Storytelling Alice Motivates Middle School Girls to Learn Computer Programming

Caitlin Kelleher, Randy Pausch, Sara Kiesler
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
(draft copy of paper submitted to CHI 2007)

ABSTRACT
We describe Storytelling Alice, a programming environment that introduces middle school girls to computer programming as a means to the end of creating 3D animated stories. Storytelling Alice supports story creation by providing 1) a set of high-level animations, that support the use of social characters who can interact with one another, 2) a collection of 3D characters and scenery designed to spark story ideas, and 3) a tutorial that introduces users to writing Alice programs using story-based examples. In a study comparing girls' experiences learning to program using Storytelling Alice and a version of Alice without storytelling support (Generic Alice), we found that users of Storytelling Alice and Generic Alice were equally successful at learning basic programming constructs. Participants found Storytelling Alice and Generic Alice equally easy to use and entertaining. Users of Storytelling Alice were more motivated to program; they spent 42% more time programming, were more than 3 times as likely to sneak extra time to work on their programs, and expressed stronger interest in future use of Alice than users of Generic Alice.

Author Keywords
Alice, gender, children, motivation, programming, computer science education, programming environments

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION

Computers and computer-based technologies touch the lives of a broad spectrum of our society and enable advances in areas as diverse as education, medicine, and basic science. With the broad impact of computer science comes the responsibility to ensure that the technologies our society creates meet the needs of all of its members. One of the easiest ways to ensure that our technologies meet the needs of our society is to involve a representative sample of the population in the creation of new technologies. While there are many important and under-represented groups in computer science, women are arguably the largest [21]. In 2005, 85.3% of computer science and computer engineering degrees were awarded to men [22].

Researchers have identified a variety of factors that may contribute to girls' low enrollments in computer science including disinterest in computers, concerns about the computing culture, lack of encouragement from peers, parents, and educators, and relatively fewer opportunities to interact with computers [3, 5]. It is likely that many of these factors play some part in girls' decisions not to pursue computer science. It may be difficult to correct some of the cultural factors that influence girls' decisions not to pursue computer science, but we can work towards making the process of learning to program, which is often a gateway to the study of computer science, more motivating for girls.

Many girls decide whether or not to seriously pursue the study of math and science based disciplines during their middle school years [2]. Although many girls express interest in science during their elementary school years, they have increasingly negative views of science, science classes, and science-based careers as they progress through middle and high school [2]. By late high school many girls have already opted out of the math and science classes that would enable them to pursue a mathematical or scientific major in college [2]. To maximize our potential impact on the number of girls who choose to pursue computer science, we chose to focus on creating a programming environment that makes the process of learning to program more motivating for middle school girls who are old enough to handle the complexity of computer programming but are less likely to have already decided against pursuing computer science.

To make the process of learning to program more motivating for middle school girls, we chose to create a programming environment that presents programming as a means to the end of creating Pixar or Dreamworks-style animated 3D movies (i.e. storytelling). We chose to focus on storytelling for the following reasons:

1. Given a little bit of time, most girls can come up with an idea for a story they would like to create.
2. Stories are naturally sequential and are unlikely to require advanced programming concepts immediately, making them a good match for beginning students.
3. Stories are a form of self-expression and provide girls an opportunity to experiment with different roles, an important activity during adolescence.
4. Non-programming friends can readily understand and appreciate an animated story, which provides an opportunity for girls to get positive feedback from their friends.

In this paper, we describe Storytelling Alice, a programming system based around the activity of storytelling. In a study comparing girls' behavior and interest using Storytelling Alice with a version of Alice without storytelling support (Generic Alice) we found that participants who used Storytelling Alice and Generic Alice were equally successful at learning basic programming constructs. Participants found both versions equally easy to use and entertaining. However, participants who used Storytelling Alice were more motivated to program; they spent 42% more time programming, were more than 3 times as likely to sneak extra time to work on their programs, and were more interested in future use of Alice than participants who used Generic Alice.

INCREASING THE POOL OF PROGRAMMERS

There is a long and rich history of research on designing programming languages and environments to broaden the pool of people who learn to program computers. The majority have focused on simplifying the process of learning to program using a variety of techniques from improving textual programming languages to creating environments that allow users to author programs without making syntax errors [10]. A smaller number of systems try to provide a motivating context for learning program.

A few systems have attempted to motivate computer programming through competitions: in AlgoArena[8] users program sumo-wrestlers to fight tournaments; in Robocode[15] users program battletanks for a "fight to the finish", in Rapunsel [4] users program competitive dancers.

Other systems enable users to create animations and video games. Although it is possible to create animated stories in these systems, most provide users only general graphics capabilities. It may be difficult to create stories from basic graphics commands because of the number of steps required. Several systems allow users to create 2D animations by moving sprites and changing the graphical image associated with the sprite. Stagecast Creator enables users to create 2D games and simulations by specifying graphical before and after conditions[19]. HANDS allows children to create 2D games using a programming language designed to more closely match the ways in which non-programmers describe the solutions to programming problems[16]. In Toontalk, users can create 2D video games through demonstrating their programs in an animated 3D virtual environment [7]. Scratch [13] and Squeak EToys[9] enable children to create 2D animations and games through dragging and dropping graphical tiles. StarLogo TNG allows users to create games and simulations by moving 3D models in a virtual world; animations that require a character to move body parts such as walking or waving hello must be created using external software [20]. Alice 2[1] enables users to move and rotate 3D objects and parts of 3D object in a virtual world through a drag and drop based program editor.

