Active vs Passive Training for Educational Software

Ruth Wylie (rwylie@cs.cmu.edu)
Human-Computer Interaction Institute, 5000 Forbes Avenue
Pittsburgh, PA 15213 USA

Benjamin Shih (shih@cmu.edu)
Machine Learning Department, 5000 Forbes Avenue
Pittsburgh, PA 15213 USA

Abstract

Computer-based instructional interventions are increasingly popular in classroom curricula. For students to benefit from these interventions, it is vital that they are properly trained to use the software. We investigate the effects of using active versus passive training techniques to familiarize users with a graphical programming environment, Alice. We examine the impact using measures of knowledge, user preferences and motivation. Participants in both training environments gained domain knowledge and interface familiarity, but passive training participants believed that Alice was easier to use, spent more time using Alice when given the choice, and demonstrated better preparation for future learning.

Keywords: Passive vs Active Instruction, Computer Science Education, Multimedia Learning, Motivation

Introduction

Many curricula incorporate software tutors, simulations, and computerized testing systems. Without adequate user training, these technologies are at best inefficient and potentially ineffective. Since this can be users' first exposure to the software, it is also important to include both procedural training to familiarize users with the interface and motivational elements to encourage users to continue using the system. However, despite the importance and complexity of this training task, it is presently unknown whether an active approach, like an interactive tutorial, or a passive approach, like an instructional video, is more effective.

Research in attention suggests that interactivity improves student engagement (Bonwell & Eison, 1991), but evidence from the worked-examples literature (Clark & Mayer, 2003) and cognitive load theory (Sweller, 1988) suggest that well-designed passive training may lead to greater student learning. Generally, interactive instruction offers scaffolded practice, immediate feedback, and allows the learner to control the pace and the experience of using the system; passive instruction offers fully worked and explained examples, opportunities to self-evaluate, and prevents the learner from engaging in poor time management and from making confidence-destroying errors. Both approaches are supported directly by theory and indirectly through empirical evidence, but the choice between them is ambiguous.

We present results comparing active and passive training for the programming environment: Alice. Developed by Carnegie Mellon University, Alice offers users the opportunity to develop programs (called worlds) using a graphical, click-and-drag interface and fully animated, three-dimensional characters (Conway, et al., 1999). This work uses an Alice variant, Storytelling Alice, that includes characters, methods, and entire worlds of props and backgrounds designed to simplify the building of worlds in Alice. The original Storytelling Alice includes an interactive tutorial (Kelleher & Pausch, 2005) that provides basic training in the use of Alice. We compare that tutorial with a narrated, passive video of an expert user completing the same tasks as those covered in the tutorial. We find that while students in both conditions learn domain knowledge equally well, participants in the passive condition show measurable gains over their active condition counterparts on measures of motivation; preparation for future learning; and attitudes and opinions of Alice.

Instruction

For active training, we used the interactive tutorial that is currently included in the version of Storytelling Alice publicly available for download. In addition to serving as an ecological control, the tutorial is a typical example of an active training environment (Kelleher & Pausch, 2005). The tutorial's goals are to introduce the interface and to motivate potential users to continue using the program. The tutorial uses a technology called stencils (Figure 1). Directions are displayed on-screen with floating yellow notes while semi-transparent blue overlays (stencils) occlude irrelevant interface elements. The stencils direct attention to the important interface elements while limiting user actions to those same elements. This makes it more difficult for users to commit errors. However, in the event that a user does make a mistake, they are immediately notified of the error and asked to repeat the step.

