Student Opinions of Alice in CS1

Daniel C. Cliburn
Assistant Professor
Department of Computer Science
The University of the Pacific, dcliburn@pacific.edu

Abstract - Alice is a novel programming environment for building 3D virtual worlds. The graphical programming interface of Alice allows beginning programmers to learn about concepts such as object-oriented design and recursion without the need to struggle with syntax errors. This paper discusses the opinions of Alice from 84 students who took an introductory programming course (CS1) that covered both Alice and Java in the same term. Fifty students (59.5%) responded on the survey that the prior experience with Alice helped them to learn Java later in the term. Fifty-six students (66.7%) recommended continued use of Alice in the course although many suggested reducing the amount of time spent on Alice. While these numbers could be interpreted as a positive result, they have actually discouraged our department from continuing to use Alice in CS1. The number of students who were not sure that Alice was helping with their understanding of Java later in the term was simply too great to justify continued coverage of Alice in the course.

Index Terms – Alice, CS1, Evaluation, Surveys.

INTRODUCTION

Concern over declining enrollments in Computer Science (CS) has motivated substantial effort to develop approaches that make undergraduate programs more attractive to potential majors. Also of interest are techniques to help students in introductory programming courses achieve success at higher rates. One of the most exciting and publicized developments recently has been the introduction of the Alice programming environment. Alice is “designed to be a student’s first exposure to object-oriented programming” by allowing students to easily create story and game applications with graphical 3D worlds [1]. The drag-and-drop interface of Alice prevents programmers from creating syntax errors, which allows students to focus on programming concepts rather than the tedium of debugging code. The Alice programming environment is shown in Figure 1.

Despite somewhat widespread adoption of Alice in college courses (see the presentation from SIGCSE 2007 available at: www.alice.org/SIGCSE_Tea_Party_2007.htm), there remains to date surprisingly little information describing the successfullness of these approaches for teaching introductory programming using Alice. One notable exception is the work of Moskal, Lurie, and Cooper [2], which found that at risk students (those with a weak background in mathematics or little previous programming experience) who took an Alice programming course prior to or concurrently with CS1 (the traditional first computer programming course in the CS major) performed better and continued on to CS2 (the second major course) more often than at risk students who did not take the Alice course. However, opinions of Alice in other college classroom settings have not always been as positive, particularly in courses that combine Alice with another high level programming language. For instance, Powers, Ecott, and Hirshfield [3] report that students in their CS0 course had a difficult time transitioning from Alice to other languages such as C++ and Java, with some aspects of Alice, such as the object model, actually leading to misconceptions amongst students when it came time to learn the second language.

FIGURE 1
THE ALICE v2.0 PROGRAMMING ENVIRONMENT

In this paper, a different approach is taken to evaluate the successfullness of Alice in the classroom. In some sense, the opinions that matter most are those of the students who are using Alice to learn programming concepts. Simply, if students do not feel that Alice is helping them learn high level language programming, then there seems to be little reason to continue to use Alice in a CS1 course. Conversely, if students feel that their first programming experience with Alice is helpful when they learn another language (Java in this case), then this provides strong support for the continued use of Alice in the course. Surveys have been used
previously to measure student attitudes towards programming before and after using Alice in an introductory Computer Science course for non-majors [4], and reflective essays and focus group discussions have been used to collect student perspectives on Alice and pair programming in a similarly structured course [5]. However, what appear to be missing in the literature are the detailed opinions of students who are taught both Alice and another high level language, such as Java or C++, in the same course. Of particular interest are the opinions on whether or not the Alice experience helped students to learn the second language. This approach of combining Alice and Java in the same course is supported by the new textbook, *Alice in Action with Java* [6]. The purpose of this paper is to present the perspectives of students on Alice who have taken such a course.

The remainder of this paper is arranged as follows. Section 2 discusses Alice in more detail, with a focus on published literature that describes the use of Alice in educational settings. Section 3 describes the structure of the CS1 course at the University of the Pacific that used Alice to expose students to programming concepts prior to learning Java. Section 4 presents the results of a survey given to students in CS1 at the end of the Fall 2006, Spring 2007, and Fall 2007 semesters. Section 5 presents the conclusions.

