hand, the ASR system can make errors that are totally unrelated to the context of the correct text. For example, the ASR system can detect "Qaeda" as "candy," whereas a captioner can misspell it as "kindi".

5. CONCLUSIONS

In this paper we described two systems that automate important processes for C-SPAN programs and presented the results of these systems. Our unique person detection system can accurately determine persons appearing in a news program. This system can be used to insert on-screen graphics automatically in programs. We have used text alignment to align program transcripts and ASR output text to generate closed-caption text with accurate time codes for each word. Text alignment can be efficiently performed using dynamic programming to find the minimum edit distance between the two texts. We have processed a number of news programs using our system. The error rates for closed-caption texts and ASR outputs for these programs were examined. The text alignment method that we presented can produce highly accurate closed-captions efficiently.

	Closed-captions	ASR output
Mean	84.84%	55.02%
Median	87.04%	59.98%
Std. deviation	9.81	17.55

Table 4. Comparison of total accuracies for closed-caption text generated by captioners and ASR output.

6. REFERENCES

[1] 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 Transactions on Multimedia*, vol. 6, no. 1, pp. 103–118, February 2004.

[2] A. Albiol, L. Torres, and E. J. Delp, "An automatic face detection and recognition system for video indexing applications," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Orlando, FL, 2002, pp. 3644–3647.

[3] C. Taskiran, A. Albiol, L. Torres, and E. Delp, "Detection of unique people in news programs using multimodal shot clustering," *Proceedings of the International Conference on Image Processing (ICIP 2004)*, Singapore, October 24-27 2004, submitted.

[4] "http://www.ncicap.org/."

[5] "http://www.fcc.gov/."

[6] A. Martone, C. Taskiran, and E. Delp, "Automated closed-captioning using text alignment," *Proceedings of the SPIE International Conference On Storage and Retrieval Methods and Applications for Multimedia*, San Jose, CA, January 20-22 2004, pp. 108–116.

[7] M. Witbrock and A. Hauptmann, "Improving acoustic models by watching television," Carnegie Mellon University, Tech. Rep. CMU-CS-98-110, March 1998. [Online]. Available: citeseer.nj.nec.com/witbrock98improving.html

[8] J. Choi, D. Hindle, J. Hirschberg, I. Magrin-Chagnolle, C. Nakatani, F. Pereira, A. Singhal, and S. Whittaker, "Scan - speech content based audio navigator: A systems overview," *Proceedings of the International Conference on Spoken Language Processsing (ICSLP 1998)*, Sydney, Australia, 30th November-4th December 1998.

[9] S. Srinivasan and D. Petkovic, "Phonetic confusion matrix based spoken document retrieval," *Proceedings of the International Conference on Research and Development in Information Retrieval (SIGIR 2000)*, Athens, Greece, July 24-28 2000, pp. 81–87.

[10] D. T. F. Popowich, P. McFetridge and J. Toole, "Machine translation of closed captions," *Machine Translation*, vol. 15, pp. 311–341, 2000.

[11] D. Gusfield, *Algorithms on Strings, Trees, and Sequences*. Cambridge University Press, New York, NY: HighText Interactive Inc, 1997.

[12] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, 2nd ed. Cambridge, MA: MIT Press, 2001.