STORYTELLING ALICE

Storytelling Alice is based on Alice 2, an open-source programming environment that helps overcome two difficulties beginning programmers often encounter: syntax errors and invisible state. Users construct programs by dragging and dropping code elements, which removes the possibility for making syntax errors. Running Alice programs are animated, which enables users to watch their programs execute and to see their mistakes. NSF-sponsored studies have shown that exposure to Alice 2 increases the academic success and retention of at-risk computer science majors (students who have no programming experience and/or who are not prepared to enroll in Calculus as freshmen) in introductory computer science courses [14].

The development of Storytelling Alice was guided by formative testing with more than 200 girls over a two-year period. The formative testing took place in a variety of formats ranging from 4 hour afternoon workshops to week-long camps with groups of 3 to 20 girls ranging in age from 10 to 17. The girls were recruited from technology camps, home-schooling groups, and the Girl Scouts. During formative testing, girls created storyboards of movies they wanted to create and then tried to implement them in a version of Storytelling Alice[10]. Storytelling Alice includes three types of supports to enable users to create stories: 1) high-level animations that support the use of social characters who can interact with one another 2) a gallery of characters and scene elements that helps girls find story ideas, and 3) a story-based tutorial.

High-Level Animations and Social Characters

When girls are asked to plan an animated movie they want to create, their movies consist almost entirely of people and characters moving around different settings, and speaking and interacting with each other [10]. A typical story might include a scene like the following:

A girl named Susie walks over to a group of more popular girls and invites them to a social event. The popular girls say something mean to Susie. Susie turns away from the popular girls and covers her eyes.

In Generic Alice, scenes like the one described above can be both tedious and difficult to create. To create a basic walk animation requires that users individually rotate a character's hips, knees, and ankles. In formative testing, we found that for many girls, animating stories using basic transformations like move and turn is both uninteresting and frustrating. Storytelling Alice includes a set of higher-level animations based on an analysis of storyboards girls created [10]. Using these high-level animations, scenes like the one described above are more readily attainable.

Generic Alice methods:
move, turn, roll, resize, play sound, move to, move toward, move away from, orient to, point at, set point of view to, set pose, move at speed, turn at speed, roll at speed
Storytelling Alice methods:
say, think, play sound, walk to, walk offscreen, walk, move, sit on, lie on, kneel, fall down, stand up, straighten, look at, look, turn to face, turn away from, turn, touch, keep touching

Figure 1: a comparison of the methods a person can perform in Generic Alice and Storytelling Alice.

In addition to the need for high-level animations, many of the stories that girls imagine creating require multiple settings (or scenes). Storytelling Alice also includes support to enable to create multi-scene stories.

Storytelling Gallery

One factor that we found influences girls' motivation to learn to program in Storytelling Alice and their perseverance when they encounter problems is their ability to find a story that they want to tell. User testing revealed that animations requiring an explanation within the story can prompt girls to generate story ideas. An early example of the potential for animations to inspire stories came through a robot character who had an animation entitled "crazy go nuts." To explain why the robot went crazy, girls created stories about topics ranging from parental troubles to the difficulties of being unpopular to failing a test. In the Storytelling Gallery, each character comes with at least four

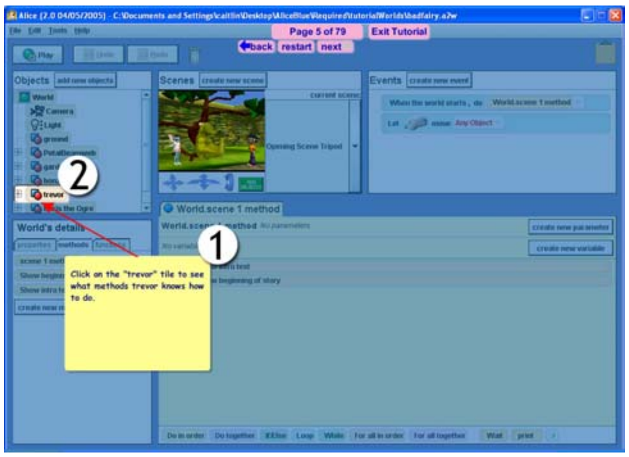
Figure 2: Harold T. Wireton's "crazy go nuts" animation inspired a broad range of stories.

animations that are unique to that character. Many of the character-specific animations require explanation to make sense within the story.

In addition to animations requiring explanation within the story, we found that characters with clear roles can also be helpful in generating stories: a teacher or a lunch lady is nearly always an authority figure and a ninja character is frequently a hero or a villain.

Story-based Tutorial

Initially, the Alice tutorial was designed around examples chosen to demonstrate concepts as simply as possible. However, user testing revealed a need to introduce users to programming in Alice within the context of stories similar to the ones they imagined creating. Story-based examples, while more motivating, also tend to be more complex. In one of the tutorials, the user constructs a story about a trouble-making fairy who casts a spell on a boy, causing the boy to fall in love with an ogre. While we found that stories like the example above tend to be more motivating for girls, they add complexity both to the user interface and to the steps that users need to perform to complete the tutorial.



