Automatic Capture of Significant Points in a Computer Based Presentation

Paul Dickson, W. Richards Adrion, and Allen Hanson
Department of Computer Science
Computer Science Building
University of Massachusetts
Amherst, MA 01003, USA
{pauld,adrion,hanson}@cs.umass.edu

Abstract

We describe an automatic classroom capture system that detects and records significant (stable) points in lectures by sampling and analyzing a sequence of screen capture frames from a PC used for presentations, application demonstrations, etc. The system uses visual inspection techniques to scan the screen capture stream to identify points to store. Unlike systems that only detect and store slide presentation transitions, this system detects and stores significant frames in any style of lecture using any program. The system is transparent to the lecturer and requires no software or training. It has been tested extensively on lectures with multiple applications and pen-based annotations and has successfully identified "significant" frames (frames that represent stable events such as a new slide, bullet, figure, inked comment, drawing, code entry, application entry etc.). The system can analyze over 20000 frames and typically identifies and stores about 100 significant frames within minutes of the end of a lecture. A time stamp for each saved frame is recorded and will in the future be used to compile these frames into a jMANIC [16] multimedia record of the class.

1 Introduction

Recently, classroom instructional delivery has moved from traditional didactic lectures toward constructivist techniques (cooperative, active, problem-based, student-centered learning). As lecturer-student interaction increases the quality of student notes often deteriorates, so it is important to have a secondary record of the lecture that students can review. A number of systems have been developed to automatically capture classroom events, e.g., [1, 2, 5, 10, 14], using various delivery modes (web-based lecture notes, record-and-playback systems). Our goal is to capture and analyze classroom events from different media streams (PC capture, cameras, other sensors and devices), store these data, and produce meaningful records for student review and for distance education. This paper describes a system that automatically creates a compact visual record of an instructor's classroom presentation as a small set of selected images (key frames) representing significant points in a computer-based presentation, including but not limited to slides, annotations, and program demonstrations. These key frames will be compiled into a jMANIC [16] presentation of the class and used in creating a table of contents for the presentation.

The system splits the output of the lecturer's PC, sending the signal both to the projector and to a second PC that samples and stores it. Using only VGA input, the system is platform independent. Because no special software is required on his PC, the lecturer requires no training or special preparation. The capture device provides a continuous series of screen captures from the lecturer's PC to the algorithm for visual inspection to determine stable points in the lecture. The algorithm is efficient enough to run at close to real time on a low-end laptop.

This paper includes a survey of related work, a description of the capture hardware and the challenges that result from introduced noise, details of the algorithm, empirical results, and conclusions and future work.

2 Related Work

A substantial body of work exists on autocapture. Much of the earliest work [4, 5, 6, 7, 10] focused on capturing "seminar formats" (lecturer in front using Powerpoint slides). These are often called record and playback systems. More recent systems have concentrated on replacing human camera operators with automated camera management [2, 8, 11, 12, 14]. Other work has focused on capturing events from more varied sources (white boards, black boards, tablets, etc.) than just slides. Those systems most

similar to our system will be described in more detail in Section 5.

There also exist a class of systems represented by MANIC [13], that expand upon record and playback systems. These systems require hand creation of their output streams, but have a better table of contents, the ability to show an enlargement of whatever the lecturer is speaking about, the ability to search the lecture and the Internet for more information on a subject, as well as other benefits. Our system is designed to further automate the creation of these improved lecture representations. The results from this program are going to be used to generate data for the cross platform jMANIC [16] version of the MANIC presentation system.

3 Data Capture and Noise

A VGA splitter is used on the presentation computer to send the output signal both to a projector and to a VGA2USB from Epiphan Systems Inc.[15]. The VGA2USB output is sent to a USB port on the capture computer where the data is stored as individual PNG images. The size of the captured images matches the screen resolution of the presentation computer. The device typically captures between 2 and 4 images per second depending on the screen resolution of the presentation computer. The output images typically contain "noise". The noise can be classified into four types: transitional capture noise, analog variation noise, shift and fuzziness noise, and signal noise.

