Evaluation of Automatic Classroom Capture for Computer Science Education

Paul E. Dickson
Hampshire College
School of Cognitive Science
893 West St.
Amherst, MA 01002, USA
pdickson@hampshire.edu

David T. Arbour, W. Richards Adrion
Dept. of Computer Science
Computer Science Building
University of Massachusetts
Amherst, MA 01003, USA
{darbour,adrion}@cs.umass.edu

Amanda Gentzel
Westminster College
319 S. Market St.
New Wilmington, PA 16172, USA
gentzeam@westminster.edu

ABSTRACT

Our research into automatic recording of the complete classroom experience has led to the development of many software systems, one of which captures an image stream of all content presented on a computer. We have just completed a first deployment of this computer capture system in which 3 separate courses were recorded for an entire semester with the content being presented to students within 24 hours of the class meeting time. This system has been envisioned as a component of a complete lecture capture system but a component with real value even when used as a stand alone. In this paper we discuss student feedback to this computer capture system, revision of system functionality, and thoughts on the usefulness of capturing computer content in computer science courses in general.

Categories and Subject Descriptors

K.3.2 [**Computer and Information Science Education**]: Computer science education; K.3.1 [**Computer Uses in Education**]: Distance learning; I.4.1 [**Digitization and Image Capture**]: Sampling

General Terms

Design, Experimentation

Keywords

Automatic Capture, Image Processing, Lecture Recording, Indexing, Evaluation

1. INTRODUCTION

The University of Massachusetts Amherst (UMass) has been involved in recording lectures and turning them into indexed presentations for over 15 years [9]. Within the past 5 years the goal was updated to automatic recording and creation of indexed presentations from lectures. Specifically, the goal is to have an automated lecture recording system that does not require the lecturer to change presentation style in any way other than wearing a microphone. Currently, our capture system, Presentations Automatically Organized from Lectures (PAOL) [5] (a joint project between UMass and Hampshire College), is undergoing alpha testing and hardware problems are being resolved that have hampered full system integration.

The most stable component of PAOL is the computer capture system [4] that captures images of any significant content sent to a digital projector. These images of significant content are part of what is used by PAOL to create the index of the lecture presentation. While we are working on other aspects of PAOL, we decided to test the computer capture component as a stand-alone system. We used the computer capture system to capture all lectures from 3 separate courses for an entire semester and made the recorded content available to students within 24 hours of the lecture. The goal of capturing these images is not to create tutorials but instead to create a visual record of all material presented on screen. Student response was positive. The students found that the computer recording, even though it included only a minimal interface, added considerable value to the course.

In this paper we discuss student feedback to the computer capture system, improvements to computer recording, and recommendations for computer capture in the future.

2. RELATED WORK

The importance of determining when significant information is presented on a computer and using this information to index a lecture presentation is open to some debate. Most lecture capture systems [1, 7, 8] record a video of the lecturer and have some ability to record what occurs on screen but can only create an index from slides. Panopto has added some ability to index from non-slide content but cannot do so in an intelligent manner and only samples frames at a given interval for index points. eClass can capture slides, tablet, and programs in general but can only grab index points from slides and URLs in browser windows. We believe that it is impossible to comprehensively record computer science lectures and create a useful, searchable presentation without being able to find significant index points regardless of con-

ITiCSE'10, June 26–30, 2010, Bilkent, Ankara, Turkey.

tent presented. For automatic indexing to be effective and transparent, it must be able to capture images of significant content regardless of the program used so that the lecturer can go off script to answer a student question and be able to use any software.