Click on the "trevor" tile to see what methods trevor knows how to do.

Figure 3: The Alice interface with a Stencils-based tutorial: 1) a sticky note that provides instructions for the current step and 2) a hole in the Stencil that allows the user to interact with the interface component beneath it.

To moderate the additional complexity of a story-based tutorial, Storytelling Alice presents the tutorial using Stencils [12], an interaction technique in which a transparent blue screen that catches mouse and keyboard events is drawn over existing interface. The instructions for each step in the tutorial are displayed on "sticky" notes that are drawn on top of the Stencil. Holes cut over interface elements necessary for the current step allow users to complete actions for the current step.

METHOD

We conducted a between subjects study comparing the learning, behavior, and attitude of girls introduced to programming using Storytelling Alice and Generic Alice. The study took as series of one-time, four hour workshops.

Participants

A total of 88 girls from local Girl Scout troops participated in the evaluation of Storytelling Alice; 45 were assigned to the control group and used primarily Generic Alice and 43 were assigned to the experimental group and used primarily Storytelling Alice. The average age for the participants was 12.6 years (12.8 years in the control group and 12.5 in the experimental group) and nearly all participants were in grades 5-9, with the majority in the 7th and 8th grades. Overall, 76 participants reported attending public school and 12 (7 in the control group and 5 in the experimental group) attend private school. To encourage broad participation, we offered a $10 donation to the Girl Scout troop for each girl who participated. The majority of the Girl Scout troops used our study as a fundraiser.

Workshop Details

Participants had two hours and fifteen minutes to complete the tutorial (30-45 minutes for most participants) and create a program (during the remaining 90-105 minutes) using the version of Alice to which they were assigned. After two hours and fifteen minutes, participants took a programming quiz and completed a survey. Then, participants were given 30 minutes to try the version of Alice to which they were not assigned (participants assigned to Generic Alice used Storytelling Alice and vice versa). At the end of the workshop, participants were asked to select either Storytelling or Generic Alice to take home. Finally, participants were asked to select one of the Alice programs they created to share with others.

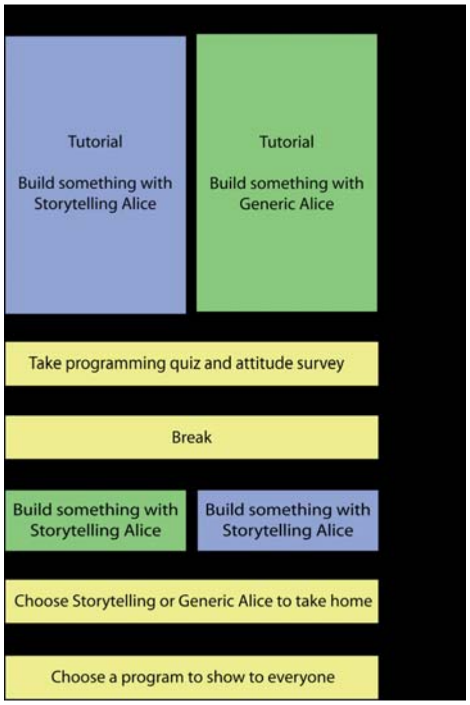

Tutorial
Build something with Storytelling Alice
Tutorial
Build something with Generic Alice
Take programming quiz and attitude survey
Break
Build something with Storytelling Alice
Build something with Storytelling Alice
Choose Storytelling or Generic Alice to take home
Choose a program to show to everyone

Figure 4: Procedure for experimental and control groups.

To avoid prematurely exposing participants in one condition to the version of Alice they were not learning, computers for Storytelling Alice and Generic Alice were set up in different parts of the room so that participants could not see what the screens of participants in the other condition. We instructed participants that they could talk freely to other participants in their own condition, but they could not talk with participants in the other condition. All verbal instructions were given to both groups.

Participants were randomly assigned to either the control group (using Generic Alice) or the experimental group (using Storytelling Alice). To avoid biasing participants based on the names Storytelling Alice and Generic Alice, we referred to Storytelling Alice as Alice Blue and Generic Alice as Alice Green.

Measures

Our measures included program archives, a programming quiz, an attitude survey, and observations of participants' behavior.

Alice Programs

We collected the programs that participants created to get a qualitative picture of the kinds of animations (e.g. stories, artistic animations, random motion, etc) participants in both conditions created.

Programming Quiz

After completing the post-survey, participants took a seven item forced-choice programming quiz that asked them to predict the behavior of short Alice programs. Each question showed a short segment of code in Alice and participants were asked to select the appropriate description of the program's behavior from four choices. The programming quiz contained questions about sequential programming, events, parallel execution, loops, method calls, and parameters.

Based on an exploratory factor analysis of the programming quiz responses, we created a programming quiz scale that included the six questions that loaded on the same factor. This factor reflected participants' understanding of programming structures like loops, do togethers, and method calls (Cronbach's $\alpha = 0.74$). The remaining question on the quiz tested users' understanding of basic events, a topic the Storytelling Alice participants who created multiple scenes were more likely to encounter.

Alice Logs

We instrumented both Storytelling Alice and Generic Alice to record all of the actions that users took within the program. These logs include both programming activities (e.g. adding, deleting, moving, or modifying a line of code, creating a method, adding a loop) and non-programming activities (e.g. adding, deleting, or positioning characters or objects within the 3D scene).

