Summarization of Videotaped Presentations: Automatic Analysis of Motion and Gesture

Shanon X. Ju, Michael J. Black, *Member, IEEE*, Scott Minneman, and Don Kimber

***Abstract*— This paper presents an automatic system for analyzing and annotating video sequences of technical talks. Our method uses a robust motion estimation technique to detect key frames and segment the video sequence into subsequences containing a single overhead slide. The subsequences are stabilized to remove motion that occurs when the speaker adjusts their slides. Any changes remaining between frames in the stabilized sequences may be due to speaker gestures such as pointing or writing, and we use active contours to automatically track these potential gestures. Given the constrained domain, we define a simple set of actions that can be recognized based on the active contour shape and motion. The recognized actions provide an annotation of the sequence that can be used to access a condensed version of the talk from a Web page.**

***Index Terms*— Automatic video annotation, gesture tracking and recognition, key-frame detection, robust motion estimation, video summarization.**

I. Introduction

BOTH analog and digital video are becoming increasingly common forms of documentation both at home (home videos) and at work (videotaped meetings and presentations). As a document, however, video (even digital video) lacks many of the basic properties that we associate with more familiar electronic document types such as Microsoft Word documents. Digital video is not easy to browse, edit, or search by content. Despite its shortcomings, the presence of digital video will continue to increase with the spread of multimedia workstations and the desire to put video on the Internet. To make digital video more like traditional electronic text, documents will require the extraction of *content* and *structure* from the video.

Research on the analysis of structure and content in video has progressed along two different dimensions. The majority of work has focused on the detection of key frames and scene breaks in general, unconstrained, video databases [17], [28], [29]. For these methods to work on general video sequences, simple image processing techniques are used that typically do not perform high-level analysis of the content of the sequence. In our work, we have chosen to constrain the domain of video sequences that we wish to analyze, and look specifically at a workplace scenario of videotaped presentations in which the camera is focused on the speaker's slides; the camera may be mounted so as to view the slides from above or as they are projected on a screen. By constraining the domain, we are able to define simple actions that people perform during a presentation, and we can automatically extract this information about the content of the video. By automatically recognizing these actions, we can provide an annotation of the video sequence that can be used, for example, to access a condensed version of the talk from a Web page.

Fig. 1 shows a simple example of a video browsing and indexing system [16] that illustrates our motivation. The original video stream is summarized and annotated off line. In previous versions of the system, this was a manual process, and the present paper addresses the automation of this process. The outputs of the process are indexes of the events and the images corresponding to these events. This information is used to make a summary Web page containing images of each event and their time indexes. Clicking on the image, the users can go to a new page with more information about this event. For instance, this page may contain a high-resolution image so that the users can actually read the words on the slides. Clicking on a time index causes the video of the presentation to begin playing at that time instant.

Generally speaking, the goal of automatic video annotation is to save a small set of frames that contain most of the relevant information in the video sequence. That is, we want to find where the important changes occur. In our restricted domain of overhead presentations, a number of "changes" can occur in the image sequence. There are two classes which we will call "nuisances" and "affordances."

Nuisance changes are those which we define to have no relevant semantic interpretation (Fig. 2). Examples of this are when the speaker occludes the slide with his hand or body or when the speaker moves the slide (an action that we observe to be very common). These nuisance changes are ones that we wish to ignore in the analysis and summarization of the video sequence.

Affordances, on the other hand, are changes in the video sequence that have a semantic interpretation with respect to the presentation (Fig. 3). For example, speakers often write, point, or make repetitive gestures at locations on the slide to which they are referring. Another common action is to cover a portion of the slide and gradually reveal the underlying text. We call these changes "affordances" because we can take advantage of them to acquire more information about the presentation (see Gibson [8]). As we will show, recognition of the affordances will allow us to produce annotated key frames from the video

Manuscript received October 31, 1997; revised May 30, 1998. This paper was recommended by Associate Editor M.-T. Sun.

S. X. Ju is with the Department of Computer Science, University of Toronto, Toronto, Ont. MSS 1A4, Canada.

M. J. Black, S. Minneman, and D. Kimber are with the Xerox Palo Alto Research Center, Palo Alto, CA, USA.

Publisher Item Identifier S 1051-8215(98)06328-9.

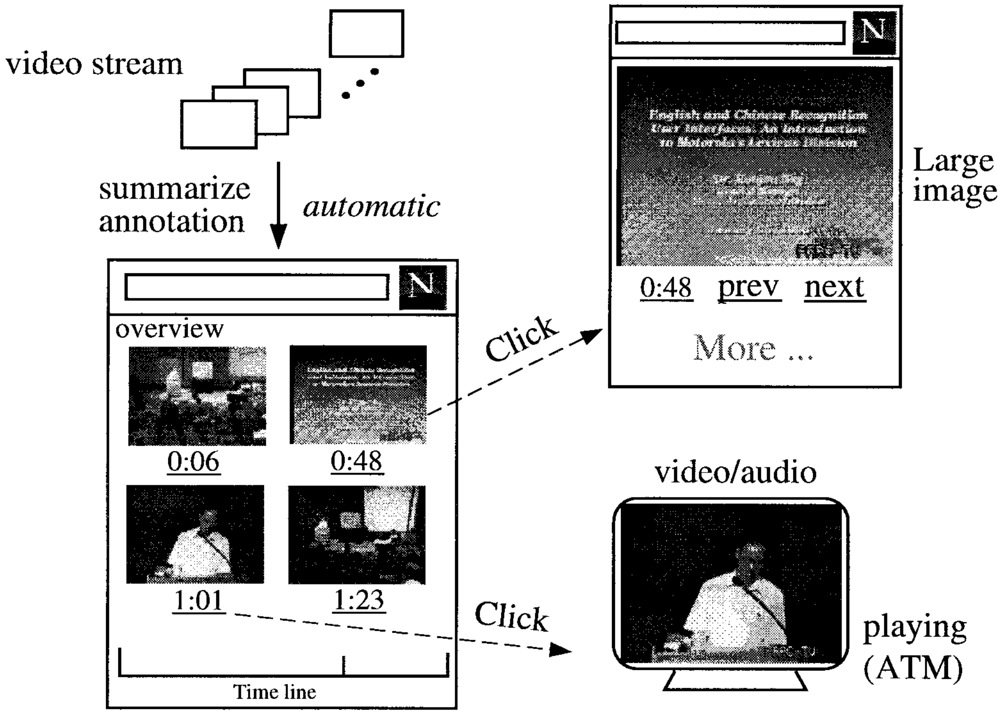


Fig. 1. An example Web-based interface for video browsing.

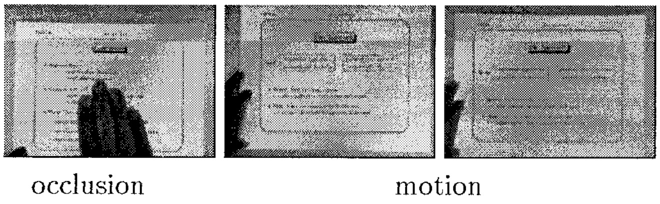


Fig. 2. Nuisance changes.

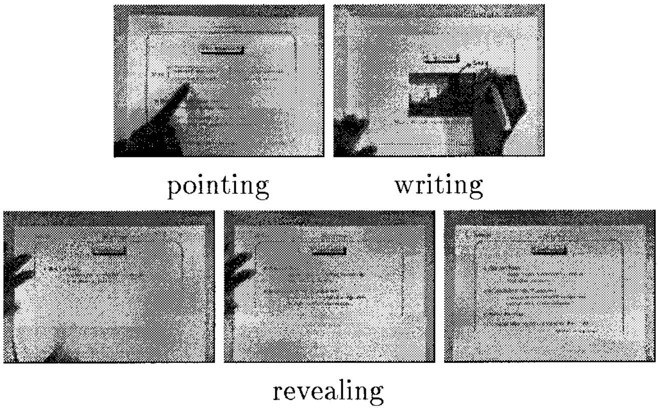


Fig. 3. Affordances.

that will allow users to later access portions of the talk where the speaker gestured at a particular location on his slides.