**BACKGROUND**

The Alice programming environment was first created as a rapid prototyping tool for building virtual reality (VR) applications [7]. Over the course of several years, the developers improved the Alice interface through detailed user studies that made programming with Alice easier for novice users. For instance, whenever possible commands are animated, and instead of using the traditional coordinate axes X, Y, and Z, Alice was modified to use more intuitive “object-centric” directional names such as Forward/Back, Left/Right, and Up/Down [8]. The potential for Alice as a tool to teach beginning programming students was quickly realized. The graphical nature of Alice programs was believed to be highly desirable to novice programmers for many reasons, including the visualization of program state, the ability to watch what goes wrong in a program as it runs, and the intrinsic motivation that 3D graphics provides over text based applications [9]-[10]-[11]. Beginning college level programming courses designed around Alice soon followed and included curriculum on topics such as classes and objects, methods, decision structures, repetition, events, and collections [10]. Published reports have appeared further describing the benefits of the Alice programming environment for teaching recursion [12] and object-oriented programming concepts [13]-[14].

Alice has also been introduced in a variety of alternative formats. These approaches include a course to teach non-majors Computer Science and programming concepts through animation and virtual worlds [15], a CS0 course intended to better prepare students for CS1 [16], the first course in a Computer Information Systems major [17], and a three-week introduction to computer programming as part of a general engineering seminar [18]. Alice has even been used to teach middle and high school students to program in summer course settings [11, 19]. The program visualization aspect and the syntax errorless drag-and-drop interface of Alice make it possible to cover numerous programming topics in a reduced time frame. These qualities seem to make Alice an ideal choice for introducing students with little or no prior experience to computer programming.

Current research with Alice has focused on motivating females to learn computer programming through storytelling. In a study comparing Alice to a special version called “Storytelling Alice”, middle school girls were found to spend more time programming and expressed a stronger interest in future programming activities when first exposed to Storytelling Alice as opposed to regular Alice [20, 21]. It is hoped that a motivational activity, such as storytelling, and an appropriate tool, such as Storytelling Alice, may be a possible solution to the problem of diminishing interest by females in Computer Science.

**CS1 AT PACIFIC**

Prior to the Fall term of 2006, the Computer Science faculty at the University of the Pacific made the decision to add Alice to the curriculum in the introductory programming course. This decision was based largely on concerns about declining enrollments in our department, and the aforementioned advantages of Alice for beginning programmers. It should be noted that this course is the first in our major program, and would thus be classified as CS1. However, since CS1 is a terminal computer programming course for some students outside of Computer Science, we felt it inappropriate to cover only Alice. Thus, we designed a curriculum that focused on Alice for the first half of the term (about seven weeks), followed by Java for the remainder. We used this format for the course for the 2006-2007 academic year. Due largely to the results of the surveys discussed in the next section, we reduced the Alice coverage to about four weeks for the Fall 2007 semester by combining many lectures into a single class period. The content for the Alice portion of the course was modeled after the online curriculum at: www.aliceprogramming.net, which accompanies the text *Learning to Program with Alice* [22].

**SURVEY RESPONSES**

During the last week of the term students were asked to complete a survey that asked about their opinions of Alice. The primary objective of the survey was to help our department determine if we should continue to use Alice in the course. A total of 84 students completed the surveys across the three terms (16 in the Fall of 2006, 25 in the Spring of 2007, and 43 in the Fall of 2007). There were 67 males, 16 females, and one student who chose not to indicate gender. Forty-four students were freshmen, 21 were sophomores, 15 were juniors, and 4 were seniors. Fifty-five students reported having no prior programming experience, 19 had prior programming experience, and 10 chose not to
respond to the question. The remaining survey questions are listed below, and the students’ responses are discussed in the following subsections.

1) What was your favorite thing about Alice?
2) What was your least favorite thing about Alice?
3) Did you like having open ended assignments in Alice that allowed you to design your own project? Why or why not?
4) Do you have any recommendations for improving the Alice portion of the course?
5) Did you think that learning Alice first made it easier for you to learn Java later in the term? Why or why not?
6) Would you recommend for the Computer Science department to continue using Alice in (CS1)? Why or why not?

1. Favorite Things about Alice

The purpose of the first question was to learn what elements of Alice the students enjoyed most. Not surprisingly, the top responses were the ease of programming in Alice and the visualization of code that Alice provides. Thirty-three students responded with answers similar to these. Some student comments that are representative of many others include:

“Ease of use and easy to understand”

“It’s very visual. Helps to see the actions happening and construct a correct set of code. Easy to learn and find your way around.”

“The graphics made it easy to see what we were learning. It allowed me to learn concepts without making syntax errors.”