Attitude Survey

The attitude survey focused on participants' experience with their assigned version of Alice. It included nine statements about participants' experience during the study and opinions about their assigned version of Alice. The survey also included 8 questions about participants' interest in using Alice in the future and their interest in pursuing computer science. They were asked to indicate whether they agreed or disagreed with the statements and answered the questions using a five-point Likert scale.

Based on exploratory factor analysis, we created four scales for the survey data: Alice's ease of use, Alice's entertainment value, participants' interest in future Alice use, and their interest in computer science. Example questions and the Cronbach's alpha for each scale are listed in Table 1.

Scale	Example Question	Cronbach's Alpha
Alice's ease of use (1= strongly disagree, 5 strongly agree)	The computer animation program I used today is confusing.	0.63
Alice's entertainment value (1= strongly agree, 5 = strongly disagree	Using the computer during the workshop today was fun.	0.86
Users' Future Alice Interest (1 =definitely not, 5 = definitely yes)	Would you be interested in taking another Alice class?	0.83
Users' Computer Science Interest (1 = definitely not, 5 = definitely yes)	Would you be interested in taking a computer science class in high school?	0.8

Table 1: Example questions and Cronbach's alpha for each of the four attitude scales.

Behavioral Measures

We created several opportunities during the workshop for participants to express an interest in Alice through their actions. We tracked which version of Alice users chose to take home, which program they selected to share with their peers, and whether or not they snuck extra time to continue working on their programs when there was no instruction or requirement that they be interacting with Alice.

RESULTS

By focusing on the motivational aspects of an educational software system, there is some risk that changes made to increase student motivation may reduce the educational value of interacting with the system. In this study, there is no evidence of negative impact. That is, there were no significant differences between participants who used Storytelling Alice and Generic Alice on either the programming structures questions or the world starts event question. The focus on storytelling did not negatively impact participants' learning of programming concepts.

To provide insight into how motivating participants found Storytelling Alice and Generic Alice, our measures provide three different lenses: what participants created within their assigned version of Alice, what participants said about their experience with and interest in their assigned version of Alice, and what participants did within the study (i.e. behavioral measures).

What Girls Created

Participants in the Generic Alice and Storytelling Alice conditions created different kinds of programs.

Generic Alice Programs

We observed that the programs participants created with Generic Alice were of four general types: arbitrary motion, character motion, story-like sequences, and choreographed dance routines.

Arbitrary Motion (62%):

28 of the 45 worlds that users created with Generic Alice consist of seemingly arbitrary animation: characters and/or their body parts move around the screen without any coherence or clear purpose. The programs in the arbitrary motion category seem to be largely experimental and show little evidence that users had explicit goals they were working towards. Figure 5-1 shows a screen shot from a typical arbitrary motion program. In this world, characters and their body parts rotate around different axes and fly to different positions in space.

Character Motions (16%):

7 of the 45 users created programs which contained one or two simple character motions but were otherwise largely arbitrary motion. Examples of character motions include having a cat swish its tail, a penguin open and close its mouth, and a bunny jump up and down. These worlds seem to the result of users transitioning from experimenting with the Generic Alice animations to combining animations to animate their 3D characters.

Choreographed Dance Routines (7%):

3 of the 45 users created choreographed dance routines for a group of characters. The dance routines made heavy use of move and turn with characters performing the same motions together and in sequence. Dance routines are a good match for Generic Alice; several characters doing simple motions in sequence or in parallel is well supported and can be visually pleasing.

Story-Like Sequences (16%):

7 of the 45 users created short story-like sequences. In most of the story-like sequences, users rely on the physical appearance of characters to identify them as heroes, villains, or victims. Users frequently incorporated simple gestures that help to communicate the action in the story

such as having character raise their arms in fear before quickly sliding away or an injured dragon turning on its side after being stabbed by a knight.

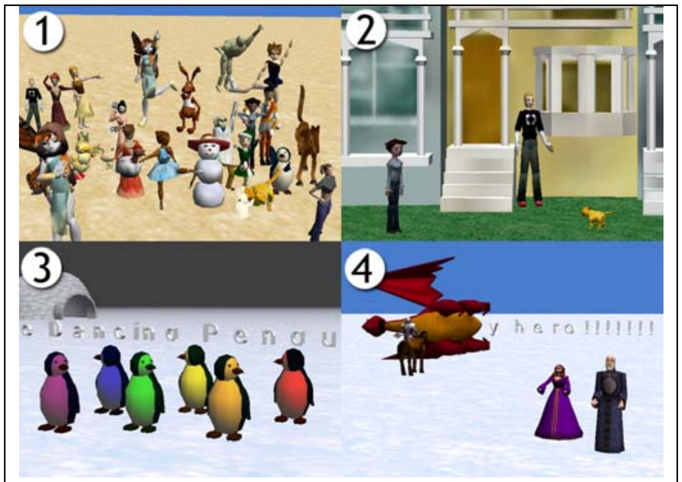


Figure 5: Examples of programs participants created in Generic Alice: 1) a program that is largely arbitrary motion; objects and their parts move around in space, 2) a program containing a character motion; a girl waves hello, 3) a choreographed dance routine for a group of penguins, and 4) a story-like program in which a knight kills a dragon.

