Robust Spatio-temporal Matching of Electronic Slides to Presentation Videos

Quanfu Fan, Kobus Barnard, Arnon Amir, and Alon Efrat

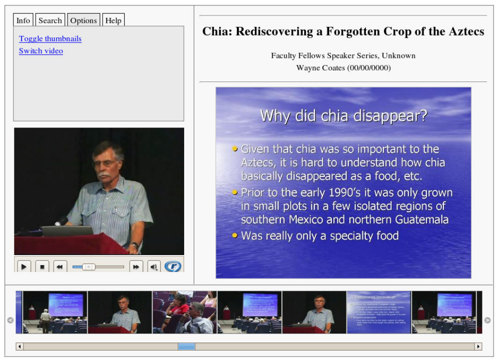
***Abstract*—We describe a robust and efficient method for automatically matching and time-aligning electronic slides to videos of corresponding presentations. Matching electronic slides to videos provides new methods for indexing, searching, and browsing videos in distance learning applications. However, robust automatic matching is challenging due to varied frame composition, slide distortion, camera movement, low-quality video capture, and arbitrary slides sequence. Our fully-automatic approach combines image-based matching of slide to video frames with a temporal model for slide changes and camera events. To address these challenges we begin by extracting scale invariant feature transformation (SIFT) keypoints from both slides and video frames, and matching them subject to a consistent projective transformation (homography) by using random sample consensus (RANSAC). We use the initial set of matches to construct a background model and a binary classifier for separating video frames showing slides from those without. We then introduce a new matching scheme for exploiting less distinctive SIFT keypoints that enables us to tackle more difficult images. Finally we improve upon the matching based on visual information by using estimated matching probabilities as part of a hidden Markov model (HMM) that integrates temporal information and detected camera operations. Detailed quantitative experiments characterize each part of our approach and demonstrate an average accuracy of over 95% in 13 presentation videos.**

***Index Terms*—Matching slides to video frames, Video indexing and browsing, Distance learning, SIFT keypoints, Homography constraint.**

I. Introduction

We describe a robust and efficient method for automatically aligning electronic slides to videos of corresponding presentations. For each video frame we identify the corresponding electronic slide (temporal alignment) and determine the geometric transformation between them (spatial alignment). This matching enables novel approaches to searching, browsing, and viewing on-line presentations in the context of distance learning and corporate training. Presentation slides provide *semantic* handles for segmenting, linking and manipulating the instructional content. Text in slides can be reliably extracted from the presentation file and used to index the video. Slide changes provide semantically meaningful segmentation of the video. Further, slide images can be back-projected into the video to improve its quality or implement compression. Enabling these benefits by automated robust matching of slides to video is the main contribution of this work.

An example application is shown in Figure 1 where we use the method developed in this paper to synchronize the video



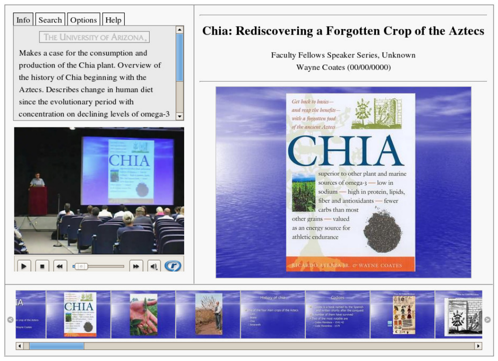
Chia: Rediscovering a Forgotten Crop of the Aztecs

Faculty Fellows Speaker Series, Unknown
Wayne Coates (00/00/0000)

Why did chia disappear?

- Given that chia was so important to the Aztecs, it is hard to understand how chia basically disappeared as a food, etc.
- Prior to the early 1990's it was only grown in small plots in a few isolated regions of southern Mexico and northern Guatemala
- Was really only a specialty food

(a)



Makes a case for the consumption and production of the Chia plant. Overview of the history of Chia beginning with the Aztecs. Describes change in human diet since the evolutionary period with concentration on declining levels of omega-3

Chia: Rediscovering a Forgotten Crop of the Aztecs

Faculty Fellows Speaker Series, Unknown
Wayne Coates (00/00/0000)

Get back to basics— and reap the benefits— with a forgotten food of the ancient Aztecs

CHIA

superior to other plant and marine sources of omega-3 — low in sodium — high in protein, lipids, fiber and antioxidants — fewer carbs than most other grains — valued as an energy source for athletic endurance

RICARDO AYERZA JR. & WAYNE COATES

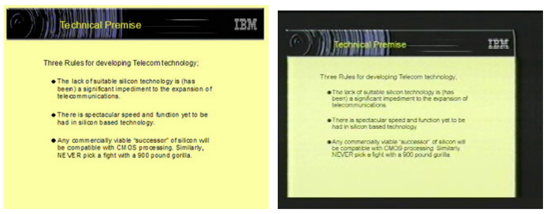
(b)

Fig. 1. Two ways of browsing videos [1]: a) by keyframes and b) by slides. Notice the difference at the bottom thumbnails slider. Slide changes provide a semantic video segmentation and are therefore more desirable for video browsing than keyframes extracted by shot boundary detection.

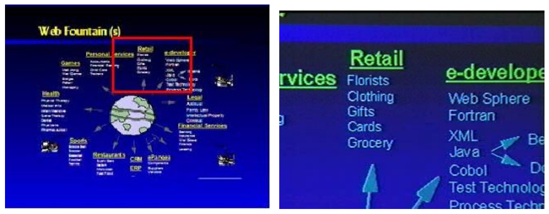
with the original slides shown side by side. Further, instead of using thumbnails corresponding to shot boundaries for video browsing (1a), we use thumbnails of slide images (1b). The later choice makes navigation to the topics of interest faster and more convenient. While such browsing is already provided in various systems (e.g., [2], [3], [4], [5]), none of these systems have an automated way to time-align and link slides to video content. Instead they use various hardware-based solutions, specially instrumented lecturer's computers, and/or manual annotation methods to align video with slides.

A key difficulty for automating the alignment is the varied appearance of slides in the video. Lecture video may be captured by amateurs or experts, and may use one or more pan-tilt-zoom (PTZ) cameras targeting the presenter, the screen, or the audience at different times. Hence we consider three kinds of video frames depending on whether the projection screen is shown in the frame and how large it is (Figure 2). A *full-screen* frame typically shows only the presentation screen, but we include all frames where more than 50% of the image

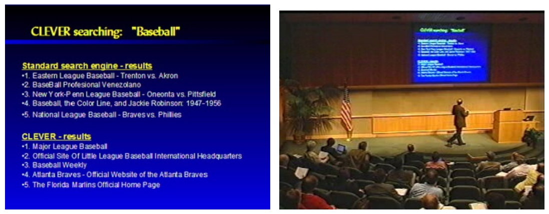
Q. Fan is with IBM T. J. Watson Research Center. K. Barnard and A. Efrat are with the U. of Arizona. A. Amir is with IBM Almaden Research Center.



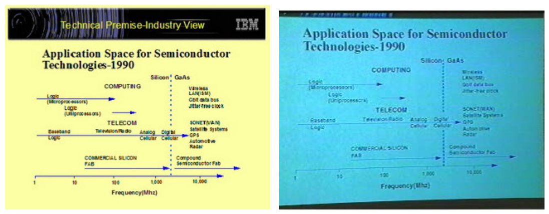
(a) *Full-screen* frame



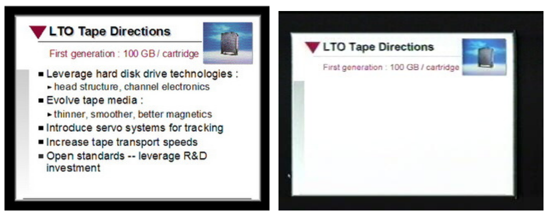
(b) Zoom-in, *full-screen* frame


CLEVER searching: "Baseball"
Standard search engine - results
CLEVER - results

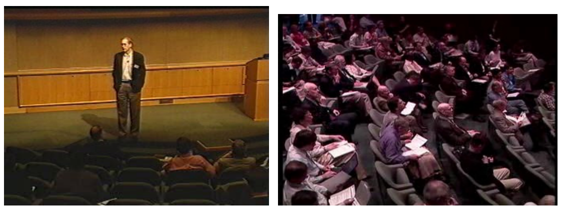
(c) *small-screen* frame


Technical Premise-Industry View
Application Space for Semiconductor Technologies-1990

(d) *Full-screen* frames with dramatic color change and small zoom-in



(e) Animated slide, *full-screen* frame



(f) *No-screen* frame (presenter, audience)

Fig. 2. *Sample video frames captured in presentations. Pairs (a), (b), (d) and (e) are* full-screen *frames, (c) is a* small-screen *frame, and (f) shows two* no-screen *frames. Each pair (a-e) shows a slide image (left) and a corresponding video frame (right). In (b), the red box marks the slide region captured by the video frame.*

contains slide content. In *small-screen* frames, the screen is present, but occupies at most 50% of the image area. These are usually wide field of view shots of the presenter along with the projection screen. Finally, in some shots, when the camera pans to the presenter or audience, the screen is left outside of the camera field of view, resulting in *no-screen* frames with no slide content. A robust system must handle all types of frames.

The situation becomes even more challenging when the slides are washed out by strong ambient illumination, occluded by the presenter, or distorted in color due to incorrect camera white balance. These complications make robust matching difficult, especially when we need to distinguish between nearly identical slides which are common in presentations.

Previous work on using image features has not produced robust results for the unconstrained matching task described above. One reason is that slide localization (done first) and recognition have been addressed separately. However, capturing systems using PTZ cameras often produce frames where part of the screen is left out, making naive slide localization difficult. Further, slide identification ambiguities that are difficult to resolve using image information can potentially be addressed using temporal information, but these two have not been integrated.

Our approach first matches video frames to slides using image features and then improves on those matches using temporal modeling of slide changes and camera operations. The image based matching process consists of three phases. In the first phase, a *global keypoint matching* process is applied to a set of keyframes. This yields some successful matches for "easier" keyframes. In the second phase, these matches are exploited to automatically build a classifier for pruning *no-screen* frames, and to build a background model for matching more difficult *small-screen frames* where the screen is less than half of the frame area. In phase 3, the slide-to-frame transformations found in the previous phases are used to efficiently and accurately match the remaining frames using a new matching scheme we call *local keypoint matching*. Finally, the resulting matches, expressed as probabilities, are used in a hidden Markov model (HMM) that integrates the spatial matching with a temporal model of slide transition probabilities and camera operation probabilities. This improves the system by disambiguating matches to similar slides.