“The fact that writing the program was very easy. All the code was pretty much done, all we had to do was drag and drop the lines together. It made learning methods easier.”

“I liked that we could actually see what we were doing through the characters. If we programmed a character to move right, we could see that he did move right.”

Some of the other responses included the lack of syntax errors, the open ended aspect of the projects, the graphical 3D game-like nature of Alice programs, learning the basic structure of object-oriented programs, and just simply that Alice was fun. Five students stated that they did not like anything about Alice. Interestingly, none of these students had any prior programming experience.

2. Least Favorite Things about Alice

In contrast to question one, the second question was designed to learn more about student dislikes in order to improve those aspects of the course. However, the most common response (given by 14 students) was concern over Alice implementation issues such as load times and the amount of system memory required by Alice. Some of the student comments were:

“It was very laggy on my home computer”

“Takes a lot of system resources. Slow! Way slow.”

The second most common response (given by 11 students) was the amount of time required to make objects do what seemed to be simplistic tasks. Some representative comments include:

“It is confusing. Takes more time to make the characters do what you want them to {than} it does to learn the concepts.”

“It was very tedious trying to do seemingly simple things like making a person raise their hand.”

“Tons of guesswork!!! More time was spent guessing things {and} distances than learning to code. I’d rather not use it again.”

Some other common concerns that were raised by more than one student included the trial and error approach to programming that Alice seems to encourage, not enough programming control over the application, and the fact that Alice was not a “regular” language. Seven students either listed no response to the question or stated that they had no dislikes.

3. Open Ended Projects

The online materials that accompany the Learning to Program with Alice text suggest assignments that are open ended in nature allowing students to be creative on their projects. Fifty-three students did answer affirmatively that they liked the open-ended assignments and gave reasons similar to the following:

“It really encouraged creative thoughts and allowed me to apply what I had learned to something that interested me.”

“I liked it very much. Being able to essentially create my own game was very satisfying and helped me to put the concepts we learned into action.”

“Yes, because you were more likely to be working on something that interested you. The only downside was it was often hard to determine how involved or how much time your individual program would take.”

A substantial number of students (31), however, were either neutral or negative with regards to the open-ended assignments. Many expressed concerns similar to that raised in the previous comment; that it was very difficult to estimate how much time a particular project idea may take to implement. Others made comments such as:

“No, because I never know what to do”

“Not really because I was never really clear on the expectations of the assignments and ended up getting bad grades.”

“I like to know what I need to do to get an A rather than feeling like I could work on it forever and it maybe not be what you want.”

The concerns raised by the students seem justified and deserve more consideration. Perhaps the solution is to offer
students a choice on each assignment: some options that are open-ended, and others that are clearly specified. This approach is in part the focus of a separate research effort.

4. Recommendations for Improvements

In question four, students were asked if they had any specific suggestions to improve the Alice portion of the course. Thirty-two either left the question blank or said “no”. The most common response from students who did have suggestions was to reduce the amount of time spent on Alice (17 students), with comments such as:

“Make the time allotted to Alice shorter, focus more on Java.”

“I thought it should be shorter, a lot of it I didn’t find that relevant to the second part of the class.”

“I don’t think it needs to be as long. I prefer Java Eclipse programming to the Alice. Its easier to me”

Seven students suggested that Alice be removed from the curriculum altogether. One such comment was:

“Don’t use it. I would have valued more time/slower pace on Java than rushing through both Alice and Java. I feel like Alice was a waste of time and took away time from more valuable material.”

Other suggestions given by more than one student included “… explain a little more in depth about the terminology”, and have “more time in class to ask questions about projects and even more group homeworks and projects.”

5. From Alice to Java

Perhaps the most important question, at least for the theme of this paper, was number five. Students were asked specifically whether or not they felt that their Alice experience helped them to learn Java later in the term. Fifty students (59.5%) answered “yes”, and provided comments such as:

“For someone with no experience it made learning the basics a whole lot easier.”

“Concepts were easy to follow and terms were gradually memorized without a lot of traditional studying.”

“Yes, because it familiarized you with basic concepts and you didn’t have to deal with so many of the details (i.e. syntax) all at once.”

“Yes, since Java isn’t as visual, it was nice to visually see the effects of comp. programming before just writing code blindly.”

Twenty-five students did not feel that Alice helped them to learn Java. Their comments included:

“The frustrations of working with Alice made me oblivious to the actual Java connection.”