3.1 Noise

Transitional capture noise occurs when the device captures a screen in the midst of a transition (Figure 1). Transitional captures are inevitable because VGA2USB runs constantly and is not synchronized to lecture events. It does not occur with every transition but can occur several times per lecture..

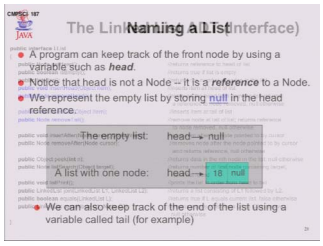

Figure 1. An image frame grabbed mid transition between lecture slides.

Analog variation noise reflects variations in the analog VGA signal from the lecturer's PC. Successive captured images of unchanging information will appear identical to the eye, but the analysis program's pixel-by-pixel comparison often finds minor differences. Typically, over 99% of the pixels in each of the red and blue channels exhibit identical intensities (i.e., the same level in the 0 to 255 range of 8-bit color) in successive, visually identical captures, while 70% of pixels in the green color channel are the same. In visually identical consecutive captured frames, over 99.9% of all pixels across all color channels differ by less then 10 levels. The average difference in a given channel is less then 2 and the standard deviation is less then 1. However, discrepancies in 0.05% of a 1400x1050 pixel image still yields 735 pixels with differences. These analog variations tend to be fairly insignificant; problems are caused when there is a visible difference between frames even though the only change is in shading (Figure 2). This example has an average difference of over 88 for all color bands and a standard deviation of about 15. This type of analog variation occurs to some degree in every lecture.

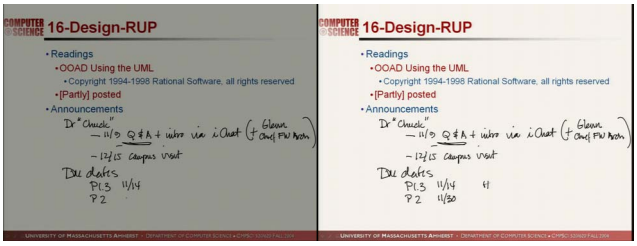

Figure 2. Two consecutive frames captured during a lecture.

Shift and fuzziness noise results from certain errors in the analog to digital conversion. Occasionally, spurious pixels are added or legitimate pixels are removed. When this occurs multiple times in an image everything becomes slightly blurred (fuzzy), although it is usually only noticeable at line edges within the image. A single large addition or removal shifts the whole image to the left or right.

Signal noise is likely due to equipment problems and, unlike the other types of noise, it is not related to information in the captured frame. It can appear in any part of the image as shown in Figure 3. Signal noise tends to last for long sequences of images; in one captured lecture over half of the lecture images had signal noise. Signal noise affects pixels in different regions from one image to the next and occasionally disappears from one image only to reappear in the next.

Signal noise does not occur frequently. In our tests, all lectures in which signal noise occurred were captured from a single laptop that appeared to have an incorrectly installed video card. The signal noise may be unique to this computer

Figure 3. A sample of some of the worst signal noise found within the data sets.

and therefore not significant to overall results, but signal noise must be expected occasionally.

3.2 Managing Noise

The first pass of the algorithm, described in Section 4.1, is designed to ensure that no frames are stored that include transitional noise or fuzziness. It also ensures that analog noise does not cause unnecessary frames to be stored. The second pass of the algorithm, described in Section 4.2, ensures that no shifts cause duplicate images to be stored. The algorithm is thus designed to handle all types of noise other than signal noise, which needs to be eliminated at its source.

4 The Algorithm

Our goal was to develop an algorithm, implemented in software on a capture computer, to analyze a large number of sampled images and store only those in which something of significance had occurred, e.g., a slide change, annotations on a slide, or changes in an application such as a pull down menu appearing. As noted, the time at which each significant point occurs is also stored. The algorithm finds these points using three passes (Figure 4). The first pass stores the last stable image preceding a change, a stable image being defined as one proceeded by three or more consecutive, identical images. The second pass analyzes the output of the first pass and eliminates repetitions. The third pass puts the output into a uniform format.