This paper describes a system for the automatic annotation described above. First, we estimate the global image motion between every two consecutive frames using a robust regression method. The motion information is used to compute a warped sequence where the slides are "stabilized." Second, the stabilized sequence is processed to extract *slide templates*, or key frames. The key-frame sequence is a condensed representation of the slides shown in the meeting. Third, we compute a pixelwise difference image between the slide templates and the corresponding frames in the stabilized image sequence. These difference images contain only occluded/disoccluded image regions which correspond to potential gestures. These gestures are tracked using a deformable contour model. By analyzing the shape of the contour and its motion over time, we recognize pointing gestures and recover the location on the slide to which the speaker is referring. Finally, the key frames and gesture information are combined to produce an annotated Web page of the presentation.

II. Related Work

The main themes explored in previous work on automatic video summarization can be broadly described as *segmentation*, *analysis*, and *presentation*. Segmentation focuses on finding scene changes or key frames in the video, while analysis focuses on detecting events or understanding actions (typically in a more restricted domain). Once the video has been processed, it must be presented to allow users to access the data. A complete video summarization system must address each of these aspects.

Scene-break detection is a first step toward the automatic annotation of digital video sequences. While in general video sequences scene breaks include *cuts*, *dissolves*, and *fades*, in our domain, a "scene change" corresponds to the speaker changing slides.

There are two basic types of algorithms for scene-break detection. The first uses image-based methods, such as image differencing and color histograming [17], [29]. These image-based differencing methods tend to oversegment the video sequence when there is motion present in the scene or when the camera is moving since many pixels will change their color from frame to frame. A second class of approaches tries to account for scene motion using motion information computed from global models, local motion information, the motion of image features [28], or global models fitted to local motion measurements [23], [30]. The most efficient methods extract motion information directly from motion JPEG or MPEG compressed video [23], [30]. While these methods can detect and classify scene breaks that are difficult to detect with image-based methods, multiple moving objects still present difficulties. In general, these image- and motion-based methods use simple models of scene change, and can provide a reasonable coarse segmentation of the video.

However, if key frames are generated at every cut point, a large number of key frames must be displayed to provide an overview of a long video sequence [25]. For example, the shots (scene breaks) of a video containing dialog usually contain repeated similar images. Additional clustering methods [2], [24], [32] can be applied to classify the video shots into groups. Color is the main image attribute used in these clustering algorithms. Yeung and Yeo [24], [25] also used shape and spatial correlation to measure similarity of the shots. Furthermore, they [26] proposed the *scene transition graph* to represent the content of the video.

In addition to work on video segmentation, a number of approaches exploit domain knowledge to parse the content of a video sequence using *a priori* models of the video's spatial and temporal structure. For example, Zhang *et al.* [31] and Chen *et al.* [7] presented methods to analyze TV news programs. Zhang *et al.* [31] defined image models of shots containing an anchorperson, and used these to split the news program into independent subjects. Chen and Faudemay [7] also detected anchorperson shots, but did so using audio information.

Other approaches summarize general videos by detecting special events. Lienhart *et al.* [14] analyzed both video and audio information to detect events, such as text appearing in the images, close-up shots of actors, explosions, etc. Smith and Kanade [20] also extracted significant information from video and audio. They identified "key words" from transcripts and detected video shots that had camera motion or contained faces or text.

Vinod and Murase [21] propose a video summarization method that exploits tracking of people in the scene. Certain types of key frames are detected, such as when two objects are close to each other, or when there is a change of visibility or size. They used color similarity for tracking manually selected image regions. Our method also uses tracking information to detect and recognize important events, but given our constrained domain, this tracking can be performed completely automatically.

The above approaches perform segmentation and low-level content analysis of video. They parse the video into its constituent parts without performing a detailed analysis of the action within each part. In each case, the analysis involves the extraction of key frames using some measure of similarity between frames. In general, key-frame detection is domain dependent, and the appropriate method and formulation will be heuristic and will depend on the application domain [14].

A number of authors have attempted to extract high-level semantic information about the action within video shots. Current methods focus on narrow domains which Intille and Bobick [9] refer to as "closed worlds." They, for example, track and analyze the motion of players in a football game. A similar analysis of soccer highlights was presented by Yow *et al.* [27]. They tracked a soccer ball and detected "exciting" actions such as when the ball was close to the goalposts. These two "closed-world" domains do not require the segmentation of the video into multiple scenes (cut detection). Like us, however, they define narrow domains in which they can express prior assumptions about the semantics of changes in the scene.

Other recent attempts to provide an analysis of video in restricted domains include the work of Mann *et al.* [15] and Siskind *et al.* [19] who propose methods for analyzing the physical interactions between objects in a video sequence, and that of Brand [6] who looks at understanding human actions in video for the purpose of video summarization. Kollnig *et al.* [13] have defined a vocabulary of motion verbs that are used to analyze the behavior of cars in video sequences of traffic scenes. Earlier work on the linguistic analysis of action in video focused on generating descriptions of a soccer match given manually generated motion information [1], [18].

In this paper, we perform scene cut detection using global motion information; this is appropriate for our particular domain. Additionally, given our application, we automatically track and recognize simple human gestures during a videotaped talk. While constrained, the domain we consider has a rich set of actions and gestures that need to be recognized. Additionally, our current system provides a useful solution to the real problem of browsing previously recorded presentations.

III. Motion Estimation

Our goal is to estimate slide motion between every pair of frames in order to detect slide changes and to stabilize the image sequence to remove nuisance changes due to motion. In our context, we may assume that the slides are always roughly perpendicular to camera's viewing axis, and they can be modeled as a rigid plane. Given these assumptions, the image motion can only be translation, scaling, or rotation which can be modeled as

$$u(x, y) = a_0 + a_1 x - a_2 y \tag{1}$$

$$v(x, y) = a_3 + a_2 x + a_1 y \tag{2}$$

where $\boldsymbol{a} = [a_0, a_1, a_2, a_3]$ denotes the vector of coefficients to be estimated, and $\boldsymbol{u}(\boldsymbol{x}; \boldsymbol{a}) = [u(x, y), v(x, y)]^T$ are the horizontal and vertical components of the motion at image point $\boldsymbol{x} = [x, y]$. The coordinates (x, y) are defined with respect to the center of the image.

To estimate the motion coefficients $\boldsymbol{a}$, we employ a robust regression approach that makes the assumption that the brightness pattern within the image region $\mathcal{R}$ remains constant while the patch may translate, scale, or rotate. The coefficients are recovered by minimizing

$$E(\boldsymbol{a}) = \sum_{\boldsymbol{x}\in\mathcal{R}} \rho(I(\boldsymbol{x}+\boldsymbol{u}(\boldsymbol{x};\boldsymbol{a}),\, t+1) - I(\boldsymbol{x},\, t),\, \sigma) \tag{3}$$

with respect to the coefficients $\boldsymbol{a}$. The term $I(\boldsymbol{x}+\boldsymbol{u}(\boldsymbol{x};\boldsymbol{a}),\, t+1)$ represents the image brightness function at time $t+1$ warped by the estimated motion field $\boldsymbol{u}(\boldsymbol{x};\boldsymbol{a})$. A robust ρ function is used to cope with violations of the brightness constancy assumption that occur when the speaker occludes the slides. The value of σ controls the rejection of "outlier" measurements.

For the experiments in this paper, we take ρ to be

$$\rho(r,\, \sigma) = \frac{r^2}{\sigma^2 + r^2} \tag{4}$$

which is the robust error function used in [3]. Pixel $\boldsymbol{x}$ is defined to be an "outlier" if $|I(\boldsymbol{x}+\boldsymbol{u}(\boldsymbol{x};\boldsymbol{a}),\, t+1) - I(\boldsymbol{x},\, t)| > 2.5\sigma$.