Several systems enable this style of information capture. Camtasia [11] enables a user to record all events that occur on the lecturer's computer screen. Camtasia can only create an index from slides and requires the program to be installed on the lecturer's computer. Camtasia has a tendency to preempt resources and slow the lecturer's computer. It also requires that the lecturer to remember to start and stop the recording process, something difficult to remember to do when focusing on teaching. The DEBBIE system [3] has playback capacity similar to Camtasia but enables each student's work to be saved. Unfortunately, DEBBIE restricts users to a specific program and requires classrooms to be equipped for tablets. Apple's similar system, Podcast Producer 2 [2], does not provide the level of slide indexing of Camtasia. Both of these systems are fundamentally different from our system regarding what is important to capture. Our view is that full-resolution images of what occurs on screen, without the introduction of compression artifacts, do a better job of capturing presented content than does a continuous video of what occurs on screen, because the images can be more easily navigated and searched for specific information from the lecture.

The TeleTeachingTool (TTT) [12] also captures information by running software on the lecturer's computer. The TTT capture algorithm is similar to ours in that it also does frame-by-frame inspection to determine when information has changed. The results are almost identical to our system in terms of accuracy but since they rely on installed software, the system is not transparent.

The system most similar to ours in terms of functionality is Mediasite [10]. Like our system, Mediasite uses an external capture device that converts a VGA signal to digital images that are processed in order to determine when significant content is presented. Our system and Mediasite differ in two areas. First, Mediasite captures an image and creates an index point every time the material on screen changes whereas our system only stores those changes that are significant; by significant we mean that a change occurs on screen then is stable for a length of time. Mediasite's choice is fine when only slides are used to present content but not effective with program demonstration and coding examples because far too many index points occur. Second, Mediasite can recognize video on the computer and recording it while we are currently developing that capacity.

3. BRIEF SYSTEM OVERVIEW

Our system uses an external capture device to grab images of all content sent to a projector and processes those images captured in order to determine when significant content is presented. The device grabs the images sent to the projector directly from the projector by converting the VGA output into images. All images are captured at full resolution and without compression so no artifacts are added that might degrade the quality of the archive. Because our process is image based, we can look for changes in presented material regardless of the program being used by the lecturer. We define significance as content that differs from previous content presented but that also stays on screen unchanging for at least 2 seconds, Mediasite saves all content that changes. Our rationale is that a lecturer would not change what was on screen unless a new point was being illustrated and that a person reviewing the content only wants to see how the change finishes and not see it, for example, mid annotation.

The system functions as a state machine with 2 states (Figure 1). The machine starts in state *unstable* and moves to state *stable* if 3 consecutive captured images are stable but stays in state *unstable* if there is a change in the captured images. Once in state *stable* it is guaranteed to save an image as this is the important content being presented. The state machine stays in state stable until a change occurs, at which point it saves the last stable image and returns to state *unstable.* This process enables the system to capture annotations and the building of diagrams while skipping individual images in which a cursor only moves a small amount. The capture is sensitive enough to capture fine detail as seen in Figure 2, which shows the menus students must select in order to add new files to projects for iPhone development.

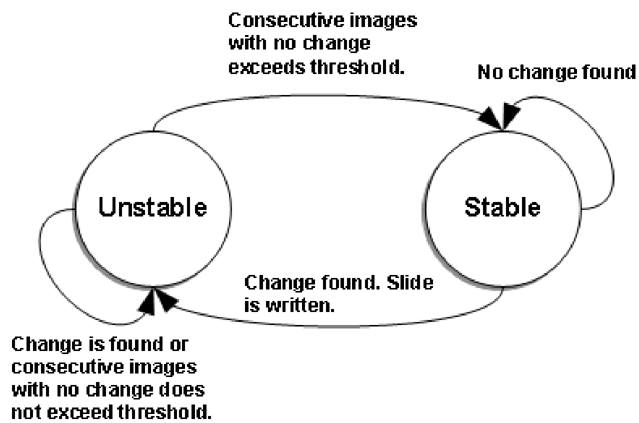

Figure 1: State machine for computer capture system.