Key contributions. To the best of our knowledge, this work is the first to: 1) simultaneously identify and register slides in unconstrained video; 2) automatically acquire a scene background model using identified slides in *small-screen* frames and apply that model to improve identification of more difficult slides; 3) improve matching speed and accuracy using efficient *local* homography search that exploits multiple homographies computed from other frames; 4) further improve slide to frame matching with temporal information that accounts for slide changes, camera operations, and camera switches. The measured accuracy (over 95%) under a wide range of conditions goes substantively beyond what has been previously reported, and opens up a number of possibilities for improving access to instructional content.

II. Related Work

The great potential of e-learning has inspired considerable research in providing effective tools to structure and navigate instructional content [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16]. The specific task of synchronizing slides with video was first addressed by manually editing time stamps (e.g., the BMRC Lecture Browser [3]). Subsequently, the Classroom 2000 Project [17] introduced hardware to record time stamps during the presentation. Their ClassPad provided easy browsing and annotation of slides in the classroom for both teachers and students. Today, several commercial e-learning systems can deliver synchronized multimedia presentations over the Internet where slides are displayed side by side with the video (e.g., the Mediasite system [4] and Microsoft Producer [18]). However, current content authoring systems do not provide automatic off-line slide-to-video matching. Rather, synchronization is achieved either by recording time stamps at slide changes on the computer [2], or by manually inserting time stamps (e.g., [5]). Another approach is directly recording the video signal from the video projector (e.g., [19]). Slide change is detected based on the difference between two subsequent frames, and a new slide is stored only when the difference exceeds a pre-specified threshold.

All these approaches require dedicated hardware and/or software systems that are engaged in advance of the presentation. This is a major limitation when the goal is to provide access to the vast stores of on-line video from disparate sources on a variety of generic platforms. To address this limitation, a few automatic slide-to-video matching methods have been proposed. Most of them can be regarded as a two-step matching process: slide extraction followed by slide identification. Mukhopadhyay et al. [20] developed a system for structuring multimedia content in which slides and audio information are automatically synchronized with video. The system uses a fixed camera where the projective transformation is predetermined by the four corner points of the projector during the system installation. Syeda-Mahmood [21] proposed a method to locate slides in videos using an illumination-invariant descriptor built upon the background color of slides. The method is capable of detecting slides appearing anywhere in the frames. The spatial layout geometry of the detected slide regions is successively used to recognize slides. Liu et al. [22] developed an algorithm to identify slides in videos by matching those of all possible slide pairs. The method detects the quadrilaterals in the frames to extract the slide regions. Erol et al. [23] proposed a method to link slide images taken by camera during presentations to the source files that generate the slides. In their method, an image is first classified as one of four different types, and depending on the image type, the slide is identified by applying one or several combinations of methods, including edge histogram matching, line profile matching, string matching and layout matching. Behera et al. [24] developed a method to spot slide change events based on analyzing the visual stability of a sequence of frames. However, no further slide recognition done.

Other approaches for slides to video alignment include the use of Optical Character Recognition (OCR) to extract text from video frames and match it with slides text [25], and aligning words from speech, extracted using speech recognition, to slide text [26]. Neither of these methods can handle slides without text, or distinguish between slides containing the same text but different graphics. Both are sensitive to errors made by their respective recognition engines. However, they suggest promising additional source of information that could be integrated into the system proposed here.

III. Spatio-temporal Matching of Slides to Presentation Videos — System Overview

Our method for matching slide images to frames sampled from a presentation video integrates *spatial matching* [27] and *temporal modeling* [28] (see Figure 3). Spatial matching links frames to slides based on visual features. To achieve robustness, speed, and accuracy, we use a three phase algorithm (Alg. 1) that progressively improves the matching based on what is learned in previous phases. We apply the first two phases to a small, sparse set of video frames ("keyframes") and leverage what is learned to efficiently match (in phase 3) the much larger target set of input frames ("sampled frames") which are sampled from the video at a fixed rate — we use one frame per second in this work. To construct the keyframe set we use the CueVideo shot boundary detection algorithm [29] to divide the video into segments, and, whenever needed, divide these further so that each shot is at most a minute long. We use the middle frames of the shots as keyframes.

In phase 1 we apply *global keypoint matching* (§IV-F) to quickly find matches for only keyframes that are easily matched to slides. These matches provide the information necessary for phase 2, which has three steps, each again is involving only keyframes.

The first step of phase 2 uses the matched *small-screen* keyframes to build an unsupervised scene background model (step 2.1). As shown later, slide regions in *small-screen* frames can be detected and spatially aligned via background matching. Next we use the matched keyframes to train a binary classifier to separate frames with slide content from *no-screen* frames without any slide content. We apply the classifier to the unmatched keyframes and prune the *no-screen* keyframes (step 2.2). We then try to match the keyframes newly classified as having slide content in a second run of RANSAC with more iterations. Further, for slides where background matching

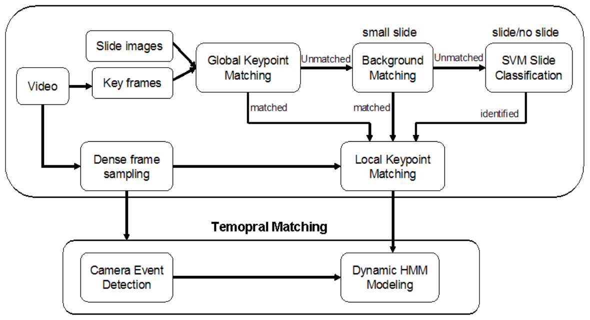


Fig. 3. *Flowchart of the spatio-temporal slides to video matching process.*

provides the slide area within the frame, we ignore keypoints outside the slide area while matching (step 2.3).

Finally, in phase 3, we process the dense set of sampled frames, using the classifier to prune out the *no-screen* frames and applying the new *local keypoint matching* algorithm to the rest. This method leverages the slide-frame transformations available from the previous phases to restrict the spatial scope of the matching thereby increasing scalability and enabling us to exploit less distinctive keypoints to improve accuracy.

The whole pipeline is efficient. Phase 1 and phase 2 only involve keyframes which are small in number. Phase 3, although applied to the much larger set of sampled frames, benefits from a fast local matching that leverages information learned from the previous phases.

Temporal modeling (§V). The spatial matching process provides frame-slide match hypothesis and associated probabilities. The probabilities for all frames are combined into a hidden Markov model (HMM) that distinguishes camera changes from slide changes and incorporates slides sequencing and jumping probabilities, set from data. This helps disambiguate weak matches and matches to similar slides by using the knowledge that slides are usually presented in order. The temporal model also integrates the detected camera operation information (§V-B), exploiting the observation that certain camera operations, such as switching between cameras, or zooming in, rarely occur simultaneously with a slide change. Hence, a slide presented before a cut (an abrupt camera change) is expected to remain displayed when the camera changes. On the other hand, the camera is very likely to remain fixed during slide changes. Notice that a camera change is artistic, triggered by the video producer, while a slide change is semantic, triggered by the presenter. In the video, both appear as abrupt change of frame content. To achieve robust matching it is important to distinguish between these two changes.

IV. Spatial Matching of Frames to Slides

Previous algorithms for matching video frames to slides have attempted to first locate the slide region in the frame, and then match it to a slide. However, reliably locating slides in frames is difficult, especially under difficult conditions or when slide boundaries are not visible which is common when PTZ cameras are used. We propose an alternative approach that simultaneously identifies the slide and solves for the geometric mapping from the slide to the video frame. Here we use random sample consensus (RANSAC) to link scale invariant feature transform (SIFT) keypoints under the constraint of a consistent mapping, as has been done in a number of computer vision applications (e.g., [30], [31], [32], [33]).

Algorithm 1 *Spatial Matching of Frames to Slides*

```
Require: A set of slides and the video of the corresponding presentation
    Initialize: Extract keyframes using shot boundary detection and create a set of frames
    sampled at a desired fix rate.
    Phase 1: Match keyframes to slides using RANSAC with global NN search.
    Phase 2 (step 2.1): Create a scene background model and use it to determine slide
    regions in small-screen frames.
    Phase 2 (step 2.2): Create an SVM classifier for no-screen frames, and use it to prune
    no-screen frames from the keyframes.
    Phase 2 (step 2.3): Run RANSAC again with a larger number of iterations on the
    remaining unmatched keyframes. For small-screen frames where the slide region within
    the frame is found by background matching, keypoints outside the slide are ignored
    in the matching.
    Phase 3. Match all sampled frames identified as having slide content with the same
    SVM classifier using RANSAC with local NN search.
    return the corresponding slides matched to any of the input frames.
```

A. SIFT Keypoints

SIFT keypoints [34] are points of local gray-level maxima and minima that are detected from a set of difference-of-Gaussian images in scale space. Points with low contrast and strong single-direction edge response are eliminated to increase stability and distinctiveness. Each keypoint is associated with a scale, an orientation, and a descriptor (128-element vector) that represents the statistics of local gradients in a patch centered at that keypoint. SIFT keypoints are scale and rotation invariant, and partially robust to change in illumination and camera viewpoint. These properties make them particularly suitable for handling camera movement in the slide matching task. In this work, we generated keypoints from images off-line by using a publicly available SIFT keypoint detector [35].

Figure 4 shows the SIFT keypoints detected in a slide (4a) and a corresponding frame (4b). As seen in the figure, heavily textured regions produce substantially more keypoints than color-homogeneous regions. Fortunately for our application, text on slides generally yields many distinctive keypoints.

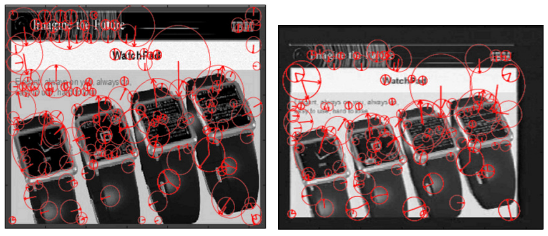
B. Keypoint Matching

The feature similarity between two keypoints can be measured by the Euclidean distance of the keypoint descriptor vectors described above. The match (or correspondence) of a keypoint in one image is defined as the nearest neighbor of all the keypoints from another image. To accept a point match as valid, one could check that the distance is less than a pre-specified global threshold. However, it is generally difficult to find a good threshold as some features are much more distinctive than others. Instead, we use the following approach suggested by Lowe [34] to discard false matches based on the two nearest neighbors

Let P_1 and P_2 be two keypoints from images 1 and 2 respectively, where P_1 is the nearest neighbor of P_2 in the feature space. Then, P_1 is considered a match to P_2 only if

$$\frac{d(f_{P_1}, f_{P_2})}{d(f_{P_1'}, f_{P_2})} \leq \tau \tag{1}$$

where $d(.,.)$ denotes the Euclidean distance between two descriptor vectors and f_P is the descriptor vector of a keypoint P. P_1' is the second nearest keypoint of P_2 in image 1. This ratio of the distances to the nearest and second nearest neighbors is more reliable than an absolute threshold since a false match is more likely to have a number of other matches within close distances in the feature space. As studied by Lowe [34], an appropriate choice of the threshold τ rejects the majority of false keypoint matches while still retaining most of the correct matches. Figure 4c shows an example of the keypoint matches achieved by this ratio measure. Since this scheme matches keypoints by searching nearest neighbors


(a) Slide SIFT keypoints (b) Frame SIFT keypoints

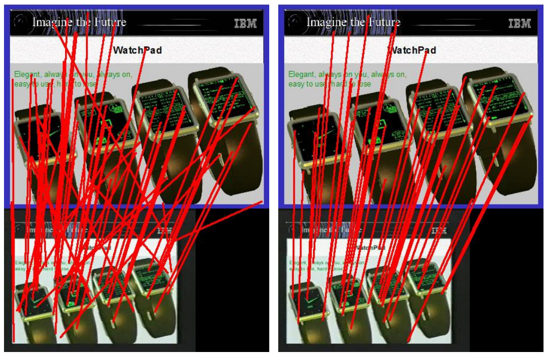

(c) Putative matching (d) Correct matching

Fig. 4. *Keypoint matching. The top two images show keypoints detected in a slide image (a) and a frame image (b). Each circle indicates the location (center) and scale (radius) of a keypoint feature detected. An arrow is attached to each circle to show the associated orientation. The bottom left image (c) shows matches proposed based on nearest neighbors alone. The bottom right image (d) shows correct matches that share a homography from slide to frame.*

(NNs) throughout an entire image, we call it *global NN search* to distinguish it from the *local NN search* we propose in §IV-I.