Storytelling Alice Programs

We observed that the programs participants created with Storytelling Alice were of three general types: relationship stories, good vs. evil stories, and miscellaneous programs.

Relationship Stories (51%):

22 of the 43 users created stories about relationships, including romantic relationships, peer relationships, and familial relationships. Some participants used their stories to think about and react to issues in their own lives. For example, one story about divorcing parents depicted the children kicking the parents out of the house. While a child would be unlikely to successfully evict their parents, this participant's story provided her with an opportunity to express frustration about her family's situation. Other participants' stories addressed topics like the difficulty of being unpopular and how to handle a crush on a boy.

Good vs. Evil Stories (21%):

9 of the 43 users created stories depicting conflicts between good and evil. The conflict between good and evil provides an easy source of tension, and one that is frequently used in mainstream movies and books. In some stories, evil characters are defeated by more powerful good characters. In one story, an evil samurai attacks an innocent pig. A good magical tree comes to the pig's defense and helps the pig defeat the samurai. In other stories, the evil characters triumph. In one story an evil sheriff wants to take over the world. When his minion expresses doubts, the sheriff disciplines him by tossing him across the room.

Other Programs (28%)

12 of the 43 programs created with Storytelling Alice do not fall neatly into a single category. These miscellaneous worlds include two stories about finding lost dogs, two stories depicting running and swimming races, and three choreographed routines (circus and cheerleading) similar in nature to the choreographed dance routines created by users of Generic Alice.

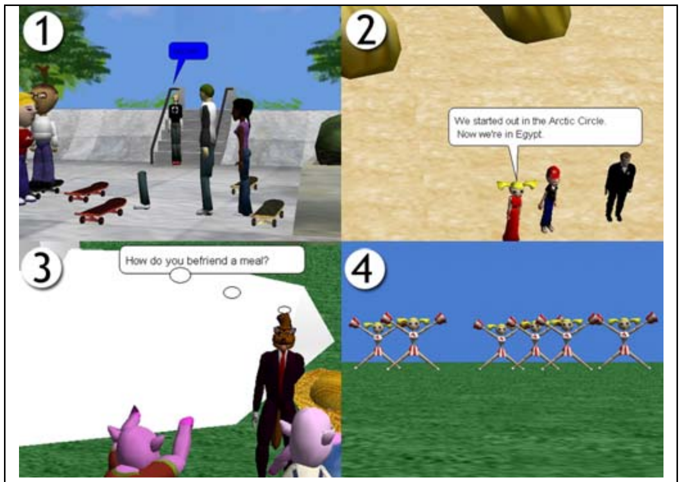

1
2
We started out in the Arctic Circle. Now we're in Egypt.
3
How do you befriend a meal?
4

Figure 6: Examples of programs users created in Storytelling Alice: 1) a romantic relationship story about a boy who is involved with three girls and gets caught, 2) a familial relationship story about a father taking his children on vacation and getting lost, 3) a good vs. evil story about the big bad wolf trying to befriend the three pigs so he can eat them later, and 4) a choreographed cheerleading routine.

Users' Activities within Storytelling and Generic Alice

There are fundamentally three activities that users can perform in either version of Alice: 1) adding and arranging 3D objects in the virtual world (i.e. scene layout), 2) creating animations that control the motions of 3D objects (i.e. programming) and 3) running their programs. We analyzed the differences in the percentage of time participants spent on programming and scene layout using Storytelling Alice and Generic Alice with an unpaired t-test. Overall, participants who used Storytelling Alice spent 42% ($p < .001$) more time editing their programs and 54% ($p < .001$) less time laying out their scenes (see Figure 7).

There is a broad range in how users choose to spend their time in Generic Alice. While some users spend as much as 60% of their time editing the programs (and another 25% of their time running their programs), other users spend nearly all of their time on scene layout (see upper left of Figure 8). Storytelling Alice seemed particularly effective in motivating all participants to spend time programming. 12 of the 45 users of Generic Alice spent more than 50% of their time on scene layout. None of the users of Storytelling Alice spent more than 50% of their time on scene layout.

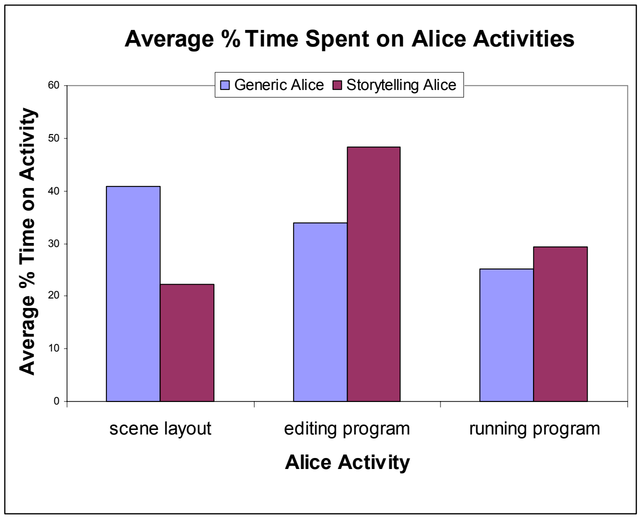


Figure 7: Average percentage of time users of Generic Alice and Storytelling Alice spent on scene layout, program editing, and running their programs.