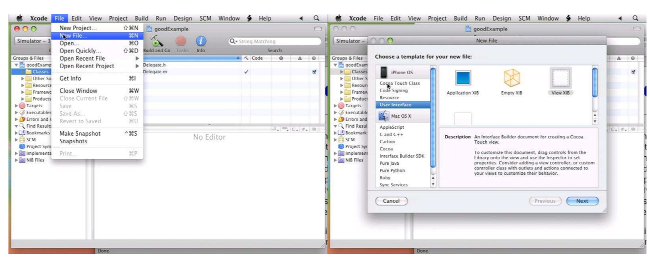

Figure 2: Captured images showing menu selection.

In previous testing our system recorded 99.8% of all significant slide transitions from a semester's worth of lectures, successfully ignoring transitions that occurred when the lecturer was flipping back to a previous slide. While no definitive analysis of non-slide-related content was performed, the system appeared to capture all information. Fall 2009 was the first semester that a comprehensive study was performed that used student feedback to determine whether all content was captured. The algorithms and results of our initial lab testing of the computer capture system were described in detail previously [4].

4. TESTING AND RESULTS

In Fall 2009 we deployed the computer capture system at Hampshire College and used it to record 3 separate classes: 2 sections of a code-based web design course and a video game design course. Of the 23 possible lectures for the web design classes, 18 and 16 were recorded, with most of the lectures not recorded being intentionally skipped because they were in-class discussions. Of the 24 possible video game design classes, 10 were recorded. All but two of the lectures not recorded were intentionally skipped because the classes were in-class work sessions or discussions. The captured images were presented using a using a Picasa-generated [6] webpage for each lecture (Figure 3) and that page was posted within 24 hours of the completion of the lecture. The left side of the webpage contains a thumbnail-based index of all captured images. The user selects the image to view by clicking on the image on the left and having it displayed in full resolution on the right. For the 3 classes the average number of images captured per lecture was 208 for each 80-minute lecture.

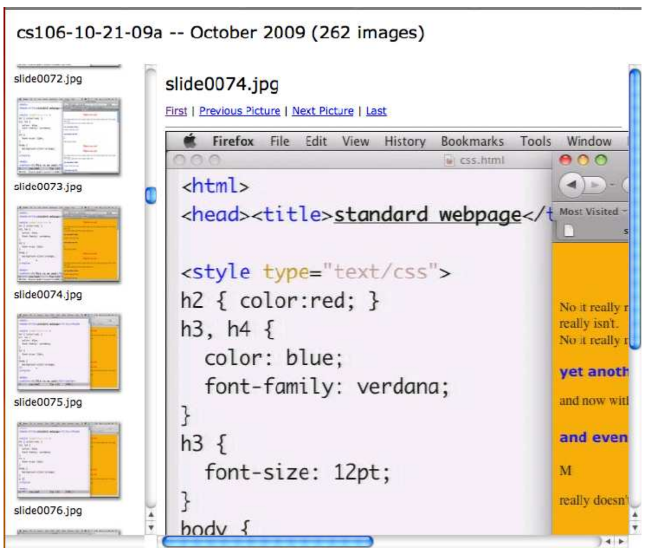

Figure 3: Captured image display webpage.

The number of images captured was far higher than expected but was determined to be appropriate. As one student said, "There were a lot of images to handle, but I fear that any fewer would mean missing a step I might have needed." The large number of images reflects that the lecturer often jumps between code and running of code when lecturing and each piece of material brought to the forefront appears important to save. This number of saved index points is more than can reasonably be used to index a video presentation though; this is addressed in Future Work.

Of the 37 students from the web design classes, 23 participated in a voluntary survey about the computer capture system. None of the 17 students in the game design course decided to take the survey though some did use the captured material. Because Hampshire College does not have grades, we could not use the incentive of a homework grade to encourage filling out a survey. Of these students who did respond, 52% said that they had reviewed the captured images after missing class. When students were asked how often they looked at the captured images, 8.7% said never, 17.4% said once or twice, 47.8% said occasionally, and 26.1% said regularly. The responses to more detailed questions about specific aspects of the captured data can be found in Table 1. This computer capture system was never intended to replace class material and the results bear this out since student disagreed with statements that the images replaced class, the textbook, or notes. These data show agreement for students who attended all lectures and those who missed some. We had believed that having access to the images might improve the course, but the student response score of 4.36 was unexpected and reflects that 87% of respondents thought this improved their course (Table 1). The response was even higher among the 12 students who missed some class as 100% agreed that the captured images made it easier to catch up, giving a response score of 4.42.