Keypoint matching using nearest neighbor search in a high dimensional feature space may be time consuming. When the dimension is higher than $10-20$, spatial data structures such as kD-trees do not perform any better than linear search (depending on the number of points [36]). Therefore, other applications [30], [37] use approximate algorithms such as BBF [38] and LHS [39], combined with dimensionality-reduction techniques. However, in our three-phase matching algorithm, linear search performance is sufficient for the limited number of keyframes used in *global keypoint matching* (§IV-F). Further, we greatly reduce the overall matching cost through a new matching scheme, *local keypoint matching* (§IV-I), that is used for the majority of frame-slide matching.

C. Estimating the Homography Between Two Images

A $2D$ *homography* is an invertible projective transformation that maps points from one plane to another plan. The points in a slide image are mapped to the corresponding points on a projector screen by a homography, and the later are mapped to their projection in a video frame by another homography. Hence we can use the transitivity property of homographies to define a homography that maps points directly from the slide to the video frame. When the camera moves, the homography changes. Yet the mapping of corresponding projected slide points from one video frame to another video frame is, by transitivity, also a homography. Moreover, as long as the camera only zooms and rotates around its focal point without translation, non-planar scene background points from one frame to another are also connected via a homography. Hence homographies model many of the relations between slides, their projections on screen and the screen and scene projection into video frames. The more general case of matching non-planar background sections under general camera motion is less relevant to our application.

We represent image points using homogeneous coordinates. Given two homogeneous points $\mathbf{x} = [u, v, w]^T$ and $\mathbf{x}' = [u', v', w']^T$ linked by a homography $\mathcal{H}$, the mapping between $\mathbf{x}$ and $\mathbf{x}'$ is

$$\mathbf{x}' = \mathcal{H}\mathbf{x}, \tag{2}$$

where $\mathcal{H}$ is a 3×3 matrix. We use the normalized direct linear transformation (DLT) [40] to compute $\mathcal{H}$ from a set of four hypothesized keypoint matches.

D. Fitting Constrained Homography using RANSAC

Some initial matches from the nearest neighbors search are false matches (*outliers*) due to the ambiguity of SIFT features. However, the correct matches (*inliers*) should agree on the homography model that links the slide image and its corresponding frame in the video. To remove outliers we use **RAN**dom **SA**mple **C**onsensus (RANSAC) [41]. RANSAC repeatedly chooses a random selection of the minimum number of data points required to fit the model. The fitted model is then evaluated based on the number of other data points that agree on the hypothesis within a specified error (or the fitting error evaluated over a required number of inliers). After a pre-specified number of iterations, the algorithm returns the hypothesis that generates the maximum number of supporting data points (or minimum fitting error) as the best model.

We consider a match between keypoints $(\mathbf{x}, \mathbf{x}')$ to be an *outlier* if the symmetric transfer error $\mathbf{e} \geq \epsilon$ where

$$\mathbf{e} = \sqrt{\mathbf{d}(\mathbf{x}, \mathcal{H}^{-1}\mathbf{x}')^2 + \mathbf{d}(\mathbf{x}', \mathcal{H}\mathbf{x})^2} \quad , \tag{3}$$

and $d(\cdot, \cdot)$ is the Euclidean distance between two homogeneous points projected to image coordinates. We empirically set $\epsilon = 3\sqrt{2}$, which is reached, for example, if the transformed distances in both directions are 3 pixels.

E. Pruning Unlikely Homography Candidates

Slides captured by a PTZ camera may be scaled, rotated and even distorted, but in a typical slide capturing situation where the intention is that the lectures will be viewed "as is," these transformations are not expected to be too extreme. If they were, the video would likely be considered unacceptable. We exploit this observation to prune proposed homographies that are unlikely to be correct. This improves performance and helps reduce false positive matches to frames where there are no slides (e.g., only audience or presenter). Of course, blindly using this heuristic in situations where there might be unusual camera placements could lead to worse performance.

To test if a proposed homography, $\mathcal{H}$, should be pruned, we first approximate it by the affine transformation

$$\mathcal{A} = \begin{pmatrix} \frac{h_{11}}{h_{33}} & \frac{h_{12}}{h_{33}} & \frac{h_{13}}{h_{33}} \\ \frac{h_{21}}{h_{33}} & \frac{h_{22}}{h_{33}} & \frac{h_{23}}{h_{33}} \\ 0 & 0 & 1 \end{pmatrix} \quad . \tag{4}$$

By dropping the third row and column in $\mathcal{A}$, we obtain a 2×2 matrix $\mathcal{A}_{2\times 2}$, which we further decompose as

$$\mathcal{A}_{2\times 2} = RSU = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} 1 & u \\ 0 & 1 \end{bmatrix} \quad . \quad (5)$$

Here R is a rotation matrix, S is a scaling matrix, and U is a shearing matrix. We constrain $\mathcal{A}$ so that:

- $|\theta| \leq 20^\circ$
- $\frac{1}{10} \leq s_x \leq 10$; $\frac{1}{10} \leq s_y \leq 10$;$\frac{2}{3} \leq \frac{s_x}{s_y} \leq \frac{3}{2}$
- $|u| \leq 0.2$

In other words, the slide projection is a transformation with rotation no larger than 20° and scaling between 1 and 10, and shearing no larger than 0.2. During the iterative process of RANSAC, for any proposed $\mathcal{H}$, if the corresponding $\mathcal{A}$ fails any of the above conditions, then $\mathcal{H}$ is discarded.

F. Global Keypoint Matching

The *global keypoint matching* phase may receive frames of any type, with or without a slide. Hence slide matching is globally applied in the entire frame area. To match a frame to slides, we start with the slide that has the greatest number of initial keypoint matches to the frame (based on (1)), and use RANSAC to prune the matches based on the homography constraint. Let k be the number of matches that remain. By observation, $k \geq 6$ can almost guarantee a correct slide transformation if it is constrained as described in §IV-D. We consider a match valid if $k \geq \mathcal{K}$, where we set $\mathcal{K}$ to 10 in our experiments for a high level of confidence. If the match is not valid, we move on to the slide that has the second most number of initial keypoint matches to the frame (and so on). If no valid match can be found, the frame is left unmatched for now. Here we only need enough matches to support the classifier training and background modeling of phase 2.

G. Exploiting the Background By Homography Consistency

Homography Consistency. Let f_i be a frame linked to slide, **s**, by homography $\mathcal{H}_i^s$. Also, let f_j be an unmatched frame of **s** from the first phase (see Figure 5). The transformation $\mathcal{H}_{i,j}^f$ between f_i and f_j is a homography, which can be found using SIFT keypoint matching. Further, denote the proposed homography between **s** and f_j by $\mathcal{H}_j^s$. A slide keypoint x_k is mapped to a keypoint y_{ik} on f_i and a keypoint y_{jk} on f_j. Accordingly, y_{ik} should correspond to y_{jk} in the matching of f_i and f_j. Thus

$$y_{ik} = \mathcal{H}_i^s x_k \ , \ y_{jk} = \mathcal{H}_j^s x_k \ , \ and \ y_{jk} = \mathcal{H}_{i,j}^f y_{ik} \quad . \quad (6)$$

Since x_k is arbitrary, we can merging these equations to yield the algebraic relationship between $\mathcal{H}_i^s$, $\mathcal{H}_j^s$ and $\mathcal{H}_{i,j}^f$ as

$$\mathcal{H}_j^s = \mathcal{H}_{i,j}^f \mathcal{H}_i^s \quad . \quad (7)$$

We refer to this simple property as *homography consistency*. It provides an indirect way to compute the slide homography for a frame. Instead of matching it to a slide directly, one can match it to another frame of the same slide whose slide homography is already identified.

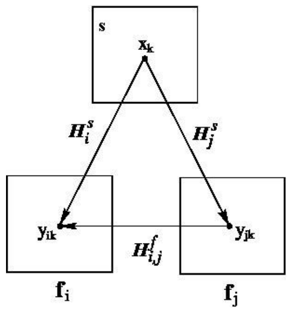


Fig. 5. *A slide* **s** *and two views (frames)* f_i *and* f_j *of it. The slide homographies mapping* **s** *to* f_i *and* f_j *are* $\mathcal{H}_i^s$ *and* $\mathcal{H}_j^s$*, respectively. The homography mapping* f_i *to* f_j *is* $\mathcal{H}_{i,j}^f$*. As shown in the text,* $\mathcal{H}_i^s$*,* $\mathcal{H}_j^s$ *and* $\mathcal{H}_{i,j}^f$*, are related by* $\mathcal{H}_j^s = \mathcal{H}_{i,j}^f \mathcal{H}_i^s$*.*

Background Matching. *Small-screen* frames are captured by a wide-view camera and usually share substantial background such as the podium, audience and projector (see Figure 2c). Since scene background usually yields plenty of keypoints,the transformation between two *small-screen* frames can be robustly established by background matching. In such cases, even if the two frames do not show the same slide, *homography consistency* still holds under the assumption that the camera only rotates about its optical center.

One reason that *small-screen* frames are hard to match is that there are many outliers from the background in the initial matching. With the slide homographies identified with background matching, we first locate the slide area and eliminate the disturbance from background, and thus gain a better chance to find a correct match with RANSAC. More importantly, the identified homographies are used for *local keypoint matching* (see §IV-I) that enable more powerful methods for disambiguating *small-screen* frames. If the assumption that the camera only rotates around its optical center does not hold strictly, the homography $\mathcal{H}_j$ would only approximate the transformation, but generally it will still be accurate enough for the subsequent *local keypoint matching*.