4.1 First Pass

The first pass of the algorithm runs in real time as the lecture is being recorded. It identifies changes by performing a pair wise comparison, pixel by pixel, of the most recently captured image with the immediately preceding one. The comparison function counts the number of pixels for which the difference between the successive images is greater than a pre-established threshold. A zero threshold cannot be used because of analog noise; a threshold of 100 levels per color

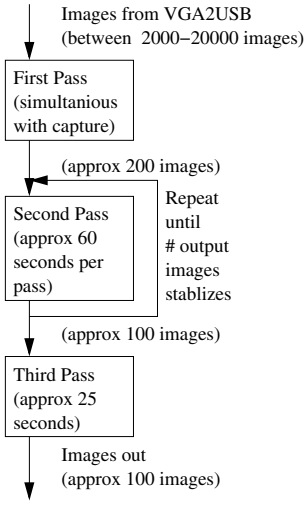

Figure 4. Overview of the algorithm.

channel was empirically chosen. This ensures that even analog noise as great as that shown in Figure 2 will be ignored. Setting the threshold is described in section 4.4.

The number of pixels differing between two successive images is divided by the size of the area searched to determine the percentage difference between the images. This percentage difference is then compared with an empirically determined threshold of 0.002%. If the threshold is exceeded, the algorithm compares the first image in the pair with prior images, and if it was stable for 3 or more consecutive images (i.e., the threshold was not exceeded) stores it (Figure 5).

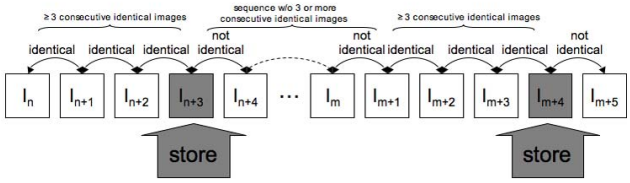

Figure 5. Selecting significant images to store.

Ensuring that an image is stable before storing it eliminates transitional and fuzzy noise which both present as constant, low level changes. This also eliminates captures that might have occurred mid-word in an annotation. The last image before a transition is stored because it will contain the most information if there has been annotation, and will be exactly the same as the first image after the previous transition if no annotation has occurred.

This pass of the algorithm runs concurrently with im-

age capture and, because of its speed, is typically finished within 3 seconds of the end of the lecture. The total number of images captured ranges between 2000 and 22000. The first pass typically reduces the count to just over 200.

4.2 Second Pass

The algorithm's second pass uses the output images from the first pass as its input and also runs pair wise image comparisons. Like the first pass, it uses a function that determines the number of pixels differing between images then uses the percentage difference between images to evaluate significance.

The first and second pass differ in important ways. In the first pass, the algorithm counts all pixels with color difference greater than 100 in at least one color channel. In the second pass, the function only counts the pixel if there is no edge in the first image within a pixel width of the located difference. This eliminates errors caused by shift and fuzziness errors. The overall difference between successive, visually distinct images is great enough that eliminating differences found at edges will not cause a change to fail to register. Annotations and data entry only rarely highlight an edge, so eliminating differences at edges usually will not affect the differences found.

The second pass uses a threshold of 0.01%. If the difference is greater than the threshold, the first image of the pair is stored; otherwise it is not. The algorithm also looks at the relative locations of differing pixels in pairs of images, and if all occur within a small region, ignores them. This technique avoids having transitory events, e.g., a blinking cursor, designated as a significant transition.

Note that the first pass, by saving the last frame in a stable image, associates the time of an image with the last time it appears in the output of the lecturer's PC. The second pass enhances the comparison made in the first pass so that a sequence of visually identical images are identified as a single stable image even if corrupted by noise. Because the second pass always stores the second of two frames that do not differ significantly, it too saves the time of the last capture prior to a transition.