The coefficients are estimated in a coarse-to-fine fashion using a simple iterative coordinate descent procedure. For a detailed description of robust motion estimation and the minimization procedure, the reader is referred to [3].

The robust formulation of (3) means that the algorithm estimates the dominant motion in the scene (i.e., the slide motion) and automatically ignores the image points that belong to other motions (the gestures). Gesture tracking is described in Section V. In the following section, we will use the motion information to *stabilize* the image sequence with respect to *key frames*.

IV. Key-Frame Detection

Given an image sequence corresponding to a particular slide, stabilization is just a process of *warping* each of the images toward a *reference image* by taking into account the cumulative motion estimated between each of the frames in the sequence. Since minimizing (3) can only estimate small image motions between two consecutive images, we need to compute the motion between the *reference image* and each following frame. Given $\boldsymbol{u}(\boldsymbol{x};\, \boldsymbol{a}_{n-1})$, the motion between the *reference image* and frame $n-1$, and $\boldsymbol{u}(\boldsymbol{x};\, \boldsymbol{a}_n^*)$, the motion between frame $n-1$ and n, the motion coefficients between the *reference image* and frame n are given by

$$\begin{aligned}\boldsymbol{a}_n &= \boldsymbol{a}_{n-1} + \boldsymbol{a}_n^* + d\boldsymbol{a}\\ d\boldsymbol{a} &= [a_{n,1}^* a_{n,2}^*]\begin{bmatrix} a_{n-1,0} & a_{n-1,1} & a_{n-1,2} & a_{n-1,3}\\ a_{n-1,3} & -a_{n-1,2} & a_{n-1,1} & -a_{n-1,0}\end{bmatrix}\end{aligned}$$

where $a_{n-1,3}$, for example, represents the coefficient a_3 from the previous frame $n-1$. The motion coefficients $\boldsymbol{a}_n$ describe the motion from the first image, or the reference image, toward the current frame. Computing a warped image involves using the motion vectors $\boldsymbol{u}(\boldsymbol{x};\, \boldsymbol{a}_n)$ to find a location $(\hat{x},\, \hat{y})$ in the current image which corresponds to pixel $\boldsymbol{x} = (x,\, y)$ in the reference image. Bilinear interpolation of the pixels in the neighborhood of $(\hat{x},\, \hat{y})$ gives the brightness value at $(x,\, y)$ in the warped image. All frames in the sequence are warped toward the reference frame.

We use a simple heuristic that the reference frame is the first nonblank image for which the motion is smaller than a threshold. For subsequent images, if the motion estimation method succeeds (has low error), then the image belongs to the same subsequence as the reference frame. If the outlier rate (i.e., the percentage of outlier points in the region) is larger than a threshold (55% in our experiments), the motion estimation algorithm is considered to have failed. This means that two consecutive frames are significantly dissimilar and cannot be modeled by (1) and (2). This situation typically corresponds to a change of slides, and we use the current frame to end the subsequence and begin looking for the next stable reference image. In the domain of overhead presentations, this simple strategy works well.

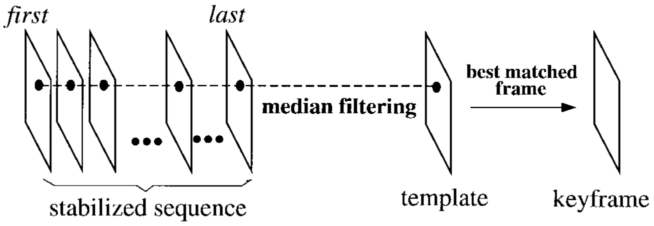


Fig. 4. The key-frame detection: filtering the stabilized sequence.

Since we only estimate the dominant (slide) motion in the scene, the warped sequence contains both the stabilized slide and moving objects, such as the hand of the speaker. To compute a template image that contains no gestures, we use a median temporal filter to remove the moving objects in the sequence [22]. At each image position, we take all of the values at this position in the stabilized frames, find the median value, and use it as the intensity value of the slide template.

Finally, we find which of the stabilized frames is most similar to the template. An error measure is defined for each frame at time t:

$$E(t) = \sum_{(x,\, y)\in\mathcal{R}} (I_{\text{template}}(x,\, y) - I_{\text{warped}}(x,\, y,\, t))^2 \tag{5}$$

where $\mathcal{R}$ represents the set of all pixels in the image, I_{template} is the template image, and I_{warped} is the stabilized image. We find the minimal error $E(t)$ among all of the frames, say $t = t_0$, and use the stabilized image $I_{\text{warped}}(t_0)$ as the "key frame" which is representative of the corresponding video segment (see Fig. 4).

A. Experimental Results

Fig. 10 shows snapshots taken every 4 s from a subsequence of a videotaped presentation. The subsequence is approximately 104 s in length, was recorded at 15 frames/s, was JPEG compressed, and the images were 320 × 240 pixels in size. Four slides were presented by the speaker during this subsequence, and the speaker moved, pointed at, and occluded the slides.

Fig. 11 shows the automatically detected key frames. All four slides are detected successfully. Note that, while it did not occur with this sequence, the median filter can produce unexpected results. For instance, if the speaker puts his hand on the slide for the majority of frames in a given segment, the hand will be considered as part of the key frame or template. More sophisticated techniques would be required to distinguish

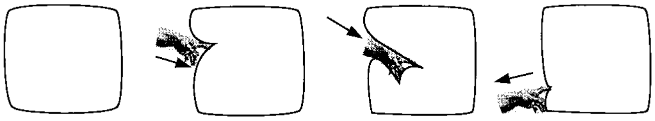


Fig. 5. Gesture tracking: a deformable image boundary.

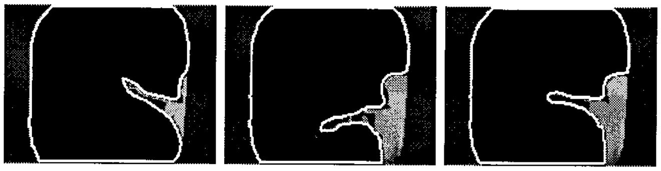


Fig. 6. Snake tracking at frames 2361, 2367, and 2375.

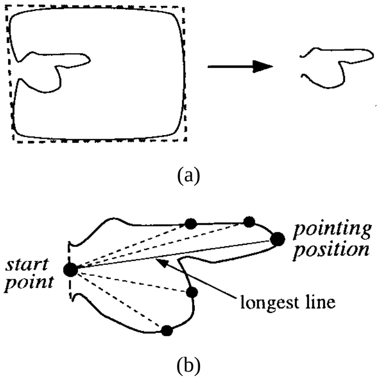


Fig. 7. (a) Finding the contour of the object. (b) Finding the pointing position.

hands from slides, but it is not clear whether this is either necessary or desirable.

V. Gesture Tracking

If we compute the absolute difference between the key frame and images in the warped sequence, the nonzero pixels in the image must correspond to gestures, covered data, or written text. Since all of the gestures must enter the scene or leave the scene from the image boundary, new material cannot suddenly appear in the middle of the image.[1] Therefore, we can let the image boundary deform when a "thing" enters, such that it tracks the contour of the entering object. If the object leaves, the deformable contour will expand to the image boundary (see Fig. 5).

We use controlled continuity splines, or "snakes" [12], to model the deformable contour. The behavior of a snake is controlled by internal forces that serve as a smoothness constraint and the external forces that guide the active contour toward image features. Following the notation in [12], given a parametric representation of an image curve $\boldsymbol{v}(s) = (x(s), y(s))$, the energy function is defined as

$$\mathcal{E}_{\text{snake}} = \int_0^1 \left(\alpha|\boldsymbol{v}_s(s)|^2 + \beta|\boldsymbol{v}_{ss}(s)|^2\right)/2 + \mathcal{E}_{\text{ext}}(\boldsymbol{v}(s))\, ds. \tag{6}$$

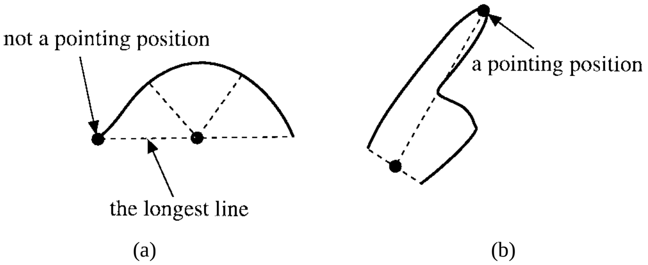


Fig. 8. Gesture recognition: the spatial constraint.