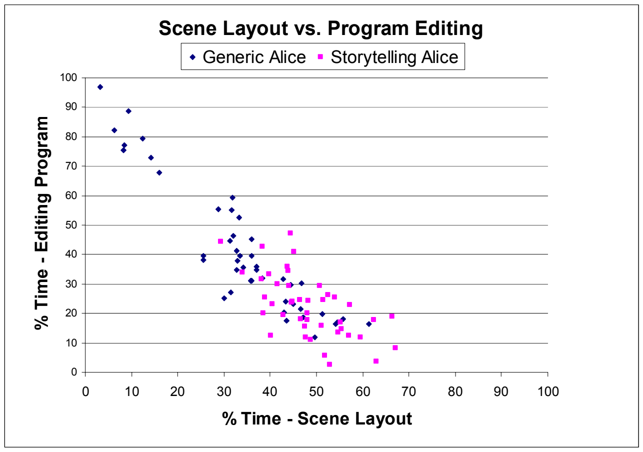


Figure 8: Percentage of time spent on scene layout vs. program editing for participants who used Generic Alice and Storytelling Alice.

Alice version	Percentage of participants who created methods	Percentage of participants who used do togethers	Percentage of participants who used loops
Generic	30	74	33
Storytelling	53	79	12
p-value	p < .05		p < .05

Table 2: Percentage of participants who used methods, do togethers, and loops in their programs.

Users of both Storytelling Alice and Generic Alice experimented with programming constructs beyond simple sequences. A majority of the participants in both groups used Do Togethers to have multiple animations occur simultaneously. Beyond parallelism, users of Storytelling Alice and Generic Alice tended to experiment with different programming constructs. In part because of the need for multiple scenes in stories, users of Storytelling Alice were more likely to create and use new methods. Users of Generic Alice were more likely to experiment with loops.

What Participants Say

We used multivariate analysis to examine the impact of the version of Alice to which participants were assigned, participants' academic performance, and confidence using computers on how participants rated the Alice's ease of use and entertainment value as well as their future interest in Alice and computer science.

Alice's Ease of Use

Users of Generic Alice and Storytelling Alice did not differ significantly in how easy they felt it was to use their version of Alice (p = .90). This is not surprising given that the process of creating a program in Generic Alice and Storytelling Alice is nearly identical.

Alice's Entertainment Value

Users of Generic Alice and Storytelling Alice did not differ significantly in how much they enjoyed using their version of Alice (p = .25). While this may initially seem surprising, it is probably at least partially attributable to participants' enjoyment of selecting and arranging objects in the 3D world. For many participants, the experience of browsing through the gallery, selecting 3D objects, and arranging them in the virtual world was a rewarding experience, although it has little educational benefit. The questions in the entertainment scale focused on the experience of using Alice as a whole, rather than the experience of programming in Alice. Generic Alice and Storytelling Alice may have been entertaining for different reasons.

Users' Interest in Future Alice Use

Participants who used Storytelling had a stronger interest in continuing to use Alice in the future than those who used Generic Alice (F[1,86]=3.9, p=.05). One potential explanation for this is that participants using Storytelling Alice may have felt that Storytelling Alice had greater "replayability" because the space of interesting stories is larger than the space of interesting arrangements of 3D objects. Users of Generic Alice may have been more motivated by scene layout than programming and therefore less interested in continued Alice usage.

Users' Interest in Computer Science

A single four-hour workshop is a fairly short period of time in which to change students' attitudes about and interest in pursuing computer science. Not surprisingly, there was no significant difference in interest in pursuing computer science between users of Generic Alice and Storytelling Alice (p =.33), although users of Storytelling Alice expressed slightly higher interest on most questions.

However there is a strong relationship between participants' interest in using Alice in the future and their interest in pursuing Computer Science ($r = .54$, $p < .0001$).

	Quiz	Ease of Use	Entertainment	Future Alice Use	CS Interest
Quiz	1.00	.05	.14	.23	.19
Ease of Use		1.00	.30*	.26*	.32*
Entertainment			1.00	.77***	.40**
Future Alice Use				1.00	.54***
CS Interest					1.00

Table 3: Correlations among continuous variables ($^{*}p < .05$, $^{**}p < .001$, $^{***}p < .0001$)

Table 3 shows the correlations between participants' scores on the programming quiz and the four attitude scales. There are strong correlations ($p < .0001$) between two pairs of variables: 1) participants' ratings of Alice's entertainment value and their interest in using Alice in the future and 2) participants' interest in using Alice in the future and their interest in computer science. While these are correlations, they do provide some support for our strategy of developing a motivating programming environment to encourage middle school girls to consider computer science.

What Participants Do

We used three behavioral measures to gauge participants' preferred version of Alice and their motivation to program. The behavioral measures were: 1) which version of Alice participants chose to take home 2) which program participants chose to share with their peers and 3) how many participants snuck extra time to continue working on their programs.

Which version of Alice do participants choose to take home?