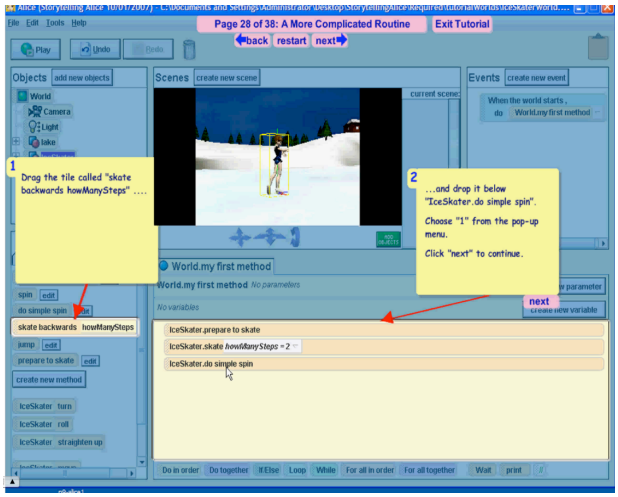

Figure 1: Screen capture of the interactive tutorial. Students read directions from the yellow notes and perform the steps themselves. Stencils (the blue overlays) focus attention and reduce student errors.

Our version of passive training was nearly identical to the interactive tutorial. During active training, students read instructions and then completed the tasks on their own; during passive training, students watched a video screen capture of an expert user completing those same steps. In the video, there were no stencils and notes. Instead of reading notes, participants listened as the expert user described her actions. The script for the passive training video was created by copying the instructions displayed in the interactive tutorial, so the material covered in both training conditions was identical.

Theoretical Predictions

The two forms of training varied on a variety of dimensions (Table 1) with each dimension offering different theoretical predictions on whether active or passive training would result in better learning. Active learning theory suggests that students need to actually do the work. Listening is not sufficient; students need to engage with the material in order to develop understanding (Bonwell & Eison, 1991). Thus, according to this line of work, students should learn more from the interactive training than the passive training. Further, the interactive training follows one of the eight cognitive tutor design principles and provides immediate feedback on student errors. Immediate feedback has been shown to make learning more efficient and help students diagnose their misunderstandings (Anderson, et al, 1995).

However, not all existing work points to the superiority of active training. Research on worked examples suggests that studying worked examples is more efficient than problem solving, especially for novices (Clark & Mayer, 2003). A cognitive theory of learning also suggests that passive training may be better due to the modality effect (Mayer, 2001): the modality effect predicts that instruction presented both visually and orally, as in the passive training condition, is better than instruction presented via only one mode, as in the active training condition where everything is presented visually.

Table 1: Differences between instructional conditions

	Interactive Tutorial	Passive Training
Behavior	Doing	Watching
Response to errors	Feedback	Error-free
Modality	Reading	Listening

Methodology

Participants

Participants in the study were 35 adults (20 female, 15 male) who responded to an online posting. The participants' mean age was 26.3 years (sd = 7.5), and on a self-report scale of computer programming experience, the mean rating was 1.6 (sd = 0.9) with one representing no computer programming experience and four representing extensive programming experience. Participants were paid for their participation.

Experimental Design

The overall study was constructed to appear as two separate but related studies conducted by two separate researchers (see Figure 2). We will refer to these as Phase 1 and Phase 2. When participants arrived, the first researcher described Phase 1 as an investigation of how best to teach Storytelling Alice. Phase 1 included a training session (either active or passive), a computerized assessment embedded in Alice, a paper assessment, and a survey, all administered by the first researcher. The first researcher then told participants that the first study was finished and introduced them to the second researcher. The second researcher described the goal of Phase 2 as an investigation of how interface affects storytelling. While participants were lead to believe that Phase 2 was a separate study, it was actually an empirical measure of participant motivation. Phase 2 included several surveys as well as a general interaction segment. The two phases were conducted by separate researchers with separate consent forms and separate final surveys.

Prior to their arrival, we sent participants a pre-study survey to determine their computer programming experience and their attitudes towards computers and computer science. Upon arrival, the first researcher briefed participants on Phase 1, obtained consent, and randomly assigned participants to either the active or passive training conditions. After the training, participants completed a domain knowledge assessment embedded in Storytelling Alice. The end product of the assessment was a functional Storytelling Alice program telling the story of *The Three Little Pigs*. The embedded assessment had three segments (Embedded 1, Embedded 2, and Embedded 3) and after each segment, the first researcher opened a new Alice world so errors in one segment would not impact other segments. Embedded 1 involved adding specific characters to the