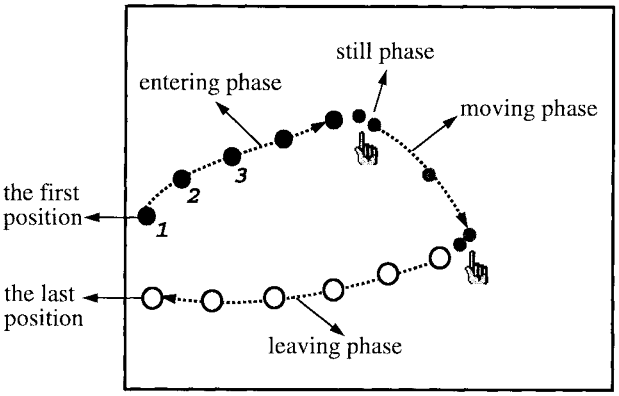


Fig. 9. Gesture recognition: the temporal constraint.

The first- and second-order derivative terms ($\boldsymbol{v}_s$ and $\boldsymbol{v}_{ss}$,, respectively) represent the internal energy. $\mathcal{E}_{\text{ext}}$ represents the external potential

$$\mathcal{E}_{\text{ext}} = P(x, y) - K(x, y) = c[G_\sigma * \Psi(x, y)] - K(x, y) \tag{7}$$

where c is a constant weight, Ψ is the absolute difference between the key frame and a stabilized image, and $G_\sigma * \Psi$ denotes the difference image convolved with a Gaussian smoothing filter. $K(x, y)$ is zero if pixel (x, y) is at the image boundary; otherwise, $K(x, y)$ is a constant. The snake is initialized at the image boundary and, in the absence of any other force, $K(x, y)$ will force the contour to remain at the image boundary. The active contour model in (6) attempts to find a contour which is both smooth and which minimizes the value of $P(x, y)$ at every snake node (x, y). $P(x, y)$ is a scalar potential function defined over the image plane. If the value of P is large over some region that overlaps the boundary of the image (because the hand has entered the slide), then the snake will deform around the region until $P(x, y)$ is small enough along the contour and both the internal and external forces are balanced. If the hand leaves the frame, then $K(x, y)$ will pull the snake out to the image boundary.

Fig. 6 shows how the snake deforms to find the object boundary. The image shows the absolute difference images between a series of frames and the corresponding key frame. Bright areas correspond to the hand of the speaker making a pointing gesture. In the figure, the white closed contour represents the snake.

Tracking can fail when the entering object has a color similar to that of the slide or is extremely thin (e.g., a metal

[1] Laser pointers are an exception to this rule, and are currently not handled by the system. A separate detector tuned to the color of laser pointers could be added to the system.

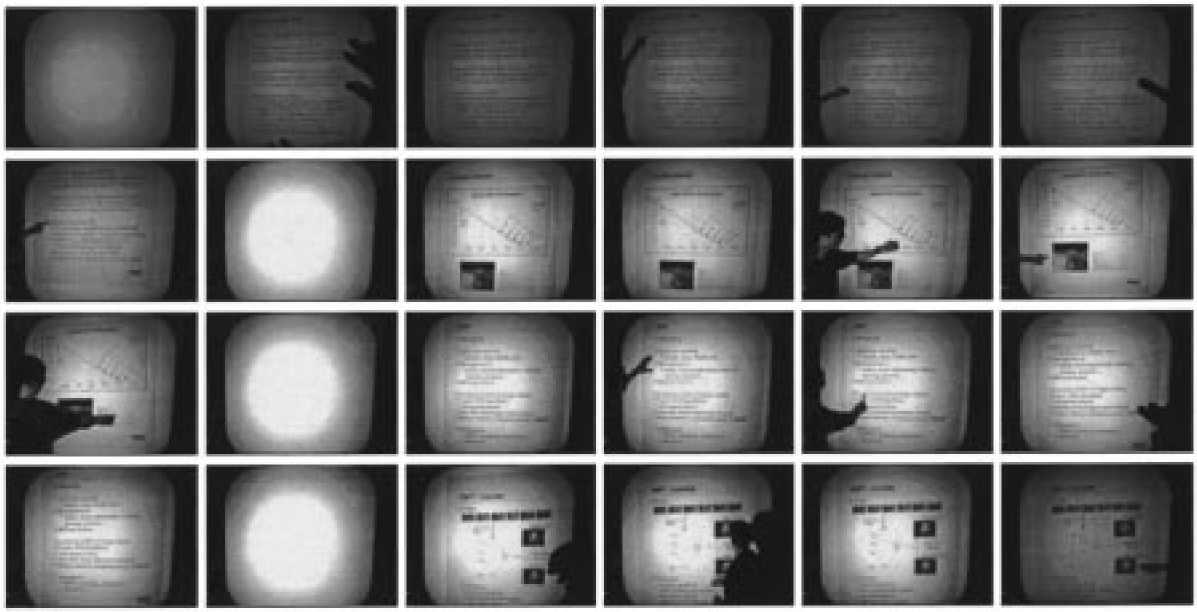


Fig. 10. Snapshot frames taken from the video sequence; shown every 4 s.

pointer). In this case, the magnitude of the image difference in the region of the object may be insufficient to attract the snake and overcome its internal forces.

VI. Recognizing Pointing Gestures

From the snake nodes, we can detect if, and where, the speaker is pointing on the slides. Our method has three steps. First, we fit a bounding quadrilateral to the snake nodes by finding the four corners. The nodes that are not close to any edge of the quadrilateral belong to the deformed part of the snake, that is, the contour of the object [Fig. 7(a)].

Second, we define a starting point to be the middle point between the first node and the last node of the contour [Fig. 7(b)]. Among all of the snake nodes on the contour, the one that is furthest from this starting point is defined as the pointing position. The line which connects the starting point and the pointing position will give us the rough pointing direction.

Finally, we recognize pointing gestures using two heuristics to filter out nongestures. The first heuristic is a constraint on the spatial structure of the gesturing object; we want it to have a definite "point." The pointing positions that are too close to the first or last nodes in the segmented gesture are therefore eliminated [Fig. 8(a)].

The second heuristic models the expected temporal pattern of pointing gestures. The temporal pattern of pointing positions is segmented into the entering phase, leaving phase, moving phase, and still phase. Fig. 9 illustrates two common temporal patterns, where the circles stand for pointing positions at each frame, and the dashed arrow lines indicate the temporal order of these positions. The left pattern represents an action that contains two valid pointing gestures, which are marked by the hand icon. The speaker pointed at the first position for a few frames, then he moved to the second position and stayed there shortly before he left the scene.

We define three temporal gesture models. First, a single still gesture is characterized by the hand entering, pausing for roughly 1/3 of a second or longer, and then exiting. Second, multiple gestures can occur in which the hand enters, pauses, then moves and pauses an arbitrary number of times before exiting. Finally, we recognize waving gestures in which the hand enters and never comes to rest, but rather moves continuously within a small spatial neighborhood; to determine the location of a waving gesture, we select a pointing position in the moving phase, which is surrounded by most other pointing positions. For all other temporal patterns that do not match these models, no gestures are recognized. More sophisticated techniques could be employed for recognizing complex gestures (e.g., [5]).