Another difference between the first and second pass is that the second pass of the algorithm runs multiple times. Occasionally a small change occurs just before a major transition, e.g., a slide change. These small changes can result in (almost) identical images being stored in the first pass. The second pass runs multiple times to eliminate this duplication and continues to run until the number of output images is stable.

The second pass of the algorithm typically takes under 60 seconds for the first run and about 30 seconds for each subsequent run. For most data sets the second pass ran only twice, although on one occasion it ran four times. The average number of runs was 2.2. Typically, the first pass stores approximately 200 images, while the second stores approximately 100.

4.3 Third Pass

The third pass of the algorithm simply takes the output images of the second pass and puts them into a standard output format. It does the same with the timing information, putting it into a format that corresponds to the output image names. This pass generally takes under 10 seconds to run.

4.4 Noise Threshold Sensitivity

Both the first and second passes of the algorithm use empirically determined thresholds to separate significant changes from noise. In developing and refining the algorithm, we experimented with a number of heuristics for dynamically determining noise thresholds. Each dynamically determined threshold failed because the noise present in the data closely mimicked changes caused by transitions. Statistical analysis failed to find recognizable patterns in the noise, which exhibited fluctuations from under 10 to over 100 levels in a single color channel. These extreme differences in noise sometimes occurred within the same images.

This led us to empirically establish thresholds and then determine the sensitivity of the system to threshold changes. We found that the algorithm is not very sensitive to the threshold used to determine pixel noise; only a substantial change of greater then plus or minus 30 to the threshold caused retention of noisy slides or omission of real changes. However, the algorithm is sensitive to the threshold for distinguishing differences between successive frames. Changing the threshold used by the first pass (currently 0.002%) affects the number of images stored. Lowering the threshold causes more small changes (e.g., cursor movement) but fewer intermediate changes (e.g., annotations) to be saved. Raising the second-pass change threshold causes fewer intermediate changes to be saved.

5 Empirical Results

5.1 Lectures

We used VGA2USB to capture class presentations in four university computer science courses taught by 3 different lecturers. The lectures captured were 75 minutes long; the captured video frames consumed 5 to 6 GB of storage and contained from 4000 to 21000 images. The wide variance resulted from differences in screen resolution and occasions when a lecturer forgot to plug in the USB cable.

Seventy-one lectures were captured from the four courses. Images were projected from Macintosh and

Toshiba-Tablet laptops. The lecturers used a combination of slides, slides with annotations, Web browsers, text editors, the telnet program, and occasionally other applications.

The effectiveness of the system is evaluated qualitatively by viewing the key frames captured and determining whether they are significant classroom events and checking the input to ensure that no significant frames were missed. The algorithm runs in close to real time, completing all passes of the algorithm within 2 minutes of the end of lecture.

5.2 Results

The ability of the algorithm to capture key frames may be evaluated by comparing its results to those generated by other systems. Most systems use slides as index points into a lecture and require the lecturer to upload all slides before the lecture. These systems will contain 100% of all slides used and will have no double slides. For lectures that contain only slides our system captured more than 99.8% of significant slide transitions (new line, bullet, animation). Transitions not considered significant include cases where the lecturer moved rapidly through slides to jump from one point in a lecture to another. Such intermediate slides tend to remain on screen for under a second and either are not stored at all or do not persist long enough to be considered stable by the algorithm. The one instance in which a significant transition was missed occurred when a bullet was added and then removed too close to the edge of the frame. (A few rows at the top and bottom of the frame are masked because unrelated operating system pop-ups interfere with the algorithm.) We are looking at ways to handle this rare event. There were no consecutive, identical key frames captured by the system. Identical key frames were captured when a lecturer revisited a previously viewed slide. These results are equivalent to those of systems that use only slides to index. These results were found for all slide based lectures that did not include signal noise caused by bad cable connections.