Background matching requires at least one identified *small-screen* frame in the *global keypoint matching* phase. The background model is nominally the keypoints found in the collection of background regions in the identified *small-screen* frames. In the case where multiple *small-screen* frames are available, an alternative is to reconstruct a panoramic view of the background scene of the classroom by stitching the frames together using a method proposed by Brown [30] that is available through the $Autostich^{TM}$ software [42], and extracting the keypoints from the mosaic. This has the advantage that we only store one keypoint for each element of the background instead of potentially several noisy duplicates from multiple frames. Further, because the keypoints are determined from multiple overlapping frames, they are potentially more accurate. We experimented with this alternative background model in the case of one of our test videos that had many *small-screen* frames (CONV1). Figure 6 illustrates the background matching process using that video as an example.

H. Detecting no-screen Frames

Detecting *no-screen* frames is not trivial because of the large visual variations in different types of frames. We separate frames without slides from others using what we learned from the first phase. The underlying assumption is that slide images from a given presentation will be far away in an appropriate

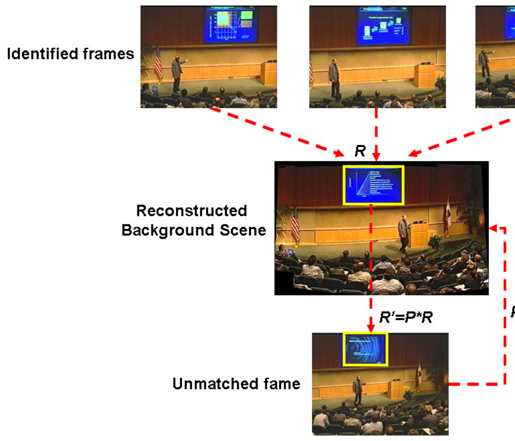


Fig. 6. *Background matching. small-screen frames usually share static objects in the background scene such as the flag and podium indicated here. This provides a way to detect small-screen frames by matching background between frames, as illustrated here. The middle image is a reconstructed background scene by stitching multiple identified small-screen frames.*

feature space from *no-screen* frames such as those of the presenter and audience.

For image classification features we use the color coherence vector (CCV), a color histogram that incorporates spatial information [43]. The CCV of an image, I, is a vector of pairs (α_i, β_i), which stores the number of coherent versus incoherent pixels with each discretized color. The CCV distance between two images I and I' [43] is defined as

$$d_G(I, I') = \sum_{i=1}^{n} |\alpha_i - \alpha'_i| + |\beta_i - \beta'_i| \quad , \tag{8}$$

where n is the number of discretized colors.

We use the slides identified in the first phase as positive training samples. Negative samples are selected among the frames that have large CCV distance from the positive samples. More specifically, the training data set is constructed as follows. Let $\mathcal{F}_l$ and $\mathcal{F}_s$ be the set of *full-screen* and *small-screen* frames detected in the first phase respectively. We set the positive samples $\mathcal{D}^+ = \mathcal{S} \cup \mathcal{F}_l \cup \mathcal{F}_s$ where $\mathcal{S}$ is the set of original slides. $\mathcal{F}_l$ augments $\mathcal{S}$ to compensate for color changes of slides in the video. The negative samples $\mathcal{D}^-$ are the k farthest *no-screen* frames away from $\mathcal{D}^+$ in the feature space. Formally, the distance of a frame f_i to $\mathcal{D}^+$ is

$$d_{\mathcal{D}^+}(f_i) = \min_{I_j \in \mathcal{D}^+} d_G(f_i, I_j) \quad . \tag{9}$$

We set k by $k = min(U, |\mathcal{D}^+|)$ where U is the number of keyframes remaining unmatched. We then use libSVM [44] to train a linear support vector machine (SVM) and classify frames unmatched in the first phase as either a frame with a slide or as a *no-screen* frame which can be excluded from further processing.

I. Local Keypoint Matching

Low resolution capture often leads to very few distinctive keypoints, especially in small or blurry slides. Here *global keypoint matching* (§IV-B) is likely to fail as most keypoints would be rejected in the initial matching due to the second NN criterion (Equation 1). Increasing the threshold τ does not help much as the majority of the newly found matches would be outliers.

To address this issue, we propose a new method to match up less distinctive keypoints locally in the frames using estimates of the homographies. The key observation is that a homography estimate dramatically constrains where we expect the match to be, and hence we only need to consider the few candidates that are near the expected position. This both speeds up the matching and increases the chance of finding matches for frames captured in challenging conditions (see §VI-C).

The effectiveness of the method relies on combining information over multiple video frames. For example, background matching over sequential frames allows us to provide homography estimates for *small-screen* frames that we have yet to match reliably. Further, homography consistency makes use of frame-to-frame matching which can sometimes be done reliably even when slide matching is not robust. Details for the two key parts of local keypoint matching follow.

1) Computing Initial Slide Homographies: Let $\{f_1, f_2, \ldots, f_n\}$ be a sequence of sampled frames from a video shot and let f_k be the keyframe with known slide homography, $\mathcal{H}^s_k$. We then apply keypoint matching to match every pair of consecutive frames in the sequence to obtain a set of homographies $\{\mathcal{H}^f_{1,2}, \mathcal{H}^f_{2,3}, \ldots, \mathcal{H}^f_{n-1,n}\}$ where $\mathcal{H}^f_{i,i+1}$ is the homography between f_i and f_{i+1}. These homographies can be found relatively quickly because frames within shots are visually similar.

The homography $\mathcal{H}^f_{i,j}$ between any two frames f_i and f_j, as illustrated in Figure 7, can be expressed as a product of a series of homographies by

$$\mathcal{H}^f_{i,j} = \begin{cases} \mathcal{H}^f_{i,i+1}\mathcal{H}^f_{i+1,i+2}\ldots\mathcal{H}^f_{j-1,j} & i < j \\ (\mathcal{H}^f_{j,j+1}\mathcal{H}^f_{j+1,j+2}\ldots\mathcal{H}^f_{i-1,i})^{-1} & i > j \end{cases} , \tag{10}$$

where the length of the series is $|i - j|$. Using *homography consistency*, the homography $\mathcal{H}^s_i$ mapping the slide to frame f_i now can be easily computed from the known $\mathcal{H}^s_k$ of the keyframe by $\mathcal{H}^s_i = \mathcal{H}^f_{i,j}\mathcal{H}^s_k$. While such a product will accumulate errors, this is not a problem, because we only need an initial approximation. Finally, in the case that a slide change occurs in the video shot, the homography series likely breaks due to no matching available at the slide change point. Here we set the missing homography to the identity matrix $\mathcal{I}$ as the camera tends to be fixed during slide changes.

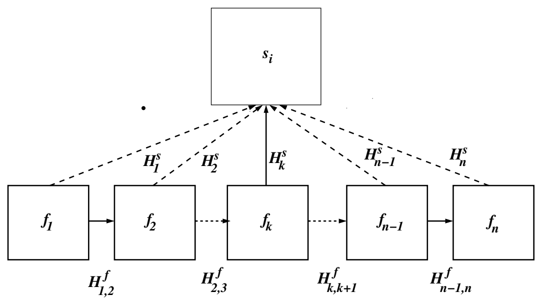


Fig. 7. *Inferring slide homographies. In a sequence of frames sharing the same slide, the slide homography of any frame can be inferred from one known slide homography in the sequence.*

2) Local Nearest Neighbor Search: For a keypoint P in f_i, let $P_{\mathcal{H}}$ be the projection of P in s_j according to $\mathcal{H}$. The *local NN* of P is a keypoint in s_j that is located around $P_{\mathcal{H}}$ within an image distance threshold r that satisfies (1), and is closest to P in feature space. To retrieve the local NN of P, we perform a range query centered at $P_{\mathcal{H}}$ in the 2D image space, and then search the NN of P in the feature space from the query results. Since the $2D$ range query is fast and returns only a few candidates to be evaluated using the 128-dimensional feature space, *local NN search* is significantly faster than global NN search. In addition, the geometric constraint from $\mathcal{H}$ rejects many outliers and yields a better initial matching, which in turn helps RANSAC achieve a solution that otherwise cannot be obtained by *global keypoint matching*. We set the radius of the $2D$ range query empirically, finding that any radius of $20 - 40$ pixels works well on our data. We used 40 for all experiments.

V. Temporal Modeling of Slide Change in Presentation Videos

The three-phase spatial matching approach has demonstrated high accuracy (over 90%) on our data. However, extremely similar slides are difficult to disambiguate using visual features alone. In addition, low-quality or *small-screen* frames can have multiple weak slide matches, or even no match, also making them difficult to identify using only appearance information. We improve the matching accuracy in these challenging situations using the temporal order of slides and cues from camera operations.

Slides generally advance sequentially according to their order in the presentation file, although the sequence is sometimes interrupted by shifting to the previous slide or jumping to an arbitrary slide. This notion of slide change can be well captured by a Hidden Markov Model (*HMM*) [45].

Camera operations also yield useful cues on slide change. For example, there usually is no slide change when the camera is zooming. Similarly, it is more likely that the camera will remain fixed when there is a slide change. In §V-B, we incorporate these hints on slide change from camera operations into the HMM to help further resolve ambiguity.

A. Modeling Slide Events By HMMs

We model slide change by a HMM where the states are the slide numbers. The matching problem then can be mathematically expressed as follows. Given a frame sequence $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$ and a set of temporally ordered slide images $\mathcal{S}_0 = \{s_1, s_2, \ldots, s_m\}$, find an optimal slide sequence $\hat{\mathcal{S}}$ that maximizes the following conditional probability

$$p(\mathcal{S}|\mathcal{F}, \mathcal{M}) = \pi(s_1) \prod_{i=2}^{n} \mathcal{A}(s_i|s_{i-1}) \mathcal{B}(f_j|s_i) \qquad , \qquad (11)$$

where $\mathcal{M} = (\pi, \mathcal{A}, \mathcal{B})$ is the model, with $\mathcal{A}$ being the slide transition probability, $\mathcal{B}$ being the probability of observing a frame given some slide, and π being the prior that is considered uniform in our case. We use the well-known Viterbi algorithm [46] to solve (11) to find an optimal slide sequence for the frame sequence.