VII. Experimental Results

To illustrate the performance of the system, we show the results of processing two different video sequences. We show only a subset of the video for each presentation. The presentations were captured using a standard video camera pointed at a screen. A standard overhead projector was used to display the transparencies. The presentations were recorded to video tape and digitized off line. In the first presentation, the speaker was familiar with the system. The second example was captured from a talk at Xerox PARC in which the speaker had no knowledge of the system and its performance.

A. Presentation 1

Recall the test subsequence illustrated in Fig. 10. Fig. 11 shows the automatically detected key frames. All four slides are detected successfully. Fig. 12 shows the recognized pointing gestures for each slide. In order to illustrate the accuracy

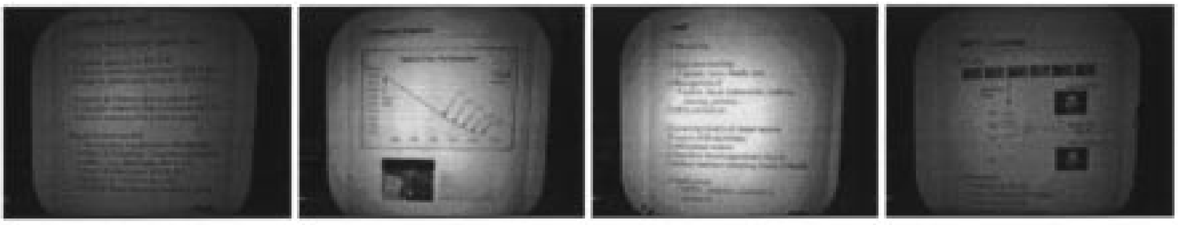
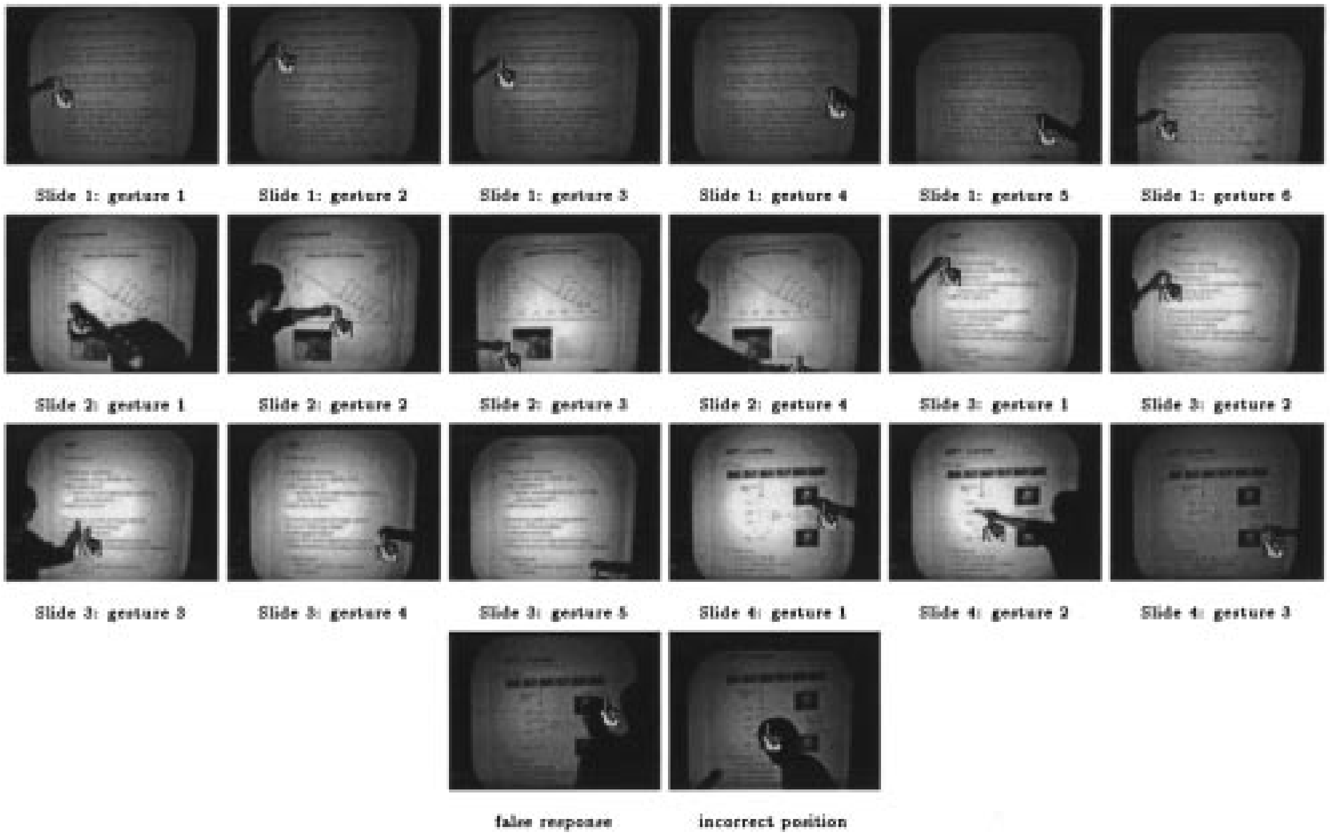


Fig. 11. Detected key frames for Presentation 1.



Slide 1: gesture 1
Slide 1: gesture 2
Slide 1: gesture 3
Slide 1: gesture 4
Slide 1: gesture 5
Slide 1: gesture 6
Slide 2: gesture 1
Slide 2: gesture 2
Slide 2: gesture 3
Slide 2: gesture 4
Slide 3: gesture 1
Slide 3: gesture 2
Slide 3: gesture 3
Slide 3: gesture 4
Slide 3: gesture 5
Slide 4: gesture 1
Slide 4: gesture 2
Slide 4: gesture 3
false response
incorrect position

Fig. 12. Recognized pointing gestures and their positions.

of the recovered pointing positions, a hand icon is shown on top of the warped frame where a gesture is detected.

System performance is compared with a manual classification of the video into subsequences and gestures. The results are promising; 18 out of 20 detected pointing gestures are correctly identified. The two errors are shown in the last row of Fig. 12. The overall error rate for this sequence is 10%. A false response happened when the speaker occluded the projection screen and stood still in this position for a few frames. The method cannot distinguish whether the object is a pointer, the hand or arm of the speaker, or the body of the speaker. Additional shape or appearance models of objects would be required to reduce this kind of false response. The only pointing gesture missed by the system occurred when the speaker's head was occluding the slide simultaneously with the gesture. The algorithm found the larger of the two entering objects (the speaker's head) and classified it as a pointing gesture.

We created a Web-based interface for this sequence to allow users to access the original video and audio at the points relevant to their interests. The first page (Fig. 15) contains the overview of the talk, which includes the title of the talk, the name of the speaker, and the automatically detected slides. Each slide image is linked to a summary page that contains the gesture information (Fig. 16). At the top of the page, three consecutive slides are shown. The center one is the current slide. On either side of it are the previous and next slides, respectively. The user can go back and forth by clicking on the prev/next thumbnails. The detected pointing gestures are shown in three ways. The image indicates where the speaker pointed. The bar beneath the image indicates when the pointing gestures occurred in the video stream of the current slide. An empty bar indicates that the gesture was at the beginning of that slide's sequence, and a full bar indicates that the gesture was at the end. The time in seconds indicates when the gesture occurred from the beginning of the talk.