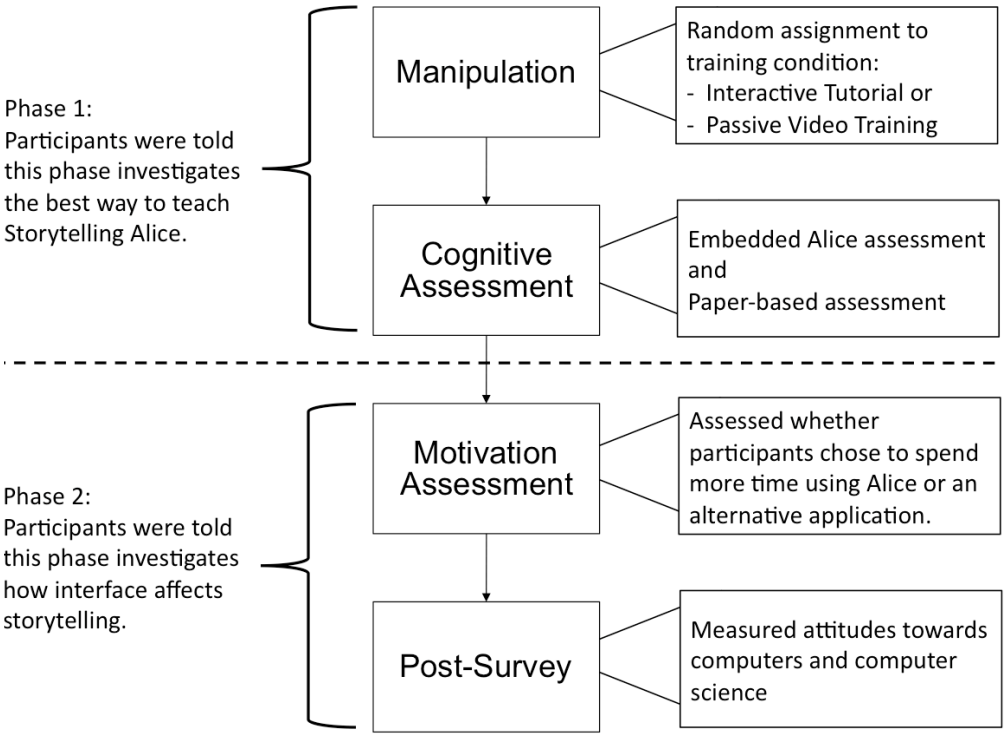

Figure 2: Presenting the experiment as two separate studies was slightly deceptive, but allowed us to measure student motivation empirically.

world and then positioning them according to instructions. Embedded 2 involved adding Storytelling Alice code to the world. The tasks in Embedded 1 and Embedded 2 were covered during training. Embedded 3, however, featured transfer tasks that were only related to, not identical to, those included in the training.

After the embedded assessment, participants completed a paper-based assessment that included: opinions on Storytelling Alice, opinions on the training, interface questions, and code interpretation questions. The first researcher then told participants that Phase 1 was over and introduced the participant to the second researcher. At the beginning of Phase 2, participants had five minutes to skim a packet containing all available Storytelling Alice characters and to brainstorm a story. The second researcher asked participants to tell the same story in both Storytelling Alice and Microsoft PowerPoint™, and explained that images of the Storytelling Alice characters would be available in PowerPoint. Participants were randomly assigned to start with either PowerPoint or Alice. Since participants had recently completed the training in Phase 1, they received no additional instruction. The second researcher did provide all participants with a brief overview of PowerPoint, mostly focused on inserting pictures, text, and dialogue boxes. Participants worked with each application for five minutes. They then had twenty minutes of free choice: they could spend the whole time with Alice, the whole time with PowerPoint, or switch between the two programs as often as they liked. After each of the 5-minute initial work periods, participants completed a brief survey about their experience with the given program. After the twenty-minute free time, participants completed a final survey that asked them to compare the stories they made using Alice and PowerPoint. The survey also included several questions assessing participant attitudes towards computers and computer science.