Data	$\mathbf{T_{k<-1}}$	$\mathbf{T_{-1}}$	$\mathbf{T_0}$	$\mathbf{T_1}$	$\mathbf{T_{k>1}}$	$\mathbf{T_{NS/SN}}$	$\mathbf{T_{NN}}$
CONF1	N/A	0.03%	48.50%	1.17%	0.11%	1.63%	48.54%
CONF2	N/A	0.06%	45.26%	1.00%	.01%	1.06%	52.53%
UNIV	N/A	0.10%	50.31%	1.74%	N/A	1.59%	46.24%

TABLE I
FREQUENCIES OF THE VARIOUS SLIDE-CHANGE EVENTS IN OUR DATA. BECAUSE WE DENSELY SAMPLE THE VIDEO AT ONE FRAME PER SECOND, THE TWO MOST COMMON "EVENTS" ARE IN FACT HAVING NO CHANGE: STAYING ON THE CURRENT ($\mathbf{T_0}$) OR CONTINUE WITH *no-screen* FRAMES ($\mathbf{T_{NN}}$). AS EXPECTED, THE NEXT MOST FREQUENT EVENT IS ADVANCEMENT BY A SINGLE SLIDE ($\mathbf{T_1}$). GOING BACKWARDS IS GENERALLY RARE.

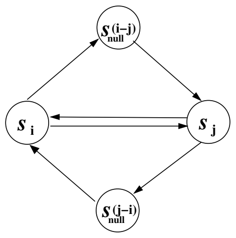


Fig. 8. Modeling *no-screen* frames. There is one *"null"* node between each ordered pair of slide nodes

1) Slide Transition: Consider a pair of consecutive frames f_j, f_{j+1}. We define the following *slide events* depending on the slides shown in the frames. When frames f_j, f_{j+1} show slides s_i, s_{i+m}, respectively, we denote the event by $\mathbf{T_m}$ (an m-slides jump). The sign of m indicates the direction of the jump and $m = 0$ means that the same slide is shown in both frames. When both frames contain no slide we write $\mathbf{T_{NN}}$, and when only one of them contains a slide we write $\mathbf{T_{NS}}$ or $\mathbf{T_{SN}}$ depending on whether the slide is seen before the frame or after it.

We consider any slide transition $\mathbf{T_m}$ to be *stateless*, that is, the probability of $\mathbf{T_m}$ only depends on m. For example, transition from slide 2 to slide 3 has equal likelihood as a change from slide 7 to 8. This assumption reduces the Cartesian product of slide transitions into a linear sized set of slide events, one for each m sized jump. Notationally, we have $\mathcal{A}(s_i|s_j) = p(T_m)$ where $m = s_i - s_j$ is the slide event.

We estimate the stateless slide transition probabilities from held out data by counting the frequency of the slide events. Because the data is limited, we enforce smoothness using three Poisson distributions as follows:

$$p(T_m) = \begin{cases} \eta P(-m, \lambda_1) & m < 0 \\ \eta P(0, \lambda_1) + (1-\eta) P(0, \lambda_2) & m = 0 \\ (1-\eta) P(m, \lambda_2) & m > 0 \end{cases} \quad , \qquad (12)$$

where $P(x, \lambda) = \frac{e^{-\lambda}\lambda^x}{x!}$ is the Poisson distribution, and η is the frequency that slide changes go backwards set using held out data. The values of λ_1 and λ_2 were fit from a held-out portion of the ground truth data using the Poissonness plot [47].

Table I shows the distribution of slide transitions in our data. Because of the high frame sampling rate we used for the final pass (1 frame per second), transitions to the same state dominate. Also, as expected, slides tend to change forward much more frequently than backward. Thus η, $\lambda 1$, and $\lambda 2$ in Eq. 12 are all small numbers.

2) Modeling Frames Without Slide: To deal with *no-screen* frames we add a *"null"* node between each ordered pair of slides (Figure 8). The *"null"* node keeps track of the slide last shown which is necessary because when the video goes back to showing slide frames, the distribution over slides depends on the last slide that was shown. Note that only a linear number of *"null"* nodes needs to be maintained in the implementation due to the stateless assumption of the slide transition.

3) Slide Observation Probability: We denote $\mathcal{B}(f_j|s_i)$ as the probability of seeing a frame f_j given a slide s_i. Let k_{ij} be the number of keypoint matches between f_j and s_i after being thresholded by $\mathcal{K}$ (the minimum number to be considered as a valid slide match). We set

$$\mathcal{B}(f_j|s_i) = \frac{w_{ij}k_{ij}}{\sum_\ell w_{i\ell}k_{i\ell}} \quad , \tag{13}$$

where w_{ij} is a weight to favor a slide with more keypoint matches to the frame. The weight w_{ij} can be interpreted as is the confidence level of matching from s_i to f_j, which we set to $P(\mathcal{X}=1|\mathcal{H}_{ij})$ ($\mathcal{X}$ is defined below), as used by Brown and Lowe [30]. To derive P, let n_k be the number of keypoints in the slide area and n_i be the number of inliers compatible with the homography $\mathcal{H}$ solved by RANSAC. We define an event whether a slide is the correct match of a frame as a random variable $\mathcal{X}$ ($\mathcal{X} = \{0, 1\}$). We assume that keypoint matches are independent Bernoulli, and thus the number of inliers is binomially distributed:

$$\begin{array}{l} P(\mathcal{H}|\mathcal{X}=0) = B(n_i, n_k, p_0) \\ P(\mathcal{H}|\mathcal{X}=1) = B(n_i, n_k, p_1) \end{array} \quad , \tag{14}$$

where p_0 is the probability of a keypoint match being an inlier when a slide match is incorrect and p_1 is the probability of a keypoint match being an inlier when a slide match is correct.

Using Bayes' rule and assuming $P(\mathcal{X}=0) = P(\mathcal{X}=1)$,

$$\begin{array}{rcl} P(\mathcal{X}=1|\mathcal{H}) & = & \frac{P(\mathcal{H}|\mathcal{X}=1)P(\mathcal{X}=1)}{P(\mathcal{H})} \\ & = & \frac{P(\mathcal{H}|\mathcal{X}=1)P(\mathcal{X}=1)}{P(\mathcal{H}|\mathcal{X}=1)p(\mathcal{X}=1)+P(\mathcal{H}|\mathcal{X}=0)P(\mathcal{X}=0)} \\ & = & \frac{B(n_i,n_k,p_1)}{B(n_i,n_k,p_1)+B(n_i,n_k,p_0)} \end{array} . \tag{15}$$

We empirically set $p_0 = 0.1$ and $p_1 = 0.6$. Finally, for s_{null} (i.e *no-screen* frames), we use

$$\mathcal{B}(f_j|s_{null}) = \begin{cases} 1.0 & \text{if } f_j \text{ is a } \textit{no-screen} \text{ frame} \\ 0.0 & \text{otherwise} \end{cases} . \tag{16}$$

B. Integrating Slide Events and Camera Events

The visual appearance of presentation videos is influenced by both presenter actions (e.g., slide changes, white board use, switching to browsing the web), and producer actions (e.g, switching cameras, zooming, or panning). Producer actions and presenter events are correlated, as the producer reacts to presentation. For example, we expect that slide to frame homographies will remain the same across slide changes and that slides will not change during camera changes. In what follows we show how the camera cues can be incorporated into the HMM to help further improve matching performance.

1) Camera Events: Camera events describe how the producer operates the cameras when capturing a presentation. Here, we define 8 types of camera events of interest: *zoom-in, zoom-out, pan-tilt, stay-fixed, slide-cut, slide-in, slide-out,* and *stay-out*. The first four events are the basic camera operations. *Zoom-in* magnifies the slide area significantly in the current frame with respect to the previous frame due the producer increases the focal length, typically with a camera zoom control. *Zoom-out* is defined inversely. *Pan-tilt* is camera panning or tilting while keeping the slide in view. *Stay-fixed* refers to continued capture of the slide area (either *small-screen* or *full-screen* frames) without movement. The next three are more related to the visual change of slides. *Slide-cut* is a special event in multi-camera capture systems that refers to switches between two cameras. *Slide-out* denotes camera movement away from the slide to capture the presenter/audience only. *Slide-in* denotes the opposite operation. *Stay-out* is the camera event between *slide-out* and *slide-in* when no slide is being captured. A camera in *stay-out* may zoom or move, but we do not further differentiate between these actions as they provide little information about slide change.

2) Detecting Camera Events: Much previous work has considered understanding camera motion as an optical flow problem, including detecting camera operations based on the motion vector field (see [48] for a review). Here we take a different approach that detects camera operations based on changes to slide position and size in the video. These changes directly correspond to camera operations such as zoom, pan and tilt, and can be easily computed from the homography between two consecutive frames.

Let $\{f_1, f_2, \ldots, f_n\}$ be the set of frames in a given shot and let R_i be the slide region in frame f_i. Then R_i can be expressed by the product of the homographies between consecutive frames as shown in §IV-I1. Specifically,

$$R_i = \mathcal{H}^f_{i-1,i}\mathcal{H}^f_{i-1,i-2}\ldots\mathcal{H}^f_{1,2}R_1 \quad , \tag{17}$$

where $\mathcal{H}^f_{i-1,i}$ is the homography between f_{i-1} and f_i. R_1 is the slide region of the first sampled frame in the shot and computed from the slide homography identified during *spatial matching*. Similar to what has been done in *local keypoint matching*, if $\mathcal{H}_i$ is not available due to slide change in the same shot, we set $\mathcal{H}_i$ to be the identity matrix. To capture slide movement, we denote the slide center of R_i by L_i, computed as the average of the 4 vertices of R_i.

We define two random variables for each pair of consecutive frames. The first one, d, is the Euclidean distance of the two slide centers, which indicates camera panning and tilting. The second one, r, is the ratio of the slide areas of the two consecutive frames related to camera zooming. We represent each of the four events (*zoom-in*, *zoom-out*, *pan-tilt* and *stay-fixed*) by a Gaussian distribution over $X = [d,\ r]$. Because the distribution is over changes between consecutive frames that are densely sampled (as opposed to the absolute final position), the unimodal assumption is effective.

The parameters of the distributions (μ_i, σ_i) were learned directly from a held-out portion of the ground truth data. We distinguish among these four events by selecting the most likely one under this model and uniform prior probability. The other events are detected by using the frame classification results. For example, a current *small-screen* frame and a following *full-screen* frame indicate a *slide-cut* event between two frames, while a current *full-screen* frame and a following *no-screen* frame yield a *slide-out* event.

Dataset	Video	Duration (min)	Full Screen	Small Screen	No Screen	Key Frames	PPT Slides
CONF1	#1	47	33	9	61	103	29
	#2	55	76	3	72	151	39
	#3	41	38	6	53	97	27
	#4	20	20	8	23	51	21
	#5	39	41	12	64	117	34
	#6	49	53	42	59	154	67
CONF2	#1	68	122	3	103	228	63
	#2	54	58	1	104	163	68
	#3	63	50	0	90	140	49
	#4	52	40	1	61	102	33
UNIV	#1	39	33	9	61	103	44
	#2	48	76	3	72	151	48
	#3	41	58	15	71	144	49

TABLE II
SUMMARY OF THE VIDEO DATA USED IN OUR EXPERIMENTS.