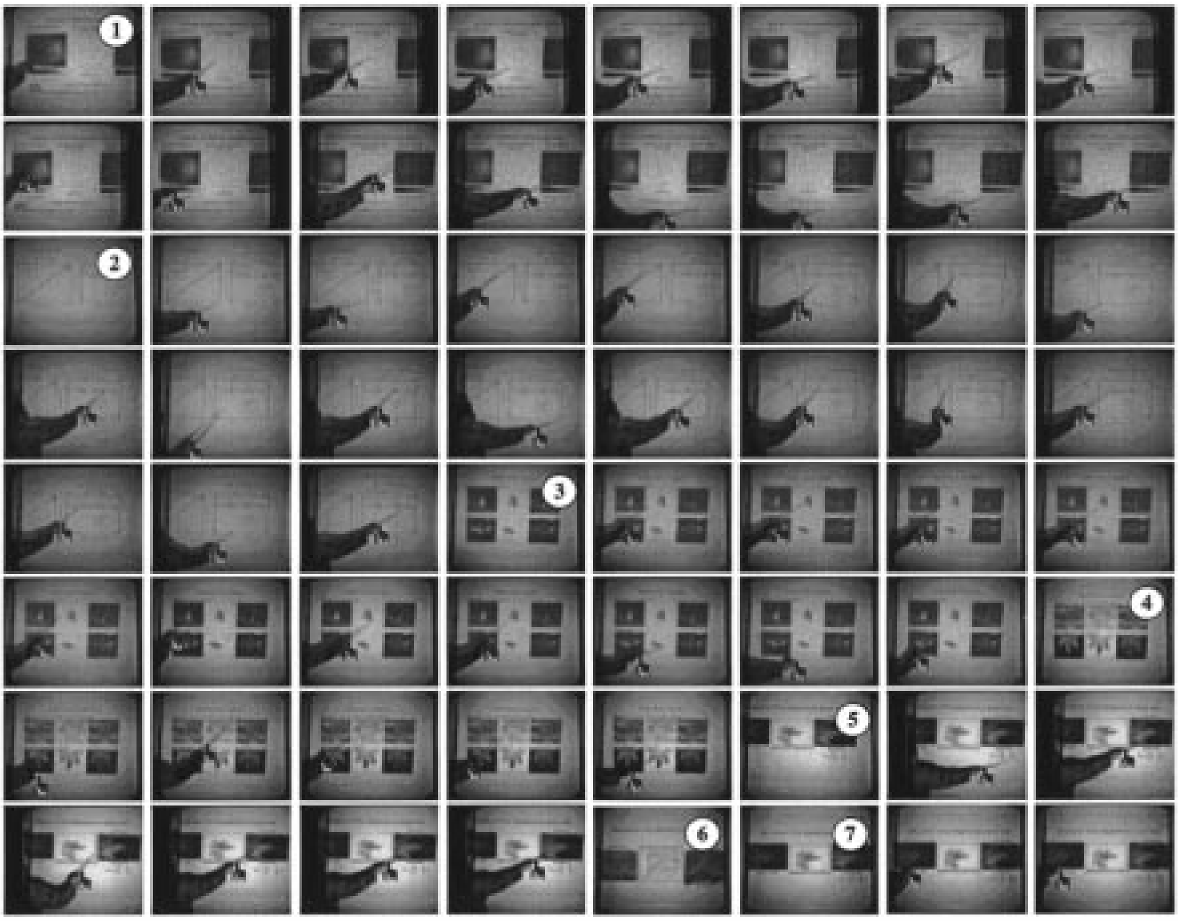


Fig. 13. Detected key frames (numbered images) and correctly recognized gestures.

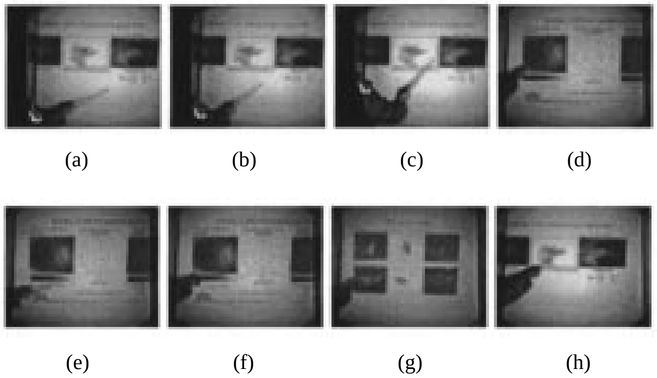
B. Presentation 2: Naive User

To cope with a longer sequence (over 9 min) we used a smaller image size (160 × 128 pixels, 15 frames/s, JPEG compressed) without any detectable loss of accuracy. In this subsequence, the speaker showed eight slides.

Fig. 13 shows seven of the eight automatically detected key frames. The fifth and sixth slides were shown twice. Our current method will not recognize if a slide has been shown before. Note that a gesture appears in the first key frame. Since the speaker pointed to a single region for a long time, the gesture was not removed by the median filter.

The speaker also used a thin metal pointer to gesture at the slides; this pointer was too small to be detected by the system given the image size and quality. The system, however, recognized those gestures when the arm of the speaker appeared in the image. The gesture model would need to be extended to recognize gestures involving thin pointing devices or laser pointers. Alternatively, speakers could be instructed to use particular types of pointing devices to aid later electronic access.

The speaker often pointed at one area of the slide, took back his arm, and then quickly pointed at the same area again. Therefore, many detected gestures actually point to the same region. Also, many of the pointing gestures involved the speaker waving his arm around the general area to which he was referring.



(a) (b) (c) (d)

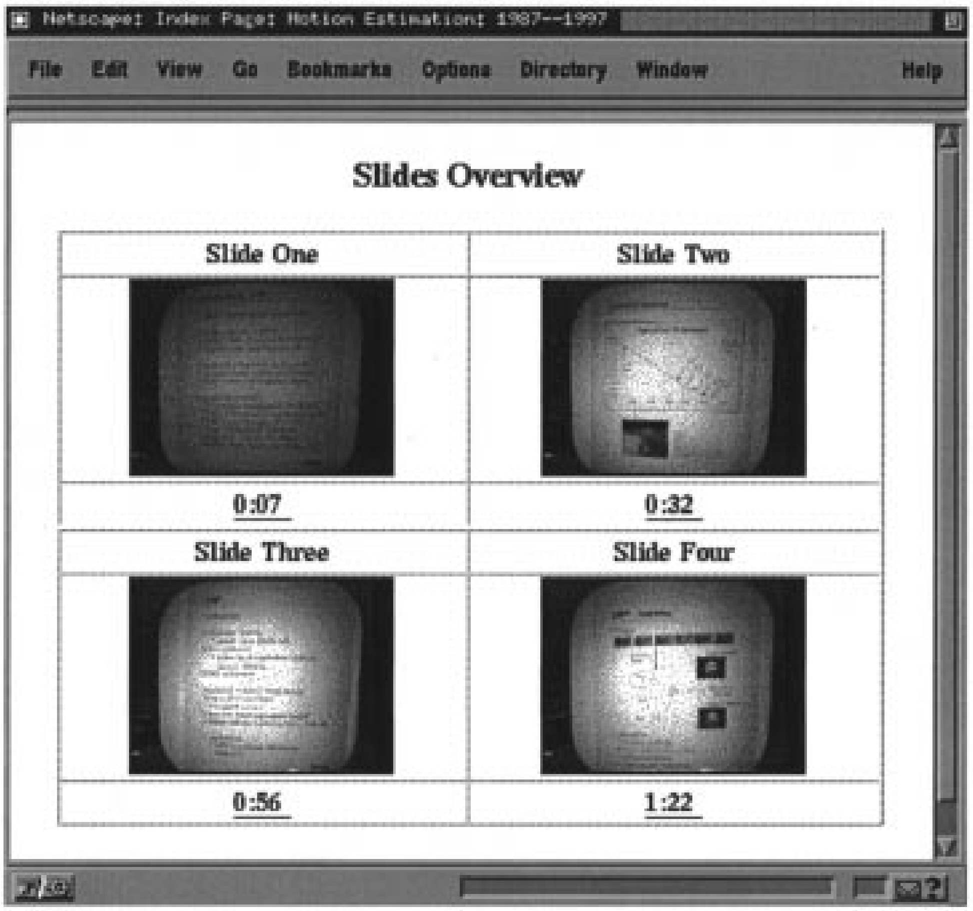
(e) (f) (g) (h)

Fig. 14. Responses with wrong position (a)–(c) and the missed gestures (d)–(h).