It is more difficult to assess the quality of the key frames captured from non-slide based lectures. The eClass system [1, 3] captures every URL visited by a browser as well as all notes made on a tablet surface and stores this information onto a web page. A viewer can use the list of slides shown, notes made, and websites visited as an index into the saved data. Our system also captured 100% of all URLs visited within the lectures and all tablet annotations. Our system has the benefit of storing intermediate annotations that show the progression of the annotations while eClass has the benefit of a more concise set of index points. Unlike eClass, our system can handle any program or event and does not require the lecturer to use special computers.

The system most comparable to ours is Mediasite [9]. Mediasite captures every change that occurs on the lecturer's screen in order to create a visual index. Therefore, like our system, Mediasite can handle any program or event used and is transparent to the user. Unlike Mediasite, our system stores only those key frames where something "significant" occurs. While Mediasite stores a key frame for every transition, our system stores only those that are stable, i.e., remain on screen long enough for the viewers to consider.

A quantitative measure of the generated sequence of key frames is not possible as there is no quantitative definition of "significant" within the context of key frames. As such only a qualitative analysis of results is possible. The system successfully captured a key frame for 100% of all annotations that appeared on screen for at least 2 seconds. It also captured intermediate key frames that occurred while the annotation was being written (Figure 6). Whether each intermediate step of the annotation is significant is open to debate. The authors intend to answer this question and determine the quality of these results through future user studies.

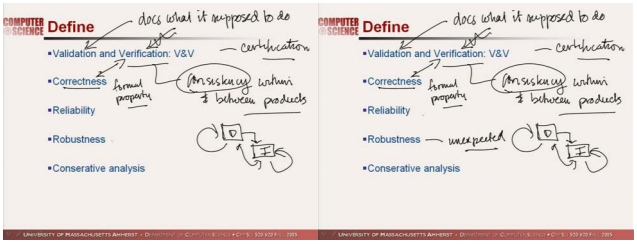

Figure 6. The two images are a pair of stored consecutive key frames stored mid annotations

The system also captured key frames from data entry and program demonstrations. During input to a program, the system captured intermediate steps when the lecturer paused typing to discuss a point (Figure 7). Similarly, when the lecturer paused in program demonstration to ensure that viewers saw each step, the system stored key frames. The pull-down menu appearing in Figure 7 is an example. Whether all intermediate saved key frames are necessary will also need to be determined through user study.

While the system stored no two consecutive, identical key frames it did occasionally store key frames that were almost identical. This occurred when a combination of minor changes, such as a cursor movement concurrent with a change in on-screen date/timer, caused the algorithm to store an extra key frame. Also the capture device occasionally produced a multiple pixel image shift resulting in two key frames being stored that were identical except for the shift. Such key frames are hard for the system to handle but

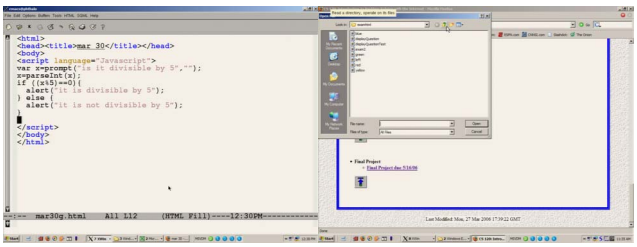

Figure 7. The image on the left is from a code implementation during lecture while the image of the right is from an application demonstration.

are not considered significant as they make up only 0.2% of the key frames stored. Currently the system fails to capture highly dynamic elements, e.g., video clips, animations. We are investigating how to handle these or if, for note-taking purposes, they need to be captured at all.

6 Conclusions and Future Work

Our algorithm is able to capture and store images to create a complete, accurate and compact visual record of computer-based classroom presentations. The system supports all PC platforms and operating systems, and a wide range of teaching and presentation styles. The system stores a small set of key frames that reflect significant classroom events and eliminates transitory events (cursor movements, system events), noise, and navigational transitions. It is efficient, completing all analysis within minutes of the end of the lecture. The system stores the set of key frames with timing information that allows the creation of Web-based notes and a variety of record and playback style formats. Importantly, the system only requires that instructors connect a PC to the sampling device as they would to a projector. The capture system can be installed on a low-end laptop or on a PC in the classroom.