Survey statements for all students	Average
For all students	
Images improved class experience.	4.36
Images were replacement for class.	1.64
Images were replacement for book.	2.18
Images made notes unnecessary.	2.50
Images affected how I took notes.	3.27
I would like this used in my next class.	4.05
Too many images captured per lecture.	3.68
More image organization would be useful	4.41
For students who missed class	
Images were replacement for class.	1.92
Images useful for catching up on content.	4.42

Table 1: Responses to survey. Responses: 1 Strongly disagree, 2 disagree, 3 neither agree or disagree, 4 agree, 5 strongly agree. Statements are paraphrased for space. Average is average response.

The score of 4.05 reflected that 82% of students expressed interest in having access to this material in the next course they took (Table 1). This is despite the fact that students largely felt that too many images were captured and overwhelmingly felt (4.41; Table 1) that more organization of the captured images was necessary.

The written responses to the survey gave a more detailed explanation of students reaction to the captured images. Many students commented that the images affected how they took notes. Many said that knowing that they could review the images made them less worried about copying down syntax correctly and more able to focus on conceptual issues. A lot of students commented that this also made it easier for them to type examples in on their own computers during class without fear of missing something. The chief complaint was not that there were too many images but instead that the captured images were too difficult to navigate. The implication was that what they could access was not bad but would have been used much more if getting to just the images they wanted to review had been easier. These responses were universally phrased in terms of "could this be improved" instead of "this isn't any good". A typical constructive response was, "Finding the three or four slides that I was interested in often took a while... [sic] with over a hundred slides per class. Perhaps if they were labeled in some way it might be easier to search." One response seemed to sum student feedback to the use of the system, "I would find it very helpful If I could access images of material we covered in my other classes. I definitely think it would help me make my notes more comprehensive and I would learn more from the class."

5. DISCUSSION

It is becoming common practice in teaching to hand out copies of slides to students before class starts to enable easier note taking. Students appreciate this and often state that this makes it easier for them to take notes. There are arguments against this though that say that there is great benefit to having to write their own notes. The idea is that it helps reinforce the points made in class by making students process the information enough to write the ideas down. The data we have collected and our results must be viewed within the context of these viewpoints.

Our computer capture system does one better than simply handing out copies of slides from the lecture. The method based on image processing that is used by our system enables capture of content from any program used. This means that a lecturer can give copies of the content presented in class to students, as we did, without having to rely on slides and without having to prepare material in advance. Our system makes it possible to capture and present important content even if the lecturer goes off script to answer questions or does dynamic examples. Furthermore, it captures a level of detail with its ability to capture progression of presentation, which is beyond the notes that any student could take. With our system it is easy to capture the progression of menu options being chosen without resorting to capturing a video of the entire lecture.

The argument can be made that the system captures too many images and that some manual indexing of material would be advisable. The problem with this argument is that the system's strength is that it functions completely automatically and transparently. The system is intended to give students an additional reference to class material without adding work for the lecturer. We address a possible solution to this problem in the next section.

As to the whether making automatically generated notes available in general is good for student learning, we make no claims. We have no data to support or refute the idea that generating notes for high-level computer science courses that focus more on concepts or for fact-based courses in other fields affects student retention of information. We do believe that our data present the case that capturing what occurs on screen in lower-level programming-based computer science courses has a positive effect on student experience. Students stated that having access to content added value to the class and we have no reason to doubt their assessment. Hampshire College has no grades and our class sizes are small enough that it is not possible for us to to run studies that have control sections of the classes that do not have access to the captured content. We can state that the final projects generated in the web development course that used the system were of better quality overall then those turned in for the previous year's sections of the course.