Fig. 13 shows all correctly recognized pointing gestures for each slide. No gestures occurred in two of the slides (six and eight). Fig. 14(a)–(c) shows the three responses with incorrect pointing position, while Fig. 14(d)–(h) shows the five gestures that were not detected by the system. The first three missed gestures happen in the first slide, where the key frame includes an arm which is visually similar to the gestures.

C. Discussion

We have shown results obtained with real presentations which demonstrate the performance of the system. The system



Slides Overview

Slide One	Slide Two
0:07	0:32
Slide Three	**Slide Four**
0:56	1:22

Fig. 15. Web-based video browsing application: the talk overview page.

is still experimental, and is not in wide use. The experiments, however, indicate that the system is robust enough to cope with the kinds of image quality available from a videotaped talk, and can cope with much of the complexity common in technical talks.

Our system is applied to a strictly defined domain, where the camera is focused on the projection screen. General scene breaks that occur in television sequences do not occur in our domain, and many of the techniques developed for extracting key frames based on shot detection will not reliably detect the change of slides. As people begin to make more “multimedia” presentations (e.g., using “Powerpoint”), we may find that our key-frame detection method will need to be generalized to cope with more arbitrary video sequences.

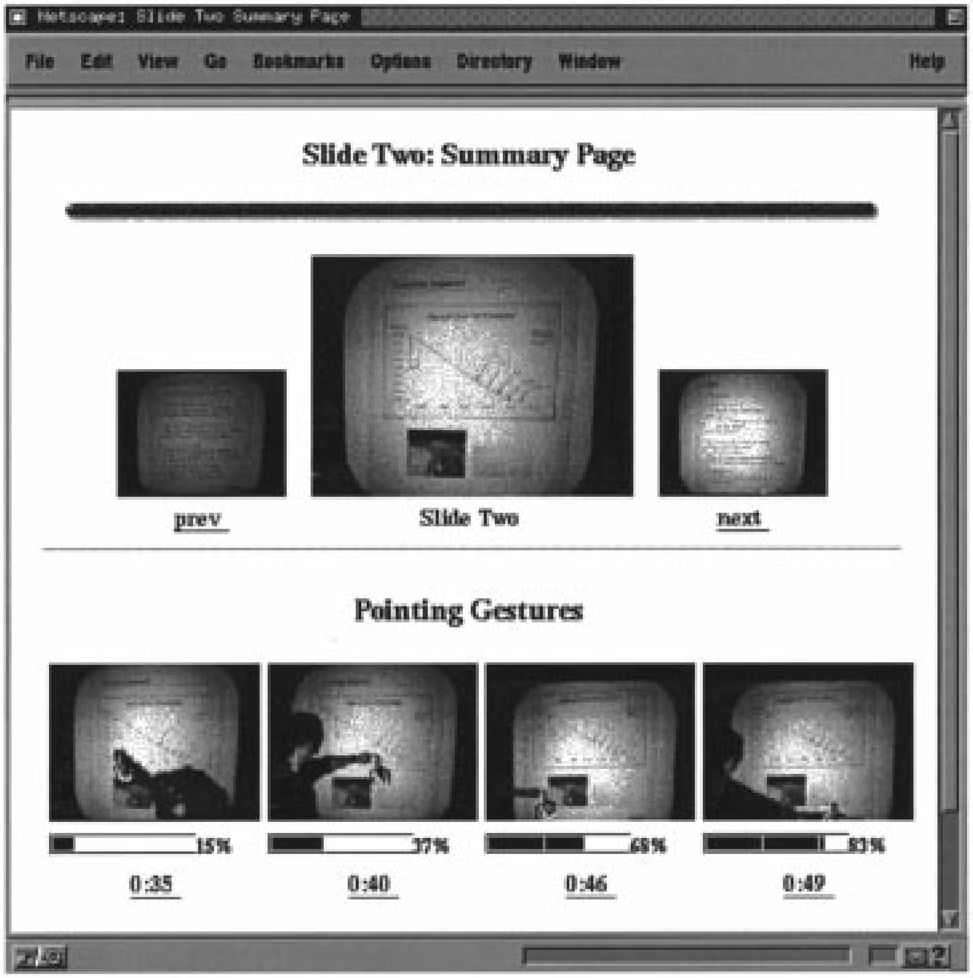
Experiments reported here were performed on an Intel Pentium II computer with a 233 MHz processor. It takes approximately 2 h and 15 min to process the 9 min sequence. According to these data, it would take 4–5 h to process a 30 min talk on a 333 MHz Pentium II. There are techniques that can speed up the performance, such as the real-time tracking system described in [10]. Compared with other motion-based shot-detection techniques [23], [30], our method recovers the dominant motion between frames with high accuracy; this is necessary for the stabilization process. A stabilized image sequence is a critical for the recovery of accurate key frames and the detection of gestures. Off-line processing, however, is acceptable for our proposed video summarization application. Real-time performance is only required for the Web-based interface (Figs. 15 and 16) and an interactive video browsing service.

Currently, our system only models pointing gestures. We would like to be able to detect when the speaker reveals text by moving the occluding surface. This is a distinctive action as the occluded material appears along the trailing edge of the occluding surface. Additionally, writing actions are typically detected as waving gestures. More sophisticated temporal models of gestures will be needed to distinguish between events such as pointing and writing (see, for example, [4], [10]).

Finally, a broader user study is called for. A wider sample of presentations will likely reveal distinct presentation types with their own affordances suggesting ways of modifying and expanding the gestures that need to be recognized. Additionally, we have currently not undertaken a user study to evaluate the user interface design.

VIII. Conclusion

We have described a fully automatic system that can robustly detect key frames in a videotaped presentation. The method is robust with respect to slide motions, occlusions, and gestures. It also provides an annotation of the slides that indicates where the speaker is pointing. This automatic



Slide Two: Summary Page

prev

Slide Two

next

Pointing Gestures

15%

0:35

37%

0:40

68%

0:46

83%

0:49

Fig. 16. First slide page.

video annotation and analysis system will help the user access presentation videos intelligently.

In future work, we would like to match the low-resolution slide template image with a stored Postscript/Powerpoint file of the slides. This would provide an automatic correspondence between the low-resolution video and an electronic version of the talk, and would allow a user to view or print specific slides at high resolution. We also leave the detection of revealing and writing affordances for future work. Finally, our current system uses only visual information for segmentation, annotation, and indexing. Other modalities such as audio may provide additional useful information about the structure of the content of the presentation (e.g., detecting questions from the audience or coarse speech understanding for text-based search).

ACKNOWLEDGMENT

The authors thank D. Fleet for his comments on this work and on the manuscript. They are also grateful to H. Haussecker for allowing them to record his talk on “tensor-based image sequence processing techniques for the study of dynamical processes” [11]. Finally, they thank the anonymous reviewers for their helpful suggestions.

REFERENCES

[1] E. Andre, G. Gergog, and T. Rist, “On the simultaneous interpretation of real world image sequences and their natural language descriptions,” in *European Conf. Artificial Intell.* Aug. 1988, pp. 449–454.

[2] H. Aoki, S. Shimotsuji, and O. Hori, “A shot classification method of selecting effective key-frames for video browsing,” in *ACM Multimedia 96*, 1996, pp. 1–10.

[3] M. J. Black and P. Anandan, “The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields,” *Comput. Vision, Graphics, Image Processing: Image Understanding*, vol. 63, pp. 75–104, Jan. 1996.

[4] M. J. Black and A. D. Jepson, “Recognizing temporal trajectories using the condensation algorithm,” in *Int. Conf. Automat. Face and Gesture Recognition*, Nara, Japan, Apr. 1998, pp. 16–21.

[5] A. F. Bobick and A. D. Wilson, “A state-based technique for the summarization and recognition of gesture,” in *Proc. IEEE Int. Conf. Comput. Vision*, June 1995, pp. 382–388.