Because participants had 2 hours and 15 minutes with their main version of Alice and only 30 minutes with the other version, it is reasonable to expect that participants would tend to choose the version of Alice with which they had the most experience. In fact, both groups showed a strong preference for Storytelling Alice ($\chi^2 = 34.12$, d.o.f. = 1, $p < .001$). Of the users assigned to Storytelling Alice, 88% of them elected to take Storytelling Alice home with them. Of the users assigned to Generic Alice, only 26.7% elected to take Generic Alice Home with them. In three cases, there were siblings (totaling six subjects) in the testing groups who colluded to ensure that they had both versions of the system at home. If I remove these pairs from the data, the preference towards Storytelling Alice becomes even stronger: 90% of Storytelling Alice users chose Storytelling Alice and 23.4% of Generic Alice users chose Generic Alice to take home. The fact that participants who used both Generic Alice and Storytelling Alice overwhelmingly chose Storytelling Alice as the system they wanted to take home demonstrates that Storytelling Alice has a stronger appeal than Generic Alice for most participants.

When asked to show a program, what do participants show?

By asking participants to select an Alice program to share with their peers, we are asking them to select the program of which they are the most proud. As with the choice of which system to take home, it seems reasonable to expect that participants would tend to show the Alice program that they had the most time to create. In this case, we do see a tendency in that direction: 98% of the participants using Storytelling Alice showed a world from their assigned version of Alice and 68% of the participants using Generic Alice. However, a surprising 32% of the Generic Alice participants chose to show a world that they created in Storytelling Alice in 30 minutes rather than the world they had approximately 90 minutes to create in Generic Alice ($\chi^2 = 20.18$, d.o.f. = 2, $p < .001$). Only a single Storytelling Alice user (2%) elected to show the program she created in Generic Alice.

How many participants sneak extra time to continue working on their programs?

One way to gauge users' affinity for Alice is to examine the numbers of users who continue programming when there is no expectation that they do so. At the end of the each evaluation session, we left a period of 5-10 minutes during which there was no expectation that users interact with Alice. Users of Storytelling Alice were almost three times as likely to sneak extra time during this period to make final changes to their Alice programs. Among the users of Generic Alice, only 16% of participants made changes to their Alice program before sharing it. Among the users of Storytelling, 51% of users made final changes to their Alice program before sharing it ($\chi^2 = 20.18$, d.o.f. = 2, $p < .001$). The increased tendency among Storytelling Alice users to sneak extra time provides additional evidence that the storytelling focus helped make learning to program more engaging for middle school girls.

DISCUSSION

The results of our study suggest that participants who used Generic Alice and Storytelling Alice were equally successful in learning programming concepts. However, we found that participants who used Storytelling Alice showed more evidence of engagement with programming; they spent a greater percentage of their time programming, were more likely to sneak extra time to continue programming, and expressed greater interest in future use of Alice than participants who used Generic Alice.

Although participants who used Storytelling Alice showed more signs of engagement with programming than participants who used Generic Alice, users found Generic Alice and Storytelling Alice equally entertaining. We believe that users of Generic Alice may have enjoyed their overall experience with Generic Alice in part because they found the process of selecting and laying out 3D objects in the virtual world to be entertaining. The fact that users of Generic Alice spent less time on programming than users of Storytelling Alice may provide some support for this explanation. However, in future studies we would like to tease apart the motivational aspects of the different activities within Alice.

LIMITATIONS

There are limitations to the study we present: the short duration of the study, the nature of our participant pool, and the presence of a highly motivated experimenter.

Limited Time

Although our initial results are encouraging, girls only spent a few hours creating programs in Generic Alice and Storytelling Alice. A longer experiment might generate different results.

Participants

Although we did offer a donation to Girl Scout troops to encourage broad participation and many troops used participation in our study as a fundraiser, participation in our study was voluntary. Storytelling Alice might not be as successful in a typical school setting.

Further, all participants were girls. We have done some informal testing of Storytelling Alice that suggests that the activity of storytelling may also be a motivating context for boys to learn programming. However, there is a risk that Storytelling Alice may not work as well for boys as girls.

Experimenter

A highly motivated experimenter was present at all study workshops, which may have influenced participants' experience. To minimize the impact of the experimenter, the experimenter did no teaching. Participants learned how to write Alice programs through completing the tutorial in their assigned version of the system. The experimenter was available to answer questions, but could not initiate contact with any of the participants. In a typical classroom setting, the teacher would likely take a more active role, which could positively or negatively influence students' experiences using Storytelling Alice.

FUTURE WORK

Although we focused initially on creating a motivating programming environment for middle school girls, our eventual goal is to create a programming environment that provides a positive introduction to computer programming for all students. There is a strong predicted need for computer scientists. The United States Bureau of Labor Statistics predicts that the computer and mathematical sciences are one of three occupational groups that will account for nearly 75% of new jobs in the period between 2004 and 2014 [6]. Further, 65% of job openings in science and engineering between 2004 and 2014 are expected to be in information technology related jobs [6]. Despite the strong need for computer scientists, student interest in studying computer science has dropped dramatically in recent years. In the period between 2000 and 2005, the number of college freshman who listed computer science as their probable major dropped by 70% and computer science enrollments at research universities dropped by 50% [21]. One important direction for future work is to determine the effectiveness of Storytelling Alice at motivating all children, including boys and members of minority groups to learn computer programming.

As we continue to develop Storytelling Alice, we would like to focus on two goals: 1) finding techniques that help to keep users engaged with programming over a longer period of time and 2) finding ways to encourage users to explore and master a wider range of programming concepts.