Results

Interface Familiarity

The first goal of Alice training is to familiarize users with the interface. In the study, we included interface questions on both the paper assessment and in the embedded Alice assessment. The paper assessment included two types of interface questions: five questions required participants to label a window of the Alice interface, e.g. as the "World Window"; five questions asked participants to describe the purpose of a window, as shown in Figure 3. In both cases, participants were offered choices a, b, c, d, and e along with a corresponding image of the interface component. For these questions, there was no significant difference between conditions. On the five labeling questions, active condition participants averaged 4.35 questions correct; passive condition participants averaged 4.67 questions correct ($p=0.407$, two-sided t-test). On the five function questions, active condition participants averaged 3.00 questions

correct; passive condition participants averaged 3.17 questions correct (p=0.732, two-sided t-test).

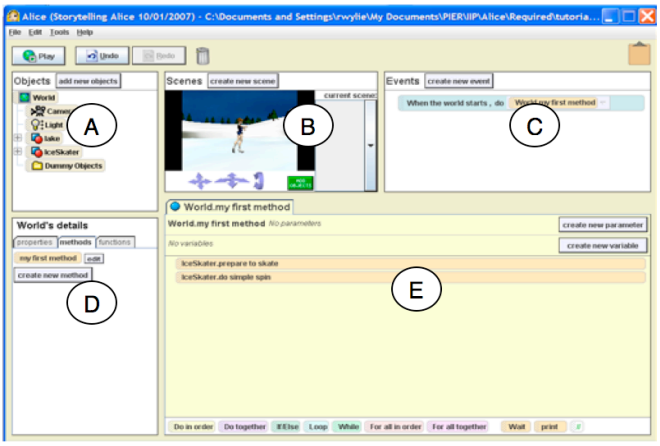

Figure 3: In the paper assessment, participants were asked to identify the names of the windows of the interface and their function.

In the embedded assessment, Embedded 1 asked students to find certain characters in the Alice gallery, place them in the world, and then position them according to instructions. This test was largely an assessment of interface knowledge and did not require any knowledge of any computer science or programming constructs: participants merely had to find and manipulate Alice characters. Out of 8 required steps in the first test, participants in the active training condition averaged 6.82 steps correct; passive condition participants averaged 6.17 questions correct (p=0.547, two-sided t-test).

Domain Knowledge

In addition to familiarizing users with the Alice interface, one goal of training is to provide users with basic programming knowledge. However, this is a secondary goal and the training only addresses domain-specific knowledge indirectly, so we limited our assessment to five domain-knowledge questions drawn from previous studies (Moskal, et al., 2004). For these questions, participants were given a sample of Alice code and asked to evaluate what would happen if it were run. There were no significant differences between conditions. In the active condition, participants averaged 4.06 questions correct; in the passive condition, participants averaged 4.39 questions correct (p=0.275, two-sided t-test). Most errors, regardless of condition, were on the last and most difficult question.

Preparation for Future Learning

After the training, the goal is for users to continue interacting with Alice for some period of time; thus, one metric for successful training is how easy it is for users to acquire new Alice skills after completing the training. To measure this preparation for future learning, Embedded 3 asked students to perform a series of tasks requiring skills not covered during training. The assessment asked students to complete seven tasks: use the copy functionality on a line of code (abbreviated as "copy"), edit a method's text parameter (edit text), change an existing numeric parameter (edit numeric), add two walk methods with parameters (add parameters), create a loop (create loop), set the number of iterations for the loop (edit iterations), and place a set of commands into the loop in a specific order (order). The first four tasks were minor variations on training tasks. For example, during training, students added methods with a text parameter, but never edited one. The last three questions were more challenging and involved loops. The training covered do-together methods which, like loops, are compound constructs that include more than one line of code; unlike loops, do-togethers have no special iteration parameters or ordering requirements. The results are shown in Table 2. Students demonstrated impressive performance across conditions for the edit text, edit numeric, and add parameter tasks. They showed lower, but condition independent, performance on the copy task, which utilized an aspect of the Alice interface that training did not cover in detail. The most interesting results, however, were in the loop tasks. When asked to add a loop and then edit its number of iterations, participants in the passive training condition performed well (80% correct), but participants in the active training condition were only correct half as often. The differences for both the create loop and edit iteration tasks are statistically significant (p=0.011 for the loop task; p=0.032 for the iterations task; Fisher's Exact Test). Performance on the ordering task, which has no comparable training task, is near floor, but the passive condition marginally outperforms the active condition (p=0.072). However it is important to note that the three tasks are strongly correlated and should be treated as only one piece of evidence. Still, participants in the passive condition perform at par with participants in the active condition on the four near transfer tasks and dramatically outperform their counterparts on the far transfer task(s).