3) Modeling Slide and Camera Events By a Dynamic HMM: The dynamic model incorporates context dependent information from the camera events into the transition probabilities of the HMM [49]. For example, camera change is associated with a higher probability that there is no slide change. More specifically, we condition the state transition probability *p*(T_m) on the camera event c: *p*($T_m|c$). These probabilities are estimated from the held-out portion of the ground truth data. A trivial modification of the Viterbi algorithm, namely replacing *p*(T_m) by *p*($T_m|c$) at each time step based on c, is sufficient to find an optimal slide sequence.

Camera events refine slide transition probabilities in temporal modeling. For example, in our data, the learned *p*($T_0|c$) (statistical counting) is perfectly 1.0 when c is a *camera zoom, pan-tilt* or *slide-cut*. In our experiment, this refinement has shown helpful in keeping the slide alignment in place when there is a switch between two cameras, i.e *slide-cut* (Table VII). We expect more contributions from the camera cues on more difficult data shot by non-professional users where significant camera movements have been observed [50].

VI. Experimental Results

We constructed three data sets containing a total of 13 presentations (MPEG video and slide file). Six presentations were from a corporate conference (CONF1) [29] and four presentations were from a scientific conference (CONF2). Both of these data sets were captured using three PTZ cameras with live video editing, with one camera tracking the speaker, one camera covering the projection screen and is used to zoom in on the slides, and the third camera capturing the audience. Three more presentations were captured from a university seminar series (UNIV) (available online [51]) using two cameras, one giving *small-screen* views and the other capturing the audience. CONF1 is more complex than the other two as there are many camera switches in the data, and two videos have dramatic color change in the slides. The data is summarized in Table II.

A. Evaluation Methods

For evaluation purposes, we manually marked each sampled frame with ground-truth slide correspondence (1 to n if a slide is present, 0 otherwise) and frame type (i.e., *full-screen*, *small-screen* or *no-screen*). A few frames showing slides that were missing from the presentation (such as demos) were marked as "missing" and were excluded from the evaluation.

A frame is *correctly identified* if the slide number determined by the algorithm is the same as the one marked in the ground truth. We report errors in two ways. First, we report the **frame** identification error rate (**FER**) which is the ratio of the number of incorrectly identified frames to the total number of frames. Notationally,

$$\textbf{FER} = \frac{\text{\# of incorrectly identified frames}}{\text{\# of total frames}} \quad . \qquad (18)$$

This measure is biased towards slides that appeared for a longer time. Thus we also aggregate results based on video segments defined as a clip without slide changes or camera switches. The **segment** identification error rate (**SER**) is computed by

$$\textbf{SER} = \frac{\text{\# of incorrectly identified frames in the segment}}{\text{\# of total frames in the segment}} \quad . \qquad (19)$$

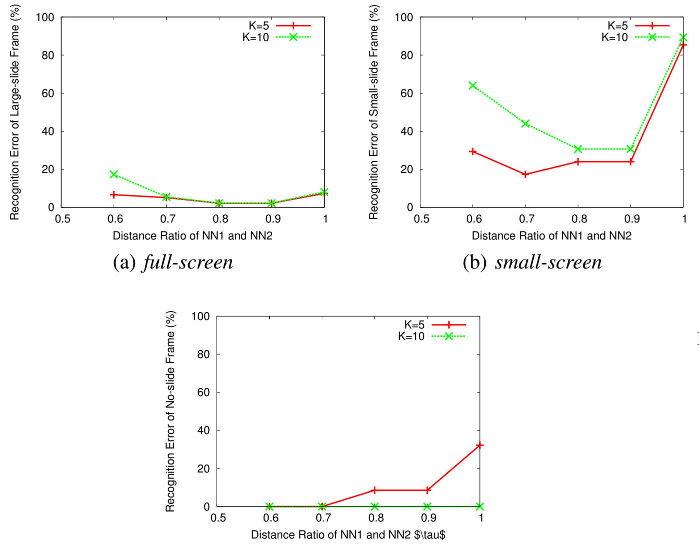
We compute an overall performance value by averaging **SER** over segments, ignoring those with less than 2 sampled frames.

B. Preliminary Experiments

1) Setting the number of RANSAC iterations: We first established suitable values for the number of RANSAC iterations by plotting the error rate against the number of iterations up to 1000 over 10 runs (plots omitted to save space). More iterations reduce the risk of not finding a good match but increases running time directly. For *global keypoint matching* we found that beyond 200 iterations there was little decrease in error, and we conservatively set the number of iterations to 500 for subsequent experiments. *Local keypoint matching* requires fewer iterations and reducing the number of iterations has a bigger impact on run time. Here we found that going beyond 100 iterations did not substantively reduce the error, and hence we used 100 iterations for *local keypoint matching* in our experiments.

2) Setting Matching Thresholds: The parameter τ determines the number of initial keypoint matches in RANSAC, and a larger τ accepts more keypoint matches. Figure 9 plots the slide recognition errors over τ for the three types of frames. As indicated by the figure, the recognition accuracy of *full-screen* frames is not affected much by the choice of τ when $\tau \geq 0.7$. However, the recognition accuracy of *small-screen* frames is quite sensitive to the value of τ, as indicated by the large errors that occur when τ is either too small or too large. This is mainly because the less distinctive keypoints on *small-screen* frames yield too few initial keypoint correspondences for a small τ and introduce too many outliers for a large τ. In addition, as shown in Figure 9c, when $K = 5$ we get a higher number false positives as τ increases. In summary, we found $K = 10$ and $\tau = 0.8$ to be good settings which we used for subsequent experiments (unless otherwise specified).

3) Frame Classification Results: The frame classification module excludes *no-screen* frames early in the matching process thus saving significant computational time. It also affects matching accuracy in two ways. First, an entire video



(a) *full-screen* (b) *small-screen*

(c) *no-screen*

Fig. 9. *Recognition error of different types of frames in the base keypoint matching algorithm: (a) full-screen; (b) small-screen and (c) no-screen. The results for two matching thresholds* 5 *and* 10 *are shown. The slide recognition performance in small-screen frames is sensitive to the value of* τ*, and is degraded dramatically when* $\tau > 0.9$*. A larger* τ *also tends to yield more false positives when a smaller threshold is used. In addition, as shown in (c), a small threshold cannot effectively cut off false positives as* τ *goes up, but a higher threshold such as* 10 *we used here, can effectively suppress all the false positives.*

Data	*#full-screen*	*#small-screen*	*#no-screen*	Total
CONF1	1/272	3/85	2/338	6/702 (0.9%)
CONF2	7/271	0/1	7/320	14/592 (2.3%)
UNIV	0/157	4/98	1/235	5/488 (1.0%)

TABLE III
NUMBER OF INCORRECTLY CLASSIFIED FRAME TYPES.

shot would be considered as *no-screen* if the keyframe of the shot was classified as *no-screen*. If the keyframe was misclassified, the slide in that video shot will not be identified. Second, correctly classified frames with matched slides have a chance to be identified by the temporal model.

Table III gives the classification errors over the three data sets. In general, our algorithm achieves high accuracy on all the videos. The scene background model is demonstrated robust in detecting *small-screen* frames in the videos. The SVM classifier performs very well on CONF1 and UNIV, but presents slightly higher false positives and false negatives on CONF2. A closer inspection of these errors reveals that some were due to slides containing embedded videos (in CONF2) and browser web pages (in CONF1).

C. Spatial Matching Results

We evaluated the accuracy of matching keyframes to slides for two algorithms. GLOB(500) is the basic *global keypoint matching* algorithm with 500 RANSAC iterations. GLOB(500)+LOC(100) is the improved three-phase matching algorithm, first using 500 RANSAC iterations in the *global keypoint matching* phase and then using 100 RANSAC iterations in any consequent local matching.

Table IV gives the matching error rates of the two algorithms broken down by frame type, as well as the overall frame

Data	Alg	full screen	small screen	no screen	Total misses	**FER**
CONF1	GLOB(500)	11	21	0	32±2	4.5%
	GLOB(500)+LOC(100)	11	12	0	23±2	**3.2%**
# frames		272	75	347	694	
CONF2	GLOB(500)	64	0	0	64±1	11.2%
	GLOB(500)+LOC(100)	52	0	0	52±2	**9.2%**
# frames		248	1	317	566	
UNIV	GLOB(500)	4	26	1	31±1	6.4%
	GLOB(500)+LOC(100)	5	16	1	22±1	**4.5%**
# frames		157	90	235	482	

TABLE IV
THE NUMBER OF FULL-SCREEN, SMALL-SCREEN, AND NO-SCREEN KEYFRAMES WITH MISIDENTIFIED SLIDES FOR THE THREE DATA SETS AVERAGED OVER 10 RUNS WITH DIFFERENT RANDOM SEEDS. ERROR ESTIMATES FOR THE NUMBER OF MISSES ARE COMPUTED USING THE VARIATION OVER THE RUNS. FER DENOTES FRAME IDENTIFICATION ERROR RATE.

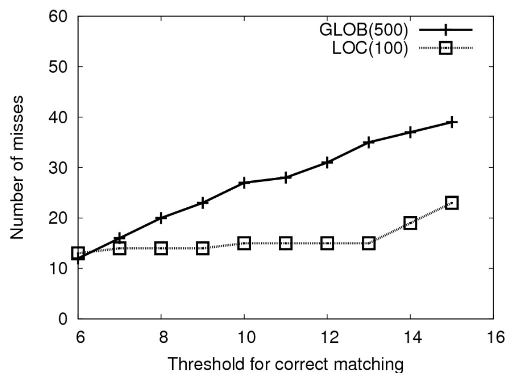


Fig. 10. *Algorithm robustness is demonstrated by repeating the experiments using different matching threshold* $\mathcal{K}$*, and measuring the number of misidentified small-screen frames. Local keypoint matching (LOC(100)) showed greater robustness than the global matching algorithm, GLOB(500).*

identification error rate defined in equation (18). We found that *local keypoint matching* can significantly improve the matching performance on the *small-screen* frames, supporting the notion that local NN search can identify more correct keypoint matches from non-distinctive keypoints.

The *full-screen* results on CONF2 are markedly worse than those on CONF1 and CONF3. Checking these presentations, we found that the higher errors on CONF2 can be attributed to the many frames with little slide content.

We further tested the robustness of different algorithms to the matching threshold $\mathcal{K}$. Figure 10 demonstrates how the number of misses goes up as $\mathcal{K}$ increases in the case of *small-screen* frames. As shown, *local keypoint matching* is almost insensitive to the choice of $\mathcal{K}$, hence providing higher matching confidence than the alternatives. Varying $\mathcal{K}$ has no measurable impact on the performance of matching *full-screen* frames where confidence levels are much higher, nor on *no-screen* frames classification where $\mathcal{K} = 6$ already yields nearly perfect results.