“No, because all of the stuff in Java is different and we had to learn it again anyways, and learning Alice took time away from fully learning Java.”

“No, because the programming concepts it taught were mostly so simplistic that it really would have been better to spend only a little time on them and the more complex concepts didn’t make sense until I learned them in Java.”

“No. Alice was all about dragging and dropping lines of code, and it wasn’t until I learned Java, that I actually understood what the code is and what it does.”

Nine students were undecided or did not express a definite opinion either way. Some of the statements provided by these students were:

“It’s difficult to say. I think the time spent on Alice could be better spent on Java. I also think I learned more about object-oriented programming from Java than from Alice.”

“Some of the terms you associate with visual objects in Alice can be confused when translated to more non-visual Java.”

The fact that 34 students were not convinced that prior experience with Alice actually helped them to learn Java is of concern. It is difficult to determine what percentage of students in a course needs to feel that an experience is beneficial to justify its continued inclusion.

6. Should We Keep Using Alice?

Question six asked the students specifically if they thought the department should continue to teach Alice as part of CS1. Fifty-six students (66.7%) answered “yes” to this question, 23 answered “no”, and five were undecided or noncommittal. Some of the comments in support of continued use of Alice were:

“Yes, it seemed to help students a lot who never had any programming experience before.”

“Yes, since I had no computer programming experience, it was a very good introduction. Alice made the transition to Java a lot easier.”

“Absolutely, it transitions to Java much better than other languages. My experience with beginning programming was very much enjoyable with Alice.”

However, four who responded yes also added comments such as:

“Yes, because it is a good introduction, but maybe spend a little less time.”

“Yes, but spend more time on Java as it is like learning a new language.”

Those who did not feel we should continue to teach Alice made statements which included:

“No, more can be learned about Java. Alice is fun and helps, but time can be spent learning other things.”
“No, it is a waste of time and not really applicable to learning Java.”

“No, because I think that Java is a lot more fun and if there was no Alice we could be learning more things about Java.”

Interestingly, more students thought we should continue to teach Alice (56) than thought it helped them to learn Java (50). Perhaps it is the case that some students felt that while Alice did not help them learn new concepts personally, they believed it was useful for the other students in the course.

CONCLUSIONS

A statistical analysis was performed to determine if there was a correlation between gender, year in college, or prior programming experience, to the responses on questions three, five, and six from the survey. No significant correlations were revealed (p > 0.05) by this analysis.

Due largely to the results of question five, our department has decided to discontinue use of Alice in CS1. We were unable to determine whether or not Alice inclusion in CS1 improved the course and we were very concerned that 34 of 84 students (40.5%) were not convinced that Alice was helping them to learn Java. Additionally, 23 of 84 (27.4%) did not feel that we should continue to use Alice in the course. From the perspective of an instructor, incorporating Alice material into the curriculum made the course more difficult to teach. Thus, given these factors, our department decided to use C++ in CS1 for the Spring 2008 term.

The purpose of this paper is not, however, to convey the message that Alice usage is inappropriate in Computer Science courses. Rather, the purpose is to caution faculty members that careful thought needs to be given to the course and manner in which Alice is introduced. Alice has been shown to help students with little prior programming background to be successful in CS1 [2]. This is certainly an important finding for the CS community. Perhaps the difference between that study and what is reported in this paper is the context and focus of the courses. When students are taught Alice and another language in the same course, possibly the goal of the course is unclear to students. Multiple students in our study made comments on the survey similar to the following: “…learning Alice took time away from fully learning Java.” It seems that at least some students felt that the main course objective was to learn Java and that Alice was only a distraction from that objective. However, when Alice is the only language introduced in the course, perhaps it is the case that students are able to focus more on the programming concepts that Alice introduces and are not distracted with the time being taken away from learning a second language later.

The much anticipated Alice v3.0, due for release in 2008, may reduce many of the concerns raised by students in the survey responses. The combination of high quality graphics and an integrated Java development environment that will actually allow for typing and editing code may make the transition from Alice to Java much more seamless for students. This may also eliminate student concerns that Alice is not a real language and that it does not relate well to learning Java. In the future it would be helpful to repeat the study described in this paper for a CS1 course that was taught using Alice v3.0 and Java.

ACKNOWLEDGMENT

The author would like to thank the students who took the time to complete the Alice survey and provide thoughtful comments.

REFERENCES


