VIDEO-BASED INSTRUCTION FOR INTRODUCTORY COMPUTER PROGRAMMING

Eric D. Manley and Timothy M. Urness
Department of Mathematics and Computer Science
Drake University
Des Moines, IA 50311
515 271-2177
eric.manley@drake.edu, timothy.urness@drake.edu

ABSTRACT

Video replacement of in-person lecture is finding its way into more and more computer science education settings such as inverted classrooms, massive open online courses, online/distance learning, and programming camps. Since the use of video is critical to some pedagogies, the question of how it impacts student attitudes and learning is important. This study investigates this by looking at experiences in the programming unit within two sections of a broad-scope CS0 course, one of which used video-based instruction while the other did not. We found that students in the video section had a more positive view of the learning activities and thought their student-instructor interactions were more meaningful. Student performance data also suggests that video instruction may benefit student learning as well.

1. INTRODUCTION

The use of video clips have long been used as a classroom teaching tool [1], and with web-based video becoming more available, video replacement of in-person lecture is finding its way into more and more computer science education settings. Among pedagogies that make heavy use of it include inverted classrooms [4], massive open online courses [6], online/distance learning [2, 8], and programming camps [7]. This study aims to help identify advantages and disadvantages of video content delivery in computer science education. In order to isolate the benefits of the content delivery mechanism, we conducted an experiment in which the entire programming unit within a broad-scope CS0 course was taught using video lecture in one section while in-person lecture was used in another. We collected data on student performance, student attitudes about computer science, and student perceptions of their learning in order to determine what, if any, benefits there are to teaching with video in introductory computer programming (using App Inventor for Android [9], a visual block-based programming environment). The remainder of this paper is organized as follows: We discuss related work in Section 2, we cover our methodology in Section 3, we analyze the results in Section 4, and we conclude and discuss directions for future work in Section 5.

2. RELATED WORK

For a recent comprehensive review of the literature on video use in education, see Ref. [3] which looked at 53 different studies. They found that students generally had a positive attitude about the use of video on their learning and study habits and concluded that video use tended to positively impact student performance. On the negative side, they noted that video use was associated with

lower attendance, and students had negative attitudes towards videos used merely as supplementary aids or used to the exclusion of in-person instruction. The use of video has also been investigated in some recent computer science (and CS-related math) courses. Hsin and Cigas found that adding short videos to an online distance course resulted in better retention and student performance as well as lower demand for online instructor chat sessions [2]. Vilner *et al.* found that video viewing was associated with higher retention but not correlated to higher student performance [8]. And, Lockwood and Esselstein found that students preferred the experience of a video-based inverted lecture model for C++ programming [4].

3. METHODOLOGY AND DATA COLLECTION

We introduced video-based instruction in an App Inventor programming unit during the middle of the Fall 2012 section (the *video group*) of our CS0 course. The unit was taught during seven consecutive class meetings (75 minutes each, two meetings each week). In the Spring 2013 section (the *lecture group*), the same content was covered in a live lecture-based demonstration. In both cases, each of the first five class meetings included 15-30 minutes of instruction (either several 5-10 minute self-paced videos on student computer screens or lecture) in which new programming concepts were introduced in the context of a tutorial for building an app. The videos showed a screen capture of the instructor working through an app project with a voice-over explanation of the concepts. For the lecture section, the instructor watched the videos again before class and attempted to demonstrate the same content as a live tutorial while allowing for natural student interaction. The content covered included event handlers, variables, procedures, UI components, math and text operations, and if/if-else control structures. Each of the tutorials was a variation on one of the beginning tutorials provided on the App Inventor website [5], and the students had access to the online tutorials which took the form of text and static images.

We collected surveys to gauge student attitudes about programming and their perceptions of learning. We assessed a variety of student work exhibiting various levels of student learning such as multiple-choice quizzes showing low-level comprehension, lab exercises showing the ability to apply concepts, and an open-ended final assignment showing the ability/willingness to synthesize and create.

We had hoped that the student make-up of the two sections would be similar enough to directly compare performance, but unfortunately, this was not the case. Three significant dissimilarities noticed were sex, previous experience, and initial disposition to programming. The lecture group contained 28 males and 12 females while the video group contained 12 males and 22 females; and, 11 out of 40 students in the lecture group indicated they had had at least some previous experience with computer programming while only 3 out of 33 from the video group indicated any previous experience (and one no-response from the video group). And, as shown in the pre-unit survey responses summarized in Table 1, the lecture group started with a much more positive attitude toward programming to begin with. They indicated a higher level of interest in programming, considered themselves quicker learners, and said computer programming was more fun and less challenging than the video group. While the dissimilarities prevent us from

making any strong quantitative conclusions, we believe that the data is still useful, in part because the group with less previous experience and less positive initial attitude (the video group) performed as well or better than the other group.