[6] M. Brand, “Understanding manipulation in video,” in *Int. Conf. Automat. Face and Gesture Recognition*, 1996, pp. 94–99.

[7] L. Chen and P. Faudemay, “Multi-criteria video segmentation for TV news,” in *IEEE 1st Workshop Multimedia Signal Processing*, 1997, pp. 319–323.

[8] J. J. Gibson, *The Ecological Approach to Visual Perception.* Boston, MA: Houghton Mifflin, 1979.

[9] S. S. Intille and A. F. Bobick, “Closed-world tracking,” in *Proc. IEEE Int. Conf. Comput. Vision*, June 1995, pp. 672–678.

[10] M. Isard and A. Blake, “A mixed-state condensation tracker with automatic model-switching,” in *Proc. Int. Conf. Comput. Vision*, Mumbai, India, Jan. 1998, pp. 107–112.

[11] B. Jahne and H. Haussecker, “Study of dynamical processes with tensor-based spatiotemporal image processing techniques,” in *5th European Conf. Comput. Vision (ECCV’98)*, Freiburg, Germany, June 1998.

[12] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: Active contour models,” in *Proc. IEEE Int. Conf. Comput. Vision*, 1987, pp. 259–268.

[13] H. Kollnig, H. H. Nagel, and M. Otte, “Association of motion verbs with vehicle movements extracted from dense optical flow fields,” in *Proc. European Conf. Comput. Vision*, vol. II, 1994, pp. 338–347.

[14] R. Lienhart, S. Pfeiffer, and W. Effelsberg, “Video abstracting,” *Commun. ACM*, vol. 40, pp. 55–62, Dec. 1997.

[15] R. Mann, A. Jepson, and J. M. Siskind, “Computational perception of scene dynamics,” in B. Buxton and R. Cipolla, Eds., *Proc. European Conf. Comput. Vision*, vol. 1064 of LNCS Series, 1996, pp. 528–539.

[16] S. Minneman, S. Harrison, B. Janssen, G. Kurtenbach, T. P. Moran, I. Smith, and W. van Melle, “A confederation of tools for capturing and accessing collaborative activity,” in *Proc. Multimedia’95*, San Francisco, CA, Nov. 1995.

[17] K. Otsuji and Y. Tonomura, “Projection-detecting filter for video cut detection,” *Multimedia Syst.*, vol. 1, pp. 205–210, 1994.

[18] G. Retz-schmidt, “Recognizing intentions in the domain of soccer games,” in *European Conf. Artificial Intell.*, Aug. 1988, pp. 455–457.

[19] J. M. Siskind and Q. Morris, “A maximum-likelihood approach to visual event classification,” in B. Buxton and R. Cipolla, Eds., *Proc. European Conf. Comput. Vision*, vol. 1064 of LNCS Series, 1996, pp. 347–360.

[20] M. A. Smith and T. Kanade, “Video skimming and characterization through the combination of image and language understanding techniques,” in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognition*, 1997, pp. 775–781.

[21] V. V. Vinod and H. Murase, “Video shot analysis using efficient multiple object tracking,” in *IEEE Int. Conf. Multimedia Computing and Syst.’97*, 1997, pp. 501–508.

[22] J. W. A. Wang and E. H. Adelson, “Layered representation for motion analysis,” in *IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognition*, June 1993, pp. 361–366.

[23] B. Yeo and B. Liu, “Rapid scene analysis on compressed video,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, pp. 533–544, Dec. 1997.

[24] M. M. Yeung, B. Yeo, W. Wolf, and B. Liu, “Video browsing using clustering and scene transitions on compressed sequences,” in *Multimedia Computing and Networking*, vol. SPIE-2417, 1995, pp. 399–413.

[25] M. M. Yeung, B. Yeo, and B. Liu, “Extracting story units from long programs for video browsing and navigation,” in *Proc. IEEE Multimedia Computing & Syst.*, 1996, pp. 296–305.

[26] M. M. Yeung and B. Yeo, “Video visualization for compact presentation and fast browsing of pictorial content,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 771–785, Oct. 1997.

[27] K. D. Yow, B. Yeo, M. M. Yeung, and B. Liu, “Analysis and presentation of soccer highlights from digital video,” in *2nd Asian Conf. Comput. Vision*, Dec. 1995, pp. 499–503.

[28] R. Zabih, J. Miller, and K. Mai, “Video browsing using edges and motion,” in *IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognition*, 1996, pp. 439–446.

[29] H. Zhang, A. Kankanhalli, and S. W. Smoliar, “Automatic partition of full-motion video,” *Multimedia Syst.*, vol. 1, pp. 10–28, 1993.

[30] H. Zhang, C. Y. Low, Y. Gong, and S. W. Smoliar, “Video parsing using compressed data,” in *Proc. IS&T/SPIE Conf. Image and Video Processing II*, 1994, pp. 142–149.

[31] H. Zhang, Y. Gong, S. W. Smoliar, and S. Y. Tan, “Automatic parsing of news video,” in *Proc. Int. Conf. Multimedia Computing Syst.*, 1994, pp. 45–54.

[32] D. Zhong, H. Zhang, and S. Chang, “Clustering methods for video browsing and annotation,” in *Storage and Retrieval for Still Image and Video Databases IV*, vol. SPIE-2670, 1996, pp. 239–246.

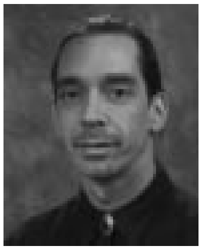
Shanon X. Ju received the B.Sc. degree in 1992 from Tsinghua University, China, and the M.S. degree in 1994 from the University of Toronto.

She is currently a Ph.D. candidate in the Department of Computer Science, University of Toronto. During the summers of 1995 and 1996, she worked as a Research Intern at Xerox Palo Alto Research Center. Her research interests include motion estimation and its applications in the video databases, rigid and nonrigid object tracking, and recognition of activities.

Michael J. Black (S’90–M’92) received the B.Sc. degree in honors computer science from the University of British Columbia in 1985, the M.S. degree in computer science from Stanford University in 1989, and the Ph.D. degree in computer science from Yale University in 1992.

Between 1990 and 1992, he was a Visiting Researcher at the NASA Ames Research Center, Aerospace Human Factors Research Division. From 1992 to 1993, he was an Assistant Professor in the Department of Computer Science at the University of Toronto. In 1993, he joined the Xerox Palo Alto Research Center where he is currently the Head of the Image Understanding Research Group.

Dr. Black serves an Associate Editor of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE. At the 1991 IEEE Conference on Computer Vision and Pattern Recognition, he received the IEEE Computer Society Outstanding Paper Award for his work with P. Anandan on robust optical flow estimation. His research interests include optical flow estimation, human motion recovery, gesture recognition, and robust statistics.



Scott Minneman received the B.A. degree in architecture and the B.S. and M.S. degrees in mechanical engineering from MIT. His doctoral work was done at the Center for Design Research, Stanford University.

He works in the Research on Experimental Documents Group at the Xerox Palo Alto Research Center. At PARC since 1987, he studies design practice and the potential of using video/audio/computing systems to support design work, focusing on synchronous and asynchronous uses of networked digital multimedia and shared drawing systems.

Dr. Minneman received the Best Paper Award at the 1995 ACM Conference on Multimedia Systems for his work with colleagues at PARC on a suite of tools for capturing and accessing collaborative activity.

Don Kimber received the B.E. degree from Stevens Institute of Technology in 1980, and the M.S. degree in computer and information science from the University of California, Santa Cruz, in 1988.

In 1995, he received the Ph.D. degree in E.E. from Stanford with a dissertation on geometric methods for modeling dynamical systems, with application to speech recognition.

He has been a member of the Collaborative Systems area at Xerox PARC since 1995. His work has included stochastic modeling for speech recognition, word spotting, speaker recognition, and audio segmentation. He has also worked on system architectures for the capture and indexing of meetings and talks. His current research interests are in understanding and supporting collaboration.