Searching Live Meeting Documents
"Show me the Action"

Laurent Denoue
FX Palo Alto Laboratory
3174 Porter Drive
Palo Alto, CA 94304
denoue@fxpal.com

Scott Carter
FX Palo Alto Laboratory
3174 Porter Drive
Palo Alto, CA 94304
carter@fxpal.com

Matthew Cooper
FX Palo Alto Laboratory
3174 Porter Drive
Palo Alto, CA 94304
cooper@fxpal.com

ABSTRACT
Live meeting documents require different techniques for effectively retrieving important pieces of information. During live meetings, people share web sites, edit presentation slides, and share code editors. A simple approach is to index with Optical Character Recognition (OCR) the video frames, or key-frames, being shared and let user retrieve them. Here we show that a more useful approach is to look at what actions users take inside the live document streams. Based on observations of real meetings, we focus on two important signals: text editing and mouse cursor motion. We describe the detection of text and cursor motion, their implementation in our WebRTC (Web Real-Time Communication)-based system, and how users are better able to search live documents during a meeting based on these extracted actions.

Categories and Subject Descriptors
H.3.1 [**Content Analysis and Indexing**]*: Indexing methods.* H.4.3 [**Communications Applications**]: *Computer conferencing, teleconferencing, and videoconferencing.* I.7.5 [**Document Capture**]: *Document analysis, Optical Character Recognition.*

General Terms
Algorithms, Experimentation, Human Factors.

Keywords
Live document search, indexing, real-time search, OCR, screen-sharing, video conferencing.

1. INTRODUCTION
WebRTC browser-based systems are powering a new revolution in video conferencing, and one important aspect concerns screen sharing during these online meetings. Oftentimes, people need to show other peers a deck of slides, web site designs, or discuss implementation details in a code editor.

Screen-sharing videos, either live or recorded, can be considered a new kind of document, respectively "live video documents" or "video documents". As with other types of documents, users might want to retrieve them, either during a meeting or after it has been recorded. By their nature, these video documents often contain text, and one natural way to retrieve them is by implementing keyword search.

However, the sheer amount of data (potential 30 new "pages" every second) does not make retrieval very manageable, both from the system's or the user's point of view. Fortunately, after observing our own use of video conferencing as well as analyzing one hour of screen sharing between members of another distributed team, we noticed that user's actions inside these documents can provide us with three useful signals, namely mouse cursor motion, text selection and editing. In this paper, we describe how to automatically extract these actions on live video documents, and how we use them to improve retrieval and presentation of search results. Figure 1 shows an example analysis of a 15 seconds video document where a user was discussing a slide; here the user circled around 2 main areas ("documents" and "demonstration"), and selected 2 words in the slide ("live" and "FXPAL").

Figure 1. Detecting user actions on live meeting documents: yellow circles indicate detected mouse motion; white bordered rectangles show detected text selections

2. RELATED WORK
The idea of using users' actions to improve document skimming and retrieval was pioneered by [4]. Videos indexing also uses motion found in videos in order to segment them, allowing users

DocEng'15, September 8–11, 2015, Lausanne, Switzerland.

to more easily visualize clips and retrieve objects (e.g. show me videos containing a cat). They focus on videos such as TV footage or casual user-generated videos (e.g. TREC competitions). To our knowledge, no previous work has specifically looked at extracting motion from screen sharing sessions and how to use it for better retrieval and presentation.

On web pages, mouse and keyboard tracking is used to monitor user's actions in order to design better web sites, detect when a search query was useful or not [7], or infer the emotional state of the user [8]. They have not been used to better index the pages being interacted with, and they can readily access mouse and keyboard events by injecting Javascript code inside web pages.

As opposed to instrumenting web pages with Javascript, detecting text and mouse actions in video documents is more challenging. In the next section, we first describe a typical scenario, and then present methods to automatically extract text and mouse actions, how they are used during indexing and presentation.

3. SCENARIO

During a hypothetical meeting, users need to edit a presentation. The keyword "live" appears on many slides. Without looking at what users do during the session, retrieval would produce many slides containing the keyword "live". Traditional document retrieval deals with this problem using the term frequency and inverse term frequency (TF/IDF). But in our meeting case, many key-frames contain "live", resulting in a very low discrimination between all the key-frames, leading to a poor results page.