4. ANALYSIS

In this section, we analyze the data that was collected and discuss take-aways from the study in terms of student attitudes, access to the instructor, pacing, student perceptions of the content delivery itself, and student performance.

4.1 Student Attitudes

We note that from responses to the student attitude statements on the survey were mostly unchanged between the pre and post surveys. However, there is one notable exception. The increase in agreement with "Computer programming is fun" was larger in the video group. Looking at responses in more detail, 61% of the video class was neutral or disagreed (indicated by a rating of less than 55 out of 100 on the agreement scale), but only 30% did so on the post surveys, whereas the numbers were about the same before and after for the lecture group (23% vs. 19%). Thus, it seems more of the skeptics were won over in the video group. Coupled with positive comments on the open-ended survey questions (discussed more in Section 4.4), this seems to indicate that the ability to play and pause the instruction videos at their own pace tended to make the learning experience more fun.

	V. Pr.	V. Ps.	L. Pr.	L. Ps.
I am interested in learning how to program computers.	57	57	69	68
Computer programming is challenging.	71	75	62	68
Computer programming is fun.	54	62	70	72
In most of my classes, I find that I learn more from class lectures than from reading textbooks.	72	72	76	74
I tend to learn new concepts more quickly than most people.	56	60	69	71
I would be likely to skip a class if I could read (or watch) the material and get the same information.	47	43	44	60
The lecture-based tutorials were effective in helping me learn programming concepts.				85
The video tutorials were effective in helping me learn programming concepts.		85		
If the lab assignments and instructional content were the same, I would prefer to learn programming using self-paced video tutorials.				61
If the lab assignments and instructional content were the same, I would prefer to learn programming in a more traditional lecture format.		39		

Table 1: Average student agreement (using a slider scale of 0 to 100 with 0 being strongest disagreement and 100 being strongest agreement). V: video group, L: lecture group, Pr: pre-unit survey, Ps: post-unit survey

4.2 Access to Instructor

The level of fun noted by the students also coincides with the experience of the instructor. The video group seemed more at ease while the lecture group was more chaotic, partly because there was more demand for instructor's attention. In the lecture group, much more of the instructor time was spent answering the same questions over and over and re-teaching things that had already been covered in the lecture. The amount of time spent with the struggling students was also much larger in the video group. In fact, one student in the lecture group commented that

> *It seemed like a lot of people were asking questions that could have been answered from looking at a set of instructions so you didn't have to run around the room answering the same question, fifty times.*

(note that students in both groups did have access to the website tutorials, and both groups were seen making use of these). This was not much of an issue with the video group. In their comments, three students in the lecture group also explicitly said that they needed more one-on-one attention from the instructor when working on the labs, while only one in the video group did.

4.3 Pacing

Even though the instructor was careful to keep the pacing of instruction similar between the two sections, the video group seemed more comfortable with the pace than the lecture group. In the open-ended remarks, four students from the lecture group commented that they wanted the instruction to be slowed down. On the other hand, the remarks from the video group were more favorable: three students from the video group explicitly commented positively on the pacing; one student said it should have "moved more rapidly" while another said that the assignments themselves could be "spaced out a little more."

4.4 Student Perceptions on Content Delivery

It is also interesting to note that while both groups of students found their method of instruction helpful (both averaging 85 out of 100 agreement), the lecture group gave 61 (out of 100) average agreement to the idea of learning *instead* with self-paced videos. The video group only gave 39 agreement (i.e. indicating disagreement) with the idea that they would prefer a more traditional lecture format. This was also apparent in the open-ended survey questions. When asked "Thinking only about the time spent learning programming concepts with App Inventor in [this class], what did you find helpful to your learning of programming concepts?", 80% of the students indicated that the videos were the most helpful. Here are two examples of such comments from the video group:

> *I found the videos to be very helpful because I could go back and review them if needed during my own time to help solidify concepts that weren't completely clear at first.*

I liked having the videos. Whenever I forgot a step, I would just watch the video again. When watching a video for the second time, I would sometimes put the video on pause and drag the time bar across to have a still-shot of what I specifically needed to do. Having a visual was great!

Additionally, in the question on what could have helped their learning better, five students suggested that the videos could be complemented with either live demonstration or class discussion. One student requested more video.

In contrast, responses to the "what did you find helpful to your learning" question on the post-unit survey in the lecture group were more mixed. For example, 18 out of 37 students mentioned the lectures. Ten students in the lecture group mentioned the tutorials on the App Inventor website, (in contrast to none in the video group who primarily used the videos themselves as their main reference). Interestingly, a few of the students in the lecture group discovered other video tutorials available on the Internet and used these to supplement their learning, and one student remarked that this was one of the things most helpful to learning. Three of the lecture group students suggested in their written comments that videos would help in their learning.

