# Storytelling Alice Motivates Middle School Girls to Learn Computer Programming

**Caitlin Kelleher, Randy Pausch, and Sara Kiesler**
Human Computer Interaction Institute
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
caitlin@cs.cmu.edu, pausch@cmu.edu, kiesler@cs.cmu.edu

## 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 [36].

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 [2, 13]. 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 [14, 37]. 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 [1, 37]. Girls' confidence in their abilities to succeed in math and science [1, 7] and their achievement in these subjects [9] drop during middle school. 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


*CHI 2007,* April 28–May 3, 2007, San Jose, California, USA.

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 positive feedback.

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.

## PROGRAMMING SYSTEMS FOR CHILDREN

There is a long history of research on designing programming languages and environments for novice programmers of all ages [21]. Among the programming systems designed for children, there are three broad motivations for introducing children to programming: 1) as a tool that enables children to explore ideas, 2) as a medium for self-expression, and 3) as a stepping stone into computing-based careers.

Many of the programming systems for exploring ideas are attempting to provide children with computational spaces in which to explore and refine their own thinking. Logo[31] and more recently EToys[19] were designed to enable children to explore ideas in music, language, and mathematics. Playground allows children to model virtual organisms [10]. Starlogo was designed to allow children to simulate the behavior of large systems of objects or organisms such as flocks of birds [32]. Logoblocks (the precursor to Lego Mindstorms) enables children program computationally-augmented Lego bricks to sense and affect the physical world [4].

A second group of programming systems explore the use of programming for self expression. Play [35], My Make Believe Castle [25], and Magic Forest[26] focus on enabling users to create simple 2D animations. Hands[30], ToonTalk[15], Klik N Play[24], StarLogo TNG[34], Scratch[27], and Stagecast[33] focus on enabling users to create their own games. MOOSE Crossing enables users to create characters and spaces to populate a multi-user networked text-based virtual world [5].

Several programming systems are designed to introduce students to programming as a pathway into computer science. Virtual Family introduces Java programming through enabling users to create story-scripts for a family of four characters [8]. Using Robocode [29] students program the behavior of virtual tanks in Java to enter into competitions. In RAPUNSEL, students program dance-moves using a specialized Java editor[12].

While children can create animated stories in several systems [19, 26, 27, 33, 35], these systems were designed to enable general animation and do not provide support targeted at storytelling.

Programming systems for children employ a variety of techniques to ease the process of learning to program for children. These include: simplifying the programming language syntax [5, 31], designing programming languages that more closely match the way that children describe the behavior of programs [30], enabling users to construct programs by assembling graphical tiles [4, 19, 27, 34] or filling in forms [24, 11, 25], specifying programs as graphical rules [33], and creating programs by demonstrating actions in a 3D animated world [15].

### Girls and Programming

One of the primary challenges in increasing girls' participation in computer programming is motivating girls to learn to program. Several studies of children programmers have found that when girls and boys have similar experience with computer programming they are equally interested in and effective at learning to program [15, 17, 23]. Yet, few girls choose to program. In a study of gender and programming achievement within MOOSE Crossing, Bruckman et al. found that gender does not affect programming performance [6]. Instead, programming performance correlated with the amount of time users spent programming and their prior programming experience. Boys who used MOOSE Crossing chose to spend more time on programming than girls [6]. One of the keys to increasing the participation of women in computer science may be motivating more girls to learn to program.

Two systems focus on introducing programming within contexts that may be more motivating for girls: Virtual Family [8] (stories about a family) and RAPUNSEL[12] (dance animations). To the best of our knowledge, no formal studies demonstrate that either system motivates girls to program.

## STORYTELLING ALICE

Storytelling Alice is based on Alice 2, an open-source programming environment that helps users overcome two difficulties beginning programmers often encounter: syntax errors and invisible state [3]. 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. Further, use of Alice 2 at the college level helped students to succeed in later computer science courses taught in Java [28].

The development of Storytelling Alice was guided by formative evaluation and usability testing with more than 200 girls over a two-year period [20, 22]. The 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[20]. 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

Based on our formative testing, when girls are asked to plan an animated movie they want to create, they frequently create movies in which humanoid characters move around different settings, speak, and interact with each other [20]. 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.*

| **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.**

Like Hoyles et al. [16], we found that providing programmatic building blocks at an appropriate level was critical. 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 high-level animations based on an analysis of storyboards girls created and usability testing with our target audience [20]. Using these high-level animations, scenes like the one described above are more readily attainable, but users are still motivated to explore a variety of programming constructs.

Many of the stories that girls imagine creating require multiple settings (or scenes). Storytelling Alice also includes support the creation of multi-scene stories.

### Storytelling Gallery

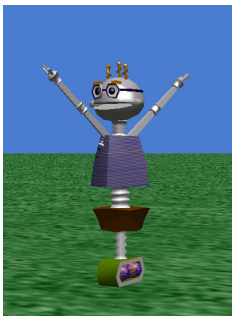

**Figure 2: Harold T. Wireton's "crazy go nuts" animation inspired a broad range of stories.**

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 custom animations. Many of these custom animations require explanation 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. For example, a teacher or a lunch lady is nearly always used as an authority figure.