Instead, if we extract user's actions from the video, one slide clearly stands out because the user was circling her mouse cursor around it. At another time in the meeting, another user was actually editing the keyword "live". With this extra information, a search engine would be able to retrieve these 2 key-frames.

Besides being used to better identify important segments in video documents, these signals can also be used to present the results to users. For example, Figure 1 shows a key-frame with mouse and text selections, giving users a fast overview of what happened when this slide was shared during the meeting. Alternatively, the recovered mouse motion could be animated over the key-frame so as to replicate what happened during the meeting without having to store the actual video sequence.

Figure 2. Text selections are shown in green rectangles along with the number of frames when motion rectangles have overlaid more than 3 bounding boxes of characters recognized by the OCR (here 14 and 28 times).

4. IMAGE PROCESSING

The system is implemented using WebRTC to connect participants. When a person shares her screen, the system receives a stream that is rendered by a VIDEO object in HTML5. The VIDEO is drawn periodically into a CANVAS object to process its pixel data. (See [2] for more implementation details)

4.1 Detecting Text Selections

In order to detect mouse cursor motion, a frame rate above 10 frames per second (FPS) is desirable. Below 3 FPS, the methods described below are able to detect text selections but fail to detect mouse motion.

To obtain a good text from the OCR, incoming frames are binarized and the bounding boxes of their connected components are used to form lines. Each line is scaled up to 20 pixels because the OCR engine we use requires text heights of at least 20 pixels.

The method for detecting mouse or text selection is to compute the frame difference between two consecutive frames.

Unlike previous work for indexing lecture videos and slides ([5] and [9], the content of screen-sharing sessions is very dynamic. A simple frame difference with too high a threshold would miss potentially important mouse motion and text edits, such as adding the word 'ok' in a text editor, or highlighting a word on a slide deck.

After experimenting on several video recordings, we found 32 to be a good threshold when computing the binary frame difference of two consecutive gray scale images with pixel values ranging from 0 to 255. We also extract their connected components, to be used to detect motion, in less than 25 milliseconds.

When the user double clicks on a word, the frame difference yields a strong rectangular area. If this area overlaps a word, the system detects a text selection; it also keeps track of the number of times this word was selected, which is used later for retrieval. Figure 2 shows the 2 detected text selections, along with the number of times a selection was detected. (14 and 28 times in this video clip)

4.2 Detecting Mouse Motion

As opposed to text selections, when the user moves her mouse cursor, the old and new positions are clearly seen by the bounding boxes of the connected components, see Figure 3.

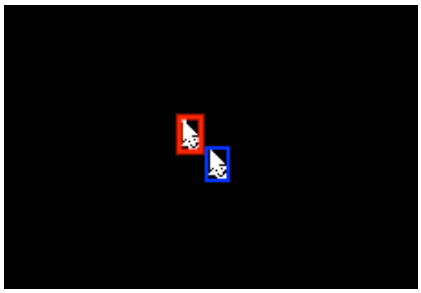

Figure 3. Mouse motion is detected by connected components of the difference between 2 frames; it is typical to see the old (blue) and new (red) cursor positions

The new mouse position is selected as the box that is most different from the previously selected mouse position. This algorithm is accurate enough to let the system know what characters have been touched or circled by the mouse cursor, see Figure 4.

Observations of actual screen-sharing sessions revealed that users often move their cursor back and forth over an area to underline a word in a document, or in small circles around a word or paragraph to "highlight" them.

To account for these observations, areas of change are ranked higher if they have a longer time span and cover a shorter area. This measure ranks low when a user moves the cursor quickly to reach a menu across the screen, but high when she moves the cursor around a word in a document during a few seconds.

Unlike the cursor of the person sharing her screen, cursors of the other participants are simply recorded in the browser window through Javascript as they move over the shared view.