Running Time. Given that N_f frames sampled from a video are matched to N_s slides, the computational complexity of the *global keypoint matching* algorithm is $O(N_f\ N_s)$ as each frame needs to be compared to all slides. Similarly, the complexity of the three-phase matching algorithm is $O(N_k\ N_s + (N_f - N_n)\ N_s)$, without considering the negligible overhead from frame classification. Here N_k is the number of keyframes and N_n is the number of *no-screen* frames that

Alg.	GLOB(500)+LOC(100)				GLOB(500)
CPU time (min)	Global	BG	Local	Total	Total
CONF1	38.12	0.62	28.36	67.51	844.09
CONF2	48.04	0.77	31.19	80.00	720.83
UNIV	20.84	0.39	18.57	38.80	576.39

TABLE V
THE AVERAGE TIME (MINUTES) SPENT MATCHING ALL FRAMES OF THE CONF1, CONF2 AND UNIV DATA SETS. THE THREE-PHASE METHOD (GLOB(500)+LOC(100)) IS ABOUT 10 TIMES FASTER THAN THE BASIC METHOD (GLOB(500)).

are dropped for consideration in the *local keypoint matching*. Real world running time is also influenced by many other factors including the number of pre-specified iterations for RANSAC, how much texture there is in the images which leads to more keypoints, and the quality of the images which affect the distinctiveness of the keypoints.

Here we provide basic timing results using a Linux machine with a 3.0GHz Pentium processor and 1GB of memory. Table V gives the average matching time for the three data sets, again with the algorithms GLOB(500) and GLOB(500)+LOC(100), but now for all frames, not just keyframes, as this is where computational cost is an issue. Because the CPU time for matching is roughly proportional to both video length and number of slides, we normalize running time to that of a one-hour long video (scale by $60/L$, where L is time in minutes), and a presentation with 30 slides (scale by $30/N_s$).

Finding the SIFT keypoints of all the frames is overhead common to both algorithms which is not counted in the tabulated matching time results. It takes an average of 38 minutes to compute the SIFT features for the CONF1 videos normalized to 60 minutes. Other overhead, such as computing the CCV features and frame classification, are negligible.

The data presented in Table V shows that the three-phase slide matching algorithm (GLOB(500)+LOC(100)) improves upon the basic method (GLOB(500)) by an order of magnitude. For the basic method, the significant matching cost dominates the SIFT feature extraction overhead. By contrast, the three-phase keypoint matching algorithm is reasonably efficient, taking less than two hours to match a one-hour video to 30 slides. Here the time needed for each of two most expensive phases, phases 1 and 3, are both roughly comparable to the SIFT feature extraction overhead, which has now become one of the main bottlenecks.

D. Temporal Modeling Results

We compare the performance of three algorithms: the best performing algorithm evaluated above based on image information alone (GLOB(500)+LOC(100)), the standard HMM without camera (HMM), and the camera-event-based HMM (CHMM). The results from temporal modeling on the three data sets are presented in Tables VI and VII. Both HMMs substantially outperform the base matching algorithm, showing the advantage of using temporal information. In particular, keyframes not correctly matched or even unmatched during the spatial matching process will have a chance to get aligned with their corresponding slides by the HMM models. Figure 11 illustrates an example which was not identified by keypoint matching, but resolved successfully by the temporal models.

Data	Alg	full screen	small screen	no screen	Total misses	**FER**
	GLOB(500)+LOC(100)	178	173	1	352±6	2.3%
CONF1	HMM	143	109	64	316±30	2.1%
	CHMM	122	110	64	295±30	**1.9%**
# frames		6146	1540	7743	15429	
	GLOB(500)+LOC(100)	1009	11	1	1021±18	7.0%
CONF2	HMM	703	9	66	778±37	5.4%
	CHM	703	9	66	778±37	5.4%
# frames		6674	45	7813	14532	
	GLOB(500)+LOC(100)	101	142	38	281±16	3.5%
UNIV	HMM	49	78	44	171±19	2.2%
	CHMM	65	31	44	139±38	**1.8%**
# frames		3596	631	3732	7959	

TABLE VI
THE NUMBER OF MISIDENTIFIED FULL-SCREEN SMALL-SCREEN, AND NO-SCREEN FRAMES FOR THE THREE ALGORITHMS FOR EACH OF THE THREE DATA SETS, AVERAGED OVER 10 RUNS WITH DIFFERENT RANDOM SEEDS. ERROR ESTIMATES FOR THE NUMBER OF MISSES ARE COMPUTED USING THE VARIATION OVER THE RUNS. FER DENOTES FRAME IDENTIFICATION ERROR RATE.

Data	Alg	full screen	small screen	no screen	Total misses	**SER**
	GLOB(500)+LOC(100)	9.3	10.8	0.0	20.1±0.7	3.3%
CONF1	HMM	8.2	6.5	0.9	15.6±0.8	2.5%
	CHMM	8.1	6.4	0.9	15.4±0.7	2.5%
# segments		274	84	253	611	
	GLOB(500)+LOC(100)	39.3	0.0	0.1	39.4±1.1	10.4%
CONF2	HMM	28.9	0.0	1.3	30.2±1.5	7.9%
	CHMM	27.9	0.0	1.3	29.2±1.5	**7.7%**
# segments		223	1	157	381	
	GLOB(500)+LOC(100)	4.1	15.8	1.1	21.0±0.9	5.6%
UNIV	HMM	1.1	7.3	1.2	9.7±0.9	2.6%
	CHMM	1.7	3.7	1.2	6.6±1.2	**1.7%**
# segments		155	83	134	372	

TABLE VII
THE NUMBER OF MISIDENTIFIED FULL-SCREEN, SMALL-SCREEN, AND NO-SCREEN SEGMENTS FOR THE THREE ALGORITHMS AVERAGED OVER 10 RUNS WITH DIFFERENT RANDOM SEEDS. ERROR ESTIMATES FOR THE NUMBER OF MISSES ARE COMPUTED USING THE VARIATION OVER THE RUNS. SER DENOTES SEGMENT IDENTIFICATION ERROR RATE.

Comparing CHMM with HMM gives some mixed results on *small-screen* and *full-screen* frames (for *no-screen* they are equivalent), but overall, CHMM performs better, exceeding the HMM aggregated score for CONF1 and UNIV, and matching it for CONF2.

Table VIII shows the percentage of frames that failed in keypoint matching, but were successfully identified by the combined spatial and temporal models. When the number of keypoints matched are less than the cutoff required for spatial matching, integrating both spatial and temporal information still has a fair chance (44% when $K = 10$; 25% when $K = 5$) of guessing the correct slide.

Finally, in Table IX we break down the results in according to the slide events. The results show that the HMMs can model the sequential change of slide very well. There are limited examples of non-sequential change in our data (Table I), but as expected, errors here are less likely to be repaired.

VII. CONCLUSIONS

Our multi-phase matching approach achieves high accuracy on a number of videos with different styles and difficulties. Further, our results suggest that the temporal information and camera cues are very promising sources of information to disambiguate the occurrence and identity of slides in videos when

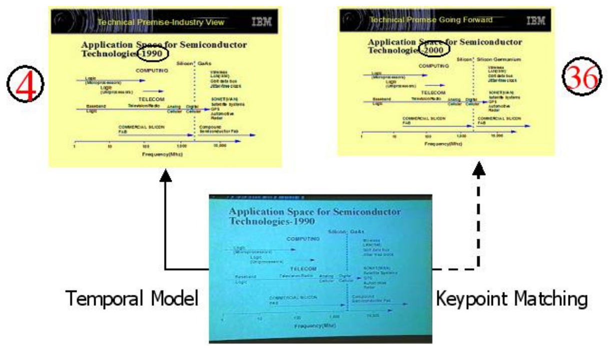

4
36
Technical Premise-Industry View
Technical Premise Going Forward
IBM
Application Space for Semiconductor Technologies 1990
Application Space for Semiconductor Technologies 2000
Temporal Model
Keypoint Matching

Fig. 11. *Two visually similar slides on the top, slide* 4 *and slide* 36. *The two ellipses in the figure indicate the slight difference between them. The bottom image is a captured frame of slide* 4. *The temporal model correctly matched the frame to slide* 36 *while the keypoint matching did not, showing that temporal information can improve the ability of the method to resolve ambiguity.*

Threshold	CONF1	CONF2	UNIV	Avg.
$\mathcal{K} = 5$	15/199 (7.5%)	126/365 (34.5%)	13/44 (29.6%)	154/608 (25.3%)
$\mathcal{K} = 10$	92/314 (29.3%)	359/741 (48.5%)	35/44 (79.6%)	486/1099 (44.2%)

TABLE VIII
PERCENTAGE OF THE FRAMES THAT WERE CORRECTLY IDENTIFIED BY THE TEMPORAL MODELS, BUT WHERE MISSED USING VISUAL INFORMATION ALONE. $\mathcal{K}$ IS THE THRESHOLD USED FOR VERIFYING CORRECT MATCHES DESCRIBED IN §IV-F.

conditions are challenging. The matching approach is sufficiently robust to support large scale alignment of presentation slides and videos. This is an important contribution because slides are very effective "semantic handles" for presentation videos, providing novel ways to index and browse content. In ongoing work, we are exploiting our matching process to build tools for making educational video more accessible [1].

ACKNOWLEDGMENTS

We thank many IBM colleagues who helped us collect data and generate the ground truth. We also thank the presenters who allowed us to use their presentations for this study. This work was supported in part by the Arizona Center for Information Science and Technology (ACIST) and by IBM. Alon Efrat was partially supported by NSF CAREER grant #CCF-0348000.

REFERENCES

[1] A. Amir, K. Barnard, Q. Efrat, Alon Fan, Y. Kharitonova, D. Mathis, R. Swaminathan, J. Torkkola, and A. Winslow, "The SLIC video browsing system," 2005. [Online]. Available: http://slic.cs.arizona.edu

[2] G. D. Abowd, "Classroom 2000: An experiment with the instrumentation of a living educational environment," in *IBM Systems Journal*, 1996.

[3] L. A. Rowe and J. M. Gonzelez, "BMRC lecture browsers," 1999. [Online]. Available: http://bmrc.berkekey.edu/frame/projects/lb/index.html

[4] MediaSite, "Mediasite," in *http://www.mediasite.com*, 2004.

[5] ViaScribe, "IBM ViaScribe," 2005. [Online]. Available: http://www-03.ibm.com/able/solution_offerings/ViaScribe.html

[6] T. Liu and R. Kender, J, "Lecture videos for e-learning: Current research and challenges," *Proceedings of IEEE Sixth International Symposium on Multimedia Software Engineering*, pp. 574–578, 2004.

[7] B. Erol and Y. Li, "An overview of technologies for e-meeting and e-lecture," in *IEEE International Conference on Multimedia and Expro (ICME)*, 2005, pp. 6–14.