Table 2: Participant performance on transfer tasks (Fisher's Exact Tests)
* p < 0.1

	Copy	Edit Text	Edit Num	Add Params	Loop*	Iterate*	Order*
Active	0.44	0.88	0.75	0.81	0.38	0.38	0.06
Passive	0.57	0.93	0.93	0.93	0.86	0.79	0.36
p-value	0.48	0.63	0.19	0.36	0.01	0.02	0.06

Surveys

At the end of the first study, participants completed a survey of their opinions about Alice and their opinions about the training they just received. The first five prompts were: "Alice is confusing", "Alice is cool", "Alice is annoying", "Alice is easy to learn", and "Alice is entertaining". Students were given 7-point Likert scales ranging from "Strongly Disagree" to "Strongly Agree". The results are

shown in Table 3. Only responses to "Alice is easy to learn" showed a significant difference between conditions. Participants in the passive condition stated a significantly (p=0.026, t-test) higher agreement that Alice is easy to learn. Participants averaged a "Disagree" with the negative statements and an "Agree" with the positive ones. On a much later survey, participants responded to the prompt: "I enjoy computer programming". Their responses were close to "Neither Agree nor Disagree", averaging about 3.6 on the 7-point Likert scale. This suggests that students do not fully associate Alice, which they agree is entertaining, with computer programming.

Table 3: Participant responses to a survey about Alice. (1=Strongly Disagree, 7 = Strongly Agree)

	Confusing	Cool	Annoying	Easy to Learn *	Entertain.
Active	2.12	4.29	2.06	3.88	4.06
Passive	2.00	4.00	1.94	4.56	3.93

Participants also completed a survey of their opinions on the training using the same 7-point Likert scale as before. They received the following prompts: “I was able to write programs in Alice”, “The instruction was fun to complete”, “I learned a lot from the instruction”, and “I could have written programs in Alice without any instruction”. The results are shown in Table 4. The responses are generally ambivalent, with participants neither agreeing nor disagreeing that they can write programs, learned a lot from the instruction, and that the instruction was fun to complete. Only one of the prompts elicited a marginally different (p=0.061, two-sided t-test) response, with students in the active condition disagreeing more strongly with the statement, "I could have written programs in Alice without any instruction". This is interesting because, based on the earlier knowledge assessments, we know that participants in the passive condition learned the same or more from the training than their active condition counterparts. Yet passive condition participants showed less agreement that they needed the instruction at all.

Table 4: Participant views on the training (1=Strongly Disagree, 7 = Strongly Agree)

	Able to Write Programs	Learned a Lot	Could Have Written	Fun to Complete
Active	3.76	4 .00	2.29	3.53
Passive	4.00	3.94	3.06	3.78

We also offered participants the following two prompts: “I am likely to pursue further study of computer programming” and “I am likely to continue using Alice”. Participants slightly disagreed to both prompts, with averages very close to 3. There was no statistical difference between conditions.

Motivation

An additional goal of Alice training is to motivate users to continue using Alice in the future. Our primary measure of motivation was the free time assessment from Phase 2. Students were given a choice between using Alice and using PowerPoint for a twenty-minute period. Active condition participants spent an average of 68% of their time in Alice; passive condition participants spent an average of 89% of their time in Alice. The time participants spent in Alice is trending towards significant (p = 0.099, t-test) and suggests that the training condition does impact perceptions of Alice and desire to continue using it (Table 5). The motivation measure is empirical evidence that participants in the passive condition actually chose to spend more time in Alice.