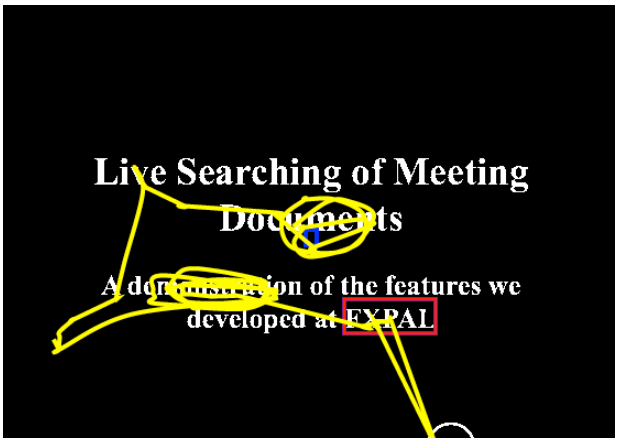

Figure 4. The rendering shows detected mouse motion, interpolated for smooth rendering between 3 points. They show the two areas where the user circled over the words "Documents" and "demonstration"

4.3 Markups

In addition to implementing shared cursors, every participant in our meeting system can markup shared screen content using an "ink" tool. When the user draws over a text area (detected again using the bounding boxes of connected components), the tool draws a straight highlight that follows the text line. Otherwise a freeform ink mark is drawn. Like the mouse and text selections detected from the video stream, these marks are also incorporated into the indexing step described below: they give further evidence that something of interest has happened over this content.

5. INDEXING AND RETRIEVAL

Many strategies could be devised to incorporate the additional signals gathered by the text and mouse detectors. For now, we are most interested in letting users search for previous frames while the meeting is taking place, not after. Our current strategy is to represent the matching frames by little dots in the main timeline, allowing users to get a quick overview of where results are located at previous times, letting them quickly jump to a particular matching key-frame by positioning the timeline at these locations.

5.1 Indexing

OCR text results are added to a normal index of words, including where on the screen they appeared. We normalize their locations by dividing the frames into 32x32 cells, and proceed similarly to normalize the locations of detected text and mouse actions. Each word thus belongs to one or more cells. Along with OCR text data, we also store the text and mouse detector results into cells, where the value of the cell indicates how much text selection or mouse motion happened. It enables users to filter search results more precisely, for example to retrieve only frames where a particular keyword was being selected as opposed to being circled over. Mouse events from remote participants are also incorporated at this stage. Figure 5 illustrates how the word weights are computed according to the amount of text and mouse actions.

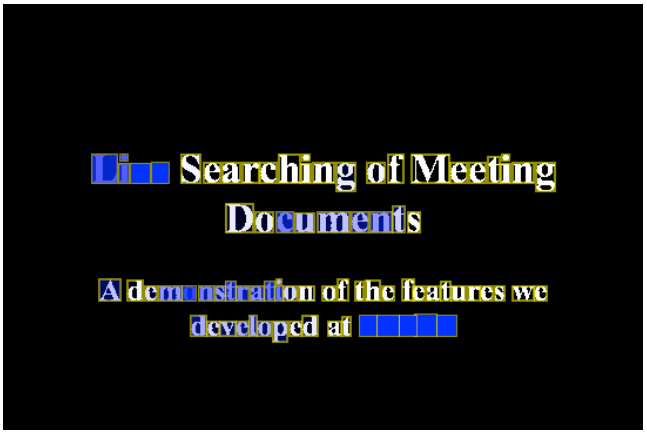

Figure 5. After analysis, the content boxes from the video document receive weights based on detected user actions; here the two text selections that happened during the meeting are clearly seen as dark blue, compared to lighter tint for mouse motion.

5.2 Retrieval and filters

When the user searches for a keyword, frames that contain this keyword are retrieved. An importance is calculated by adding the amount of action linked to the matching keywords. Obviously, the result set is filtered by user preferences (e.g. "show only text selections").

6. USER INTERFACE

Since the main focus in this work is to support search during an ongoing meeting, we chose to use the timeline as the main interface to show results matching the user's query. The timeline shows matching times with white bars, as shown in Figure 6 bottom.

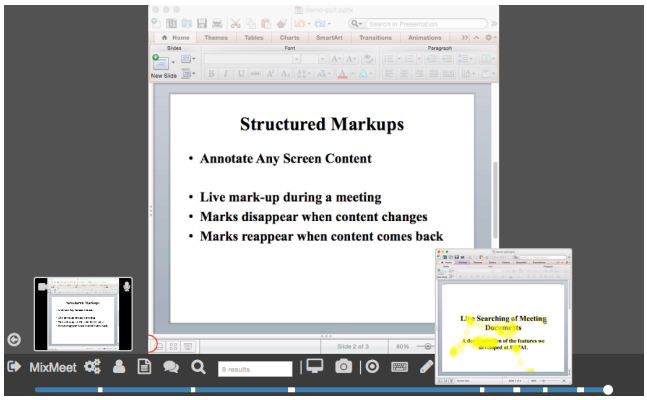

Figure 6. The user searched for "live"; the timeline shows white marks indicating matching frames; the user skims the timeline and sees the matching key-frame (bottom right of screenshot) enhanced with mouse actions