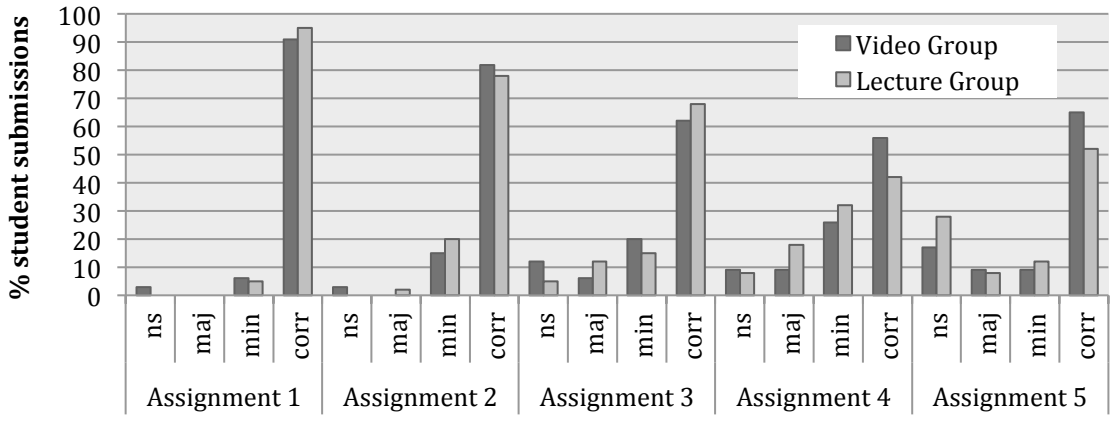

Figure 1: Student performance on lab exercises(ns: no submission, maj: major error, min: minor error, corr: correct)

4.5 Student Performance

As is apparent from student performance on lab exercises (Fig. 1), there was not a significant difference between the video and lecture groups on student performance. However, it is interesting that the lecture group did not outperform the video group despite starting more interested and viewing themselves as quicker learners. While students were allowed to base their open-ended assignment submissions on a web-based tutorial, we found that more students in the video group opted to make an original app (29% vs. 11% in the lecture group). We propose that the video group benefited from self-pacing of the initial learning as well as more meaningful interaction with the instructor during class which led to a deeper understanding of the programming constructs that students could leverage in novel ways.

5. CONCLUSIONS AND FUTURE WORK

From this study, we found reasons to believe that video-based instruction may have several benefits over lecture-based instruction in introductory computer programming with a programming environment like App Inventor for Android. Even though our video group was less interested in programming and considered themselves not-as-quick of learners, they performed at about the same level as the lecture group on comprehension and application learning tasks while exhibiting more originality on an open-ended assignment. The videos also appeared to have a bigger impact on changing student attitudes about whether programming is fun. Student comments and instructor experiences indicated that the video-based instruction led to more meaningful interaction between the instructor and students while working on lab assignments. The individualized pacing allowed by video was also seen as a benefit, and student comments indicated that they found the video method to be more helpful for their learning. In the future we would like to do a study like this on more similarly composed groups and extend the study to more advanced computer science topics.

REFERENCES

[1] Berk, R. A., Multimedia teaching with video clips: TV, movies, YouTube, and mtvU in the college classroom. *International Journal of Technology in Teaching and Learning*, 5 (1), 1–21, 2009.

[2] Hsin, W.-J., Cigas, J., Short videos improve student learning in online education. *Journal of Computing Sciences in Colleges*, 28 (5), 253–259, 2013.

[3] Kay, R. H., Exploring the use of video podcasts in education: A comprehensive review of the literature. *Computers in Human Behavior*, 28, (3), 820–831, 2012.

[4] Lockwood, K., Esselstein, R., The inverted classroom and the CS curriculum. *Proceeding of the 44th ACM technical symposium on Computer science education*, 113–118, 2013.

[5] Massachusetts Institute of Technology, Tutorials, http: //appinventor.mit.edu/explore/tutorials.html, retrieved October-December 2012 and April-May 2013.

[6] Rodriguez, O., MOOCs and the AI-Stanford like courses: two successful and distinct course formats for massive open online courses, *European Journal of Open, Distance, and E-Learning*, 2012.

[7] Urness, T., Manley, E. D., Generating interest in computer science through middle-school android summer camps. *Journal of Computing Sciences in Colleges*, 28 (5), 211–217, 2013.

[8] Vilner, T., Zur, E., Sagi, R., Integrating video components in CS1. *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, 123–128, 2012.

[9] Wolber, D., Abelson, H., Spertus, E., Looney, L., *App Inventor*. O'Reilly Media, 2011.