One potential avenue for extending the length of time that users stay engaged with creating programs in Storytelling Alice draws inspiration from techniques used in interactive games. Games, while not always educational, share the goal of keeping users engaged in an activity for lengthy stretches of time. One interesting area for future research lies in applying the kinds of reward strategies used in games to reinforce users' engagement with computer programming.

Girls' storyboards and the questions they asked during user testing revealed a need for the ability to create film extras (the background characters in movies such as students in a classroom, people walking in a park, or the audience members at a show) for their movies. Modeling the behavior of extras provides an opportunity to introduce conditional and random behavior as well as a context for discussing the use of computing as simulation. Some of the systems for novice programmers have focused on simulation [17, 18, 19, 20], but none within the context of allowing users to create extras for animated movies. Simulations of extras may provide a context that motivates users to learn more advanced programming concepts. Armed with an understanding of more complex programming concepts, users may be able to more directly relate their experiences with Alice to real-world use of computer simulations like predicting earthquakes and searching for new medications. The ability to see a strong connection between creating animated movies in Storytelling Alice and real-world computer science may further reinforce users' interest in pursuing computer science.

CONCLUSION

More than 30 years of research has gone into developing programming systems that open the activity of computer programming to a broader group of people. Yet, relatively few of these systems have formally validated success at drawing a broader group of people into programming, and, to our knowledge, no programming systems have formally demonstrated success at drawing middle school girls into computer programming. Storytelling Alice provides a strong first step towards a programming system that can give girls a positive first experience with computer programming. These positive first experiences with computer programming may help to inspire more girls to pursue computer science and begin to correct the longstanding under-representation of women. As we continue to develop Storytelling Alice, we hope that it will become a motivating way to learn computer programming for all children.

ACKNOWLEDGMENTS
This work was funded by a National Science Foundation grant. The authors wish to thank <anonymized> and <anonymized> for their helpful comments on earlier drafts.

REFERENCES

1. Alice 2.0. http://www.alice.org
2. American Association of University Women, *How Schools Shortchange Girls: A Study of Major Findings on Girls and Education*. Marlowe & Company, New York, NY, USA, 1992.
3. American Association of University Women, *Tech-Savvy: Educating Girls in the New Computer Age*. American University of Women Educational Foundation, Washington, DC, USA, 2000.
4. Flanagan, M., Howe, D. and Nissenbaum, H. Values at play: design tradeoffs in socially-oriented game design. In *Proc. CHI 2005.* ACM Press (2005), 751-760.
5. Furger, R. *Does Jane Compute?: Preserving Our Daughter's Place in the Cyber Revolution.* Warner Books, Inc. New York, NY, USA 1998.
6. Hecker, D. Occupational employment projections to 2014. Monthly Labor Review. November 2005.
7. Kahn, K. Drawings on napkins, video-game animation, and other ways to program computers. *Communications of the ACM 43,3* (1996), 104-106.
8. Kato, H. and Ide, A. Using a Game for Social Setting in a Learning Environment: AlgoArena – A Tool for Learning Software Design. In *Proc. CSCL,* ACM Press (1995), 195-199.
9. Kay, A. Etoys and Simstories in Squeak. Available at: http://www.squeakland.org/author/etoys.html
10. Kelleher, C. and Pausch, R. Lessons Learned from Designing a Programming System to Support Middle School Girls Creating Animated Stories. In *Proc VL/HCC*. IEEE (2006).
11. Kelleher, C. and Pausch, R. Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programming. *ACM Computing Surveys 37, 2* (2005), 83-137.
12. Kelleher, C. and Pausch, R. Stencils-based tutorials: design and evaluation. In *Proc CHI 2005.* ACM Press (2005), 541-550.
13. Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., and Resnick, M. Scratch: A Sneak Preview. In *Proc of Creating, Connecting, and Collaborating through Computing* (2004). 104-109.
14. Moskal, B., D. Lurie, et al. *Evaluating the Effectiveness of a New Instructional Approach.* In *Proc SIGCSE 2004.* ACM Press (2004), 75-79.
15. Nelson, M. Robocode. IBM Advanced Technologies, 2001.
16. Pane, J. Myers, B.A., and Miller, L.B. Using HCI Techniques to Design a More Usable Programming System. In *Proc. HCC 2002,* IEEE (2002), 198-206.
17. Repenning, A. Agentsheets: a tool for building domain-oriented visual programming environments. In *Extended Abstracts CHI 1993*. ACM Press (1993), 142-143.
18. Resnick, M. StarLogo: an environment for decentralized modeling and decentralized thinking. In *Extended Abstracts CHI 1996.* ACM Press (1996), 11-12.
19. Smith, D., Cypher, A., and Tesler, L. Programming by example: novice programming comes of age. *Communications of the ACM 43, 3* (2000), 75-81.
20. StarLogo TNG. http://education.mit.edu/starlogo-tng/index.htm
21. Vegso, J. Drop in CS Bachelor's Degree Production. *Computing Research News 18, 2* (2006).
22. Zweben, S. Ph.D. Production at an All-Time High with More New Graduates Going Abroad; Undergraduate Enrollments Again Drop Significantly. *Computing Research News 18, 3* (2006).