We believe that capturing computer content for low-level computer science courses is so useful primarily because of code syntax and the code writing process. When students initially learn a programming language, they do not understand the intricacies of programming and often miss small bits of syntax such as semicolons and commas versus periods. The captured images provide an archive that students use to check syntax demonstrated in class. Also, when first programming, students may try to learn the steps of saving, compiling, and running code and may record this in their notes. The difference is great between written notes and being able to review images of the progression of the lecturer saving, compiling, and running code that include where the steps are typed and performed. An advantage of images over a constant video stream is that it is easier to flip through images to find the portion of a lecture that corresponds to notes then to scan through a video.

6. CURRENT AND FUTURE WORK

We are expanding the computer capture system so that it can capture videos and rapid animations when they are presented in class. We have reimplemented the capture system with a new state machine (Figure 4) that adds a state for detection and recording of videos, etc. When the system detects sections of continuous change, it will shift into a state where it records the screen capture image stream as a continuous video. When the video or animation ends, it will return to functionality like that described above and by the previous state machine (Figure 1).

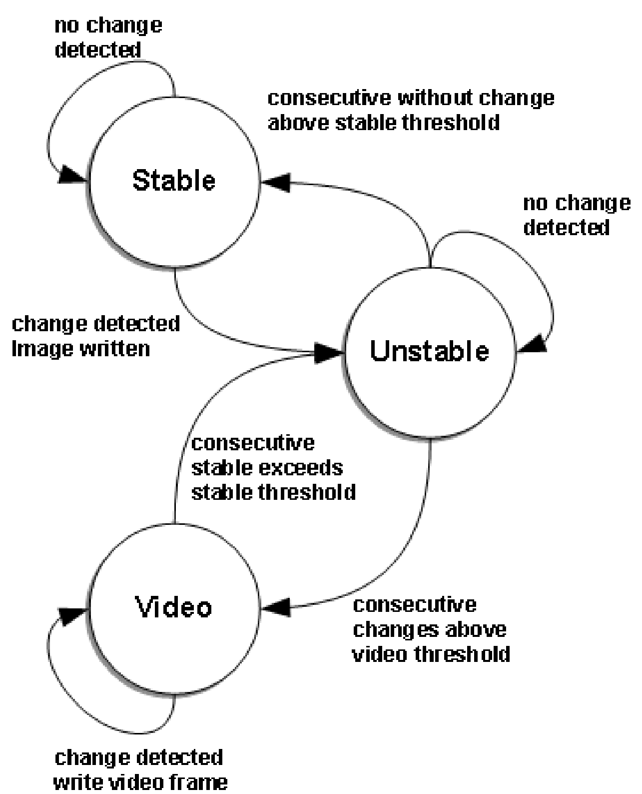

Figure 4: New state machine.

We are also improving the organization of the captured images to make it easier for students to find information.

This is a more difficult challenge. We plan to address this by a complete reorganization of our presentation and shifting from a static webpage-based view method to a dynamic method. This computer capture system is intended as a portion of PAOL [5], our larger capture system, and captured images are intended to also be used as thumbnails in a thumbnail-based index. We are beginning the process of building a web interface that pulls data from a database that includes all captured images and the data on how much one captured image varies from the next. We are developing an interface that groups similar consecutive images together under the initial image and to show only that initial image in the index that appears on the left side of the screen (Figure 5). This should make it easier to navigate between images as a user can more easily find a part of the lecture to review in this less-dense index before selecting the image and expanding the more finely grained images that follow it. Because we have complete data on how much images differ, this process can be automated and easily changed so that the user can set the level of detail they want for the displayed images. We hope to deploy this functionality within the next semester.