We intend to develop a number of extensions and enhancements. We are equipping a lecture room with computers and cameras to allows us to capture key frames from what is written or drawn on white boards as well. We will integrate these results with the computer capture results discussed in this paper to create a complete key frame set for a lecture. We then intend to use this key frame set to automatically create a jMANIC [16] presentation.

References

[1] G. D. Abowd. Classroom 2000: An experiment with the instrumentation of a living educational environment. *IBM Systems Journal*, 38(4):508–530, 1999.

[2] M. H. Bianchi. Autoauditorium: a fully automatic, multi-camera system to televise auditorium presentations. 1998 Joint DARPA/NIST Smart Spaces Technology Workshop, 1998.

[3] J. A. Brotherton and G. D. Abowd. Lessons learned from eclass: Assessing automated capture and access in the classroom. *ACM Trans. Comput.-Hum. Interact.*, 11(2):121–155, 2004.

[4] B. Erol, J. J. Hull, and D.-S. Lee. Linking multimedia presentations with their symbolic source documents: algorithm and applications. In *MULTIMEDIA '03: Proceedings of the eleventh ACM international conference on Multimedia*, pages 498–507. ACM Press, 2003.

[5] D. Franklin. Cooperating with people: the intelligent classroom. In *AAAI '98/IAAI '98: Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, pages 555–560. American Association for Artificial Intelligence, 1998.

[6] D. Franklin and J. Flachsbart. All gadget and no representation makes jack a dull environment. In *Proceedings of AAAI 1998 Spring Symposium on Intelligent Environments*. AAAI, 1998.

[7] D. Franklin, J. Flachsbart, and K. Hammond. The intelligent classroom. *Intelligent Systems, IEEE*, 14(5):2–5, 1999.

[8] Q. Liu, Y. Rui, A. Gupta, and J. J. Cadiz. Automating camera management for lecture room environments. In *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 442–449. ACM Press, 2001.

[9] Mediasite. Mediasite.com. http://www.Mediasite.com/, Apr 2006.

[10] S. Mukhopadhyay and B. Smith. Passive capture and structuring of lectures. In *MULTIMEDIA '99: Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, pages 477–487. ACM Press, 1999.

[11] Y. Rui, A. Gupta, and J. Grudin. Videography for telepresentations. In *CHI '03: Proceedings of the conference on Human factors in computing systems*, pages 457–464. ACM Press, 2003.

[12] Y. Rui, A. Gupta, J. Grudin, and L. He. Automating lecture capture and broadcast: technology and videography. *ACM Multimedia Systems Journal*, (10):3–15, 2004.

[13] A. Schapira, K. de Vries, and C. Pedregal-Martin. Manic: An open-source system to create and deliver courses over the internet. In *SAINT-W '01: Proceedings of the 2001 Symposium on Applications and the Internet-Workshops (SAINT 2001 Workshops)*, page 21, Washington, DC, USA, 2001. IEEE Computer Society.

[14] B. C. Smith, L. A. Rowe, J. A. Konstan, and K. D. Patel. The berkeley continuous media toolkit. In *MULTIMEDIA '96: Proceedings of the fourth ACM international conference on Multimedia*, pages 451–452, New York, NY, USA, 1996. ACM Press.

[15] E. Systems. Epiphan systems vga2usb. http://www.epiphan.com/products/vga2usb/index.php, Apr 2006. Webpage.

[16] B. Wallace, W. Adrion, P. Dickson, W. Cooper, I. Ros, and K. Watts. Evolution of a cross-platform, ultimedia courseware presentation system. Submitted to: ACM Transactions on Internet Technology, June 2006.