When the user hovers her mouse over the timeline, the system shows a thumbnail of that key-frame, similar to what other video players do. Unlike previous tools, an enhanced key-frame is generated by overlaying text and mouse actions, giving users a better sense of what happened during that time. This visualization was inspired by previous work on enhanced thumbnails for bookmark retrieval [10], where the authors overlaid keywords over web thumbnails and showed an improved retrieval time. Similarly, emphasizing the motion provides users with a quick way to determine if this segment in the meeting is indeed the one they want to review, or if they should keep skimming the timeline for more relevant hits.

7. CONCLUSIONS AND FUTURE WORK

We described how using content analysis helps better retrieve meeting content from screen-sharing sessions. These video documents contain a lot of redundant text; content analysis focused on text and mouse motion detection helps retrieval by identifying the points in time when the keywords were selected or acted on by meeting participants. We also described how to use this motion data to enhance the presentation of matching key-frames.

Future work will include testing the retrieval efficiency against a traditional text retrieval benchmark as well as evaluate the usefulness of these enhanced key-frames. In particular, they might become more useful during post-meeting retrieval tasks: users could be shown animated key-frames instead of statically tinted key-frames in place of static search snippets.

Because content is analyzed in real-time, other interesting services could be added such as on-demand translation for keywords, text selection for copy and paste, and increased awareness of remote participants' cursors by artificially emphasizing their appearance through color, size and ghost trailing.

Speech is yet another important signal we could use for retrieval; some browser vendors such as Google Chrome offer live speech to text for web applications through the Web Speech API; while the quality is not yet good enough for live transcripts, the text will provide enough words for retrieval purposes, especially if combined with words recognized on screen by the OCR engine, as was shown in [1] and [3].

8. ACKNOWLEDGMENTS

We thank Lynn Wilcox and Dick Bulterman for supporting this research.

9. REFERENCES

[1] Cooper, M. (2013, March). Presentation video retrieval using automatically recovered slide and spoken text. In IS&T/SPIE Electronic Imaging (pp. 86670E-86670E). International Society for Optics and Photonics.

[2] Denoue, L., Carter, S., & Cooper, M. (2013, September). Content-based copy and paste from video documents. In Proceedings of the 2013 ACM symposium on Document engineering (pp. 215-218). ACM.

[3] Hauptmann, A. G., Jin, R., & Ng, T. D. (2003, January). Video retrieval using speech and image information. In Electronic Imaging 2003 (pp. 148-159). International Society for Optics and Photonics.

[4] Hill, W. C., Hollan, J. D., Wroblewski, D., & McCandless, T. (1992, June). Edit wear and read wear. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 3-9). ACM.

[5] Yang, H., Siebert, M., Luhne, P., Sack, H., & Meinel, C. (2011, November). Lecture video indexing and analysis using video OCR technology. In Signal-Image Technology and Internet-Based Systems (SITIS), 2011 Seventh International Conference on (pp. 54-61). IEEE.

[6] Gutwin, C., Dyck, J., Burkitt, J. 2003. Using Cursor Prediction to Smooth Telepointer Jitter. In *Proceedings of the ACM Conference on Supporting Group Work*.

[7] Huang, J., White, R. W., & Dumais, S. (2011, May). No clicks, no problem: using cursor movements to understand and improve search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1225-1234). ACM.

[8] Weinmann, M., Schneider, C., & Robra-Bissantz, S. (2011). MOUSEREC—Monitoring Online Users' Emotions by Recording and Evaluating Cursor Movements.

[9] Adcock, J., Cooper, M., Denoue, L., Pirsiavash, H., & Rowe, L. A. (2010, October). Talkminer: a lecture webcast search engine. In *Proceedings of the international conference on Multimedia* (pp. 241-250). ACM.

[10] Woodruff, A., Faulring, A., Rosenholtz, R., Morrsion, J., & Pirolli, P. (2001, March). Using thumbnails to search the Web. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 198-205). ACM.