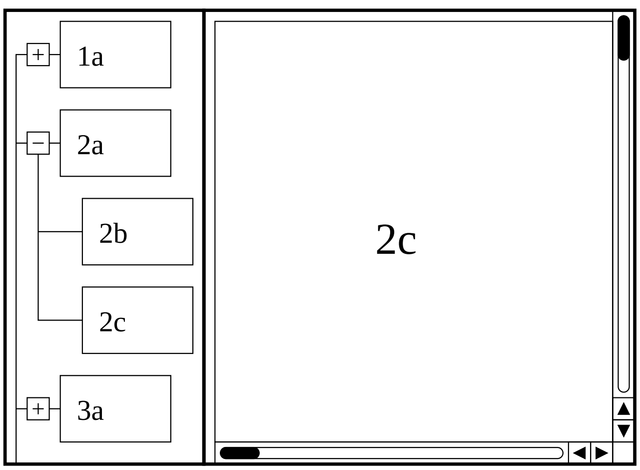

Figure 5: Layout of envisioned presentation window.

7. CONCLUSIONS

Using our computer capture system to capture the content that appears on screen is of great benefit to students in programming-based computer science classes. With no extra effort on the part of the lecturer, all content of any type presented on computer is captured and made accessible to students after the lecture. Students gave positive feedback for the system and expressed hope that it would be used for their future classes. Our system does a good job of capturing all content but needs improvement in its presentation of archive because it is currently difficult for students to find the exact piece of information they hope to review. The ease of use and benefit of the system make it an easy way to add value to a class and improve students' experience.

8. ACKNOWLEDGMENTS

The system described in this paper was made possible with funding from the National Science Foundation under grants CNS-0749756 and CNS-9979833, and by the University of Massachusetts under an ITC Strategic Initiative Grant.

9. REFERENCES

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

[2] Apple. Apple-mac os x server snow leopard-podcast producer 2. http://www.apple.com/server/macosx/features/podcast-producer.html, Jan 2010.

[3] Dave Berque, David K. Johnson, and Larry Jovanovic. Teaching theory of computation using pen-based computers and an electronic whiteboard. In *ITiCSE '01: Proceedings of the 6th annual conference on Innovation and technology in computer science education*, pages 169–172, New York, NY, USA, 2001. ACM Press.

[4] Paul Dickson, W. Richards Adrion, and Allen Hanson. Automatic capture of significant points in a computer based presentation. In *ISM '06: Proceedings of the Eighth IEEE International Symposium on Multimedia*, pages 921–926, Washington, DC, USA, 2006. IEEE Computer Society.

[5] Paul E. Dickson, W. Richards Adrion, Allen R. Hanson, and David T. Arbour. First experiences with a classroom recording system. In *ITiCSE '09: Proceedings of the 14th annual ACM SIGCSE conference on Innovation and technology in computer science education*, pages 298–302, New York, NY, USA, 2009. ACM.

[6] Google. Picasa 3: Free download from google. http://picasa.google.com/, Jan 2010.

[7] E. Machnicki and L. A. Rowe. Virtual director: Automating a webcast. In *Multimedia Computing and Networking 2002, Proc. IS&T/SPIE Symposium on Electronic Imaging: Science & Technology*, pages 208–225, 2002.

[8] Panopto. welcome to panopto. http://www.panopto.com/, Mar 2008.

[9] Agustin Schapira, Kimberly de Vries, and Cris 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.

[10] SonicFoundry. Mediasite by sonic foundry - create online multimedia presentations, briefings and courses automatically. http://www.sonicfoundry.com/mediasite/, Jan 2010.

[11] TechSmith. Camtasia studio screen recorder for demos, presentations and training. http://www.techsmith.com/camtasia.asp, Dec 2007.

[12] Peter Ziewer and Helmut Seidl. Transparent teleteaching. In Andy Williamson, Cathy Gunn, Alison Young, and Tony Clear, editors, *ASCILITE*, pages 749–758. UNITEC Institute of Technology, Auckland, New Zealand, 2002.