[8] M. S. H. Wactlar, T. Kanade and S. Stevens, "Intelligent access to digital video: The informedia project," *IEEE Computer*, vol. 29, no. 5, pp. 46–52, May 1996.

Data	$\mathbf{T_{k<-1}}$	$\mathbf{T_{-1}}$	$\mathbf{T_0}$	$\mathbf{T_1}$	$\mathbf{T_{k>1}}$	$\mathbf{T_{NS}}$	$\mathbf{T_{NN}}$
CONF1	N/A	0.00	2.7%	9.9%	5.9%	0.1%	4.8%
CONF2	0.0%	77.8%	10.3%	22.2%	76.9%%	4.3%	16.3%
UNIV	N/A	100%	1.9%	10.1%	N/A	1.3%	6.8%

TABLE IX
THE ERROR PERCENTAGE OF THE MISIDENTIFIED SLIDE EVENTS FOR CHMM.

[9] D. E. Atkins, W. P. Birmingham, E. H. Durfee, E. J. Glover, T. Mullen, E. A. Rundensteiner, E. Soloway, J. M. Vidal, R. Wallace, and M. P. Wellman, "Toward inquiry-based education through interacting software agents," *IEEE Computer*, vol. 29, no. 5, pp. 69–76, 1996.

[10] D. Phung, Q., C. Dorai, and S. Verikatesh, "Narrative structure analysis with education and training videos for e-learning," in *16th International Conference on Pattern Recognition*, 2002.

[11] C. Dorai, V. Oria, and N. V., "Structuralizing educational videos based on presentation content," in *International Conference on Image Processing*, 2003.

[12] Y. Li and C. Dorai, "Video frame identification for learning media content understanding," in *IEEE International Conference on Multimedia and Expo*, 2005, pp. 1488–1491.

[13] T. F. Syeda-Mahmood and S. Srinivasan, "Detecting topic events in digital video," in *8th ACM international conferences on Multimedia*, 2000, pp. 85–94.

[14] C. Ngo, F. Wang, and T. Pong, "Structuring lecture videos for distance learning applications," in *International Symposium on Multimedia Software Engineering*, 2003, pp. 215–222.

[15] M. Lin, M. Chau, J. Cao, and J. F. Nunamaker, "Automated video segmentation for lecture videos: A linguistics-based approach," in *International Journal of Technology and Human Interaction (IJTHI)*, 2005.

[16] Y. Li and C. Dorai, "Atomic topical segments detection for instructional videos," in *ACM Multimedia*, 2006.

[17] G. D. Abowd, C. G. Atkeson, A. Feinstein, C. E. Hmelo, R. Kooper, S. Long, N. N. Sawhney, and M. Tani, "Teaching and learning as multimedia authoring: The classroom 2000 project," in *ACM Multimedia*, 1996, pp. 187–198.

[18] "Microsoft producer: - a capturing and authoring tool for distance learning," 2003. [Online]. Available: http://www.microsoft.com/windows/windowsmedia/technologies/producer.mspx

[19] C. Zhang, J. Crawford, Y. Rui, and L. He, "An automated end-to-end lecture capturing and broadcasting system," in *ACM Multimedia*, 2005, pp. 808–809.

[20] S. Mukhopadhyay and B. Smith, "Passive capture and structuring of lectures," in *ACM Multimedia (1)*, 1999, pp. 477–487.

[21] T. F. Syeda-Mahmood, "Indexing for topics in videos using foils," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2000, pp. II: 312–319.

[22] T. Liu, R. Hjelsvold, and R. Kender, J, "Analysis and enhancement of videos of electronic slide presentations," *IEEE International Conference on Multimedia and Expo (ICME)*, 2002.

[23] B. Erol, J. J. Hull, and D. Lee, "Linking multimedia presentations with their symbolic source documents: algorithm and applications." in *ACM Multimedia*, 2003, pp. 498–507.

[24] A. Behera, D. Lalanne, and R. Ingold, "Looking at projected documents: Event detection document identification," in *IEEE International Conference on Multimedia and Expo (ICME)*, 2004.

[25] F. Wang, C.-W. Ngo, and T.-C. Pong, "Synchronization of lecture videos and electronic slides by video text analysis." in *ACM Multimedia*, 2003, pp. 315–318.

[26] Y. Chen and W. J. Heng, "Automatic synchronization of speech transcript and slides in presentation," in *International Symposium on Circuits and Systems (ISCAS)*, 2003, pp. 568–571.

[27] Q. Fan, K. Barnard, A. Amir, A. Efrat, and M. Lin, "Matching slides to presentation videos using sift and scene background matching," in *8th ACM SIGMM International Workshop on Multimedia Information Retrieval*, 2006.

[28] Q. Fan, A. Amir, K. Barnard, R. Swaminathan, and A. Efrat, "Temporal modeling of slide change in presentation videos," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2007.

[29] A. Amir, G. Ashour, and S. Srinivasan, "Automatic generation of conf. video proceedings," in *Journal of Visual Communication and Image Representation, JVCI Special Issue on Multimedia Databases*, 2004, pp. 467–488.

[30] M. Brown and D. Lowe, "Recognising panoramas," pp. 1218–1225, 2003.

[31] W. Zhang and J. Kosecka, “Image based localization in urban environments,” in *International Symposium on 3D Data Processing, Visualization and Transmission, 3DPVT 2006*, 2006.

[32] N. Snavely, S. M. Seitz, and R. Szeliski, “Photo tourism: Exploring photo collections in 3d,” in *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 2006, pp. 835–846.

[33] F. Bertolli, P. Jensfelt, and H. I. Christensen, “Slam using visual scan-matching with distinguishable 3d points,” in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS’05)*, 2006. [Online]. Available: http://www.cas.kth.se/ patric/publications/fedepaper.pdf

[34] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 20, pp. 91–110, 2004.

[35] ——, “SIFT demo program (version 4),” July 2005. [Online]. Available: http://www.cs.ubc.ca/ lowe/keypoints/

[36] R. Weber, H.-J. Schek, and S. Blott, “A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces,” in *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, 24–27 1998, pp. 194–205.

[37] Y. Ke, R. Sukthankar, and L. Huston, “An efficient parts-based near-duplicate and sub-image retrieval system,” in *MULTIMEDIA ’04: Proceedings of the 12th annual ACM international conference on Multimedia*. New York, NY, USA: ACM Press, 2004, pp. 869–876.

[38] S. Arya and D. Mount, “Approximate nearest neighbor searching,” in *The 14th annual ACM-SIAM Symposium on Discrete Algorithms*, 1993, pp. 271–280.

[39] P. Indyk and R. Motwani, “Approximate nearest neighbors: towards removing the curse of dimensionality,” in *Proc. of 30th STOC*, 1998, pp. 604–613.

[40] R. Hartley and A. Zisserman, *Multiple view and geometry in computer vision*. Cambridge University Press, 2002.

[41] A. Fischler, M. and C. Bolles, R., “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” *Comm. of the ACM*, vol. 24, pp. 381–395, 1981.

[42] M. Brown, “Autostitch,” 2005. [Online]. Available: http://www.cs.ubc.ca/ mbrown/autostitch/autostitch.html

[43] G. Pass, R. Zabih, and J. Miller, “Comparing images using color coherence vectors.” in *ACM Multimedia*, 1996, pp. 65–73.

[44] C. Chang and C. Lin, “LIBSVM–A library for support vector machines,” 2006. [Online]. Available: www.csie.ntu.edu.tw/ cjlin/libsvm/

[45] L. Rabiner, “A tutorial on hidden markov models and selected applications,” *Proc of the IEEE*, 1989.

[46] A. J. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Transactions on Information Theory*, vol. IT-13, pp. 260–267, 1967.

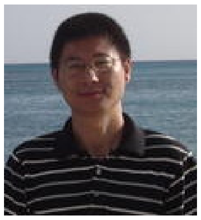
[47] D. C. Hoaglin, “A poissonness plot,” *The American Statistician*, pp. 146–149, 1980.

[48] I. Koprinska and S. Carrato, “Temporal video segmentation: A survey,” in *Signal Processing: Image Communication*, no. 5, 2001, pp. 477–500.

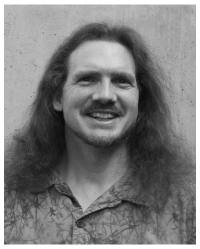
[49] S. Young, J. Odell, and P. Woodland, “Tree-based state tying for high accuracy acoustic modelling,” in *Proceedings ARPA Workshop on Human Language Technology*, 1994, pp. 307–312.

[50] “Videolectures.net.” [Online]. Available: http://videolectures.net

[51] “The univ data set for matching frames to slides,” 2007. [Online]. Available: http://vision.cs.arizona.edu/SLIC/UNIV-07



Quanfu Fan is a research staff member at IBM T. J. Watson Research Center. He received his Ph.D. in computer science from the University of Arizona in 2008 . He then joined the Exploratory Computer Vision Group at IBM, working on the Smart Surveillance System project. His research interests cover multiple aspects of computer vision, including image and video understanding, human activity recognition and video analytics.



Kobus Barnard is an associate professor of computing science at the University of Arizona. He received his Ph.D. in computer science from Simon Fraser University in the area of computational color constancy. He then spent two years at the University of California at Berkeley as a post doctoral researcher working on image understanding and digital libraries. His current interests include statistical modeling of 3D object structure in everyday, robotic, and biological domains, extracting semantics from multimodal data, and applying new algorithmic approaches to scientific data.

Arnon Amir is a research staff member at the IBM Almaden Research Center, California. He earned his B.Sc. in electrical and computer engineering from the Ben-Gurion University in 1989, and his M.Sc. (1992) and D.Sc. (1997) in computer science from the Technion - Israel Institute of Technology. His research interest includes computer vision, image and video segmentation, video and speech analysis, indexing and search and their applications for large video archives. Dr. Amir has published more than 70 peer-reviewed papers in journals and conferences and holds 15 US patents. He was awarded the 1997/98 Rothschild fellowship and several IBM research and innovation awards. Dr. Amir is a senior member of the IEEE.

Alon Efrat is an Associate Professor at the University of Arizona. The National Science Foundation granted him a CAREER award (2004), to work on ”Pattern Matching, Realistic Input Models and Sensor Placement. Useful Algorithms in Computational Geometry.” He is on the editorial board of IJCGA and of Transactions on Algorithms Engineering. He has served as program committee member of SoCG05, Broadnets06-09, ACM GIS 07-10, FOCS 09, INFOCOM 09-11 and others. Alon gained his Ph.D. in Computer Science from the Tel-Aviv University, Israel in 1998 after earning his M.Sc. in Computer Science and B.Sc. in Applied Mathematics from The Technion, Israel. He was a postdoctorate research assistant at Stanford University, and at IBM Almaden Research Center.