Table 5: Percent of time spent using Alice and percent of students spending more time in Alice during the 20 minute “free time” period

	% Time Spent in Alice	% Students Spending More Time in Alice
Active	68%	70%
Passive	89%	89%

After each of the initial 5 minute periods, whether with Alice or PowerPoint, participants completed a short survey with three questions: “How easy or difficult was using Alice/PowerPoint to tell a story?”, “How fun or boring was using Alice/PowerPoint to tell a story?”, and “How pleased or displeased are you with your story at this point?” These questions used a 7-point Likert scale ranging from “Very Fun” to “Very Boring”, with the appropriate substitution of terms. At the end of Phase 2, participants completed one final survey. They were asked: “Which would you rather use to tell stories?” and “If we were to share one of your stories online, which would you prefer to share?” These two questions used a 7-point Likert scale ranging from “Definitely Alice” to “Definitely PowerPoint”. None of the responses were statistically different between conditions. Overall, participants strongly preferred Alice on all but one count, their relative contentment with their stories, about which they were ambivalent.

Conclusions

While we found no differences between active and passive training on measures of interface familiarity, domain knowledge, or near-transfer, we found significant differences on several measures. Participants who received passive training believed that Alice was easier to use, spent more time using Alice when given the choice, and demonstrated better preparation for future learning. We also found that passive training made participants believe the training was less critical. Overall, it appears that passive

training offered several advantages over active training, but without any corresponding disadvantages. This is especially significant since active training systems are more difficult and time consuming to build.

There are several theoretical explanations for these results. Under cognitive load theory, the passive training condition, having a lighter cognitive load, would engender participant beliefs that Alice was easier to learn and that training was not required. Further, perhaps the lighter cognitive load allowed participants to notice interface elements during the training that were not explicitly covered in the training. For example, participants in the passive training condition may have been more likely to notice the loop widget, which is physically next to the do-together widget and thus be more likely to successfully complete the transfer tasks. Finally, the high confidence resulting from lower cognitive load could provide motivation to continue using Alice over PowerPoint.

Another possible explanation is novelty. Alice may have been perceived as easier to learn for the passive training condition participants since their first hands-on experience with Alice was post-training; while active training condition participants would have had potential frustrations and errors during their training. This would also extend to the transfer tasks. For passive condition participants, their early (post-training) experiences with Alice required searching for interface components because they had no stencils to guide them. This may have granted them a mastery of search not shared by their active condition counterparts, which could yield improved performance on the loop task. This experience with self-driven learning may have diminished the perceived gap between Alice, with which the participants are relatively inexperienced, and PowerPoint, with which they are familiar.

Unfortunately, our distinction between active and passive training is confounded across several dimensions: user behavior (doing vs. watching), system response to errors (immediate feedback vs worked-examples), and training modality (reading vs listening). It would be invaluable to have future studies that separately tested training systems along each of these dimensions.

Acknowledgements

Special thanks to Jamie Jirout, Ken Koedinger, David Klahr, Tom Lauwers and the rest of the PIER community for their help and feedback on this project. The research reported here was supported in part by the Institute of Education Sciences, U.S. Department of Education, through Grant R305B040063 to Carnegie Mellon University.

References

Bonwell, C. & Eison, J. Active Learning: *Creating Excitement in the Classroom*. ASHE-ERIC Higher Education Report No. 1. Washington, D.C.: 1991.

Clark, R. C., & Mayer, R. E. (2003). e-Learning and the Science of Instruction : Proven Guidelines for Consumers and Designers of Multimedia Learning. San Francisco: Jossey-Bass.

Conway, M., et al. (1999) Alice: lessons learned from building a 3D system for novices. Proceedings, CHI 2000, 486-493.1998, 153-162.

Kelleher, C., & Pausch, R. (2005). Stencils-based Tutorials: Design and Evaluation. Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 541-550). New York: ACM Press.

Mayer, R.E. (2001). *Multimedia learning.* London: Cambridge University Press.

Moskal, B. Lurie, D. & Cooper, S. (2004) Evaluating the effectiveness of a new instructional approach. SIGCSE 2004: 75-79

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. Cognitive Science, 12(2), 257-285