### Story-based Tutorial

Initially, the Alice tutorial was designed around examples specifically 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.

To moderate the additional complexity of a story-based tutorial, Storytelling Alice presents the tutorial using Stencils [22], an interaction technique which visually guides users to the interface components necessary for the current step and prevents them from interacting with other interface elements (see Figure 3).

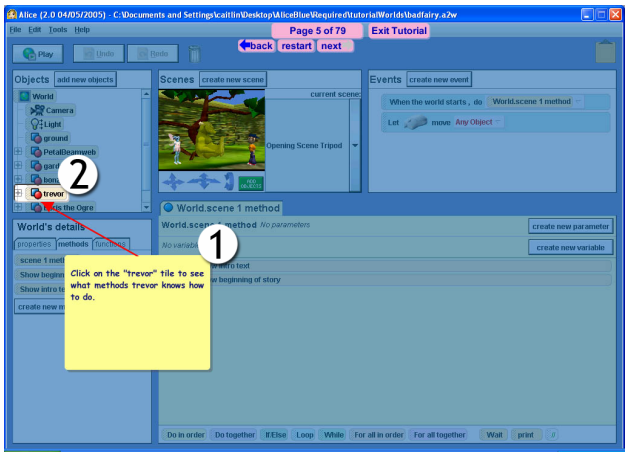


**Figure 3: The Alice interface with a Stencils-based tutorial: 1) a sticky note that provides instructions for the current step and 2) a hole in the Stencil that allows the user to interact with the interface component beneath it.**

## METHOD

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

### System Descriptions

Storytelling Alice and Generic Alice differ in three ways.

Available animations: Storytelling Alice provides high-level animations inspired by girls' storytelling goals. Generic Alice provides animations inspired by common 3D graphics transformations (see Figure 1).

Tutorial: In Storytelling Alice, users' programs animate simple stories. In Generic Alice, users' programs cause 3D objects to move, turn, and resize. The same programming constructs and explanations of the constructs are provided in both systems. Both systems present tutorials using Stencils [22].

Gallery of 3D Objects: The Storytelling Alice gallery includes characters with custom animations designed to require explanation within the story. 3D objects in the Generic Alice gallery do not include custom animations.

### 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 using Generic Alice and 43 were assigned to the experimental group using Storytelling Alice (see Figure 4). 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 all except four participants (two in each condition) were in grades 5-9. 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 donated $10 to the Girl Scout troop for each girl who participated.

### Workshop Details

Participants had two hours and fifteen minutes to complete the tutorial and create a program 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.

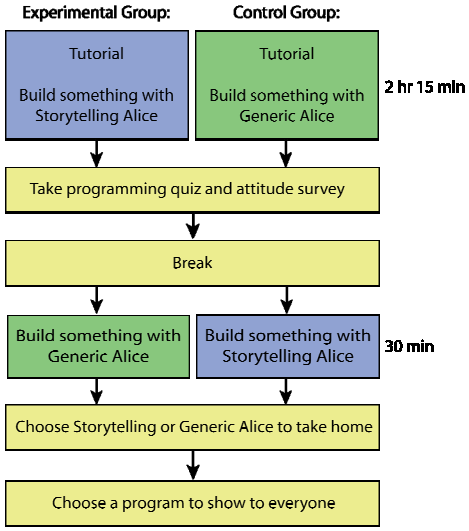


**Figure 4: Procedure for experimental and control groups.**

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 (Generic Alice) or the experimental group (Storytelling Alice). To avoid biasing participants based on their names, we referred to Storytelling Alice as Alice Blue and Generic Alice as Alice Green.

### Data

We collected program archives, a programming quiz, an attitude survey, and observations of participants' behavior.

#### *Alice Programs*

We collected participants' programs to get a qualitative picture of the kinds of animations (e.g. stories, artistic animations, 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 asked participants to select the correct description of the program's behavior from four choices. Questions covered 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. See Table 1 for Example questions and the Cronbach's alpha for each scale.

#### *Behavioral Data*

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.

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

## 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, we examine three different kinds of data: 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 data).

### What Girls Created

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

#### *Generic Alice Programs*

We observed that Generic Alice encouraged users to initially adopt an exploratory style of programming. 38% of Generic Alice users produced programs which show evi-

dence of intentional animation. 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 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.


**Figure 5: Examples of programs participants created in Generic Alice: 1) an arbitrary motion program; 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.**

Character Motions (16%):

7 of the 45 users created programs which contained one or two simple character motions (such as a cat swishing its tail or a penguin opening its beak) but were otherwise largely arbitrary motion. These worlds seem to the result of users transitioning from experimenting with the Generic Alice animations to combining animations to animate their 3D characters. See Figure 5-2.

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. See Figure 5-3.

Story-Like Sequences (16%):

7 of the 45 users created short story-like sequences. 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. See Figure 5-4.

*Storytelling Alice Programs*

We observed that Storytelling Alice encouraged users to identify a story goal quickly and begin working towards that goal. 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. Based on their stories, we believe that some participants used Storytelling Alice 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. Other participants' stories addressed topics like the difficulty of being unpopular and how to handle a crush on a boy. See Figures 6-1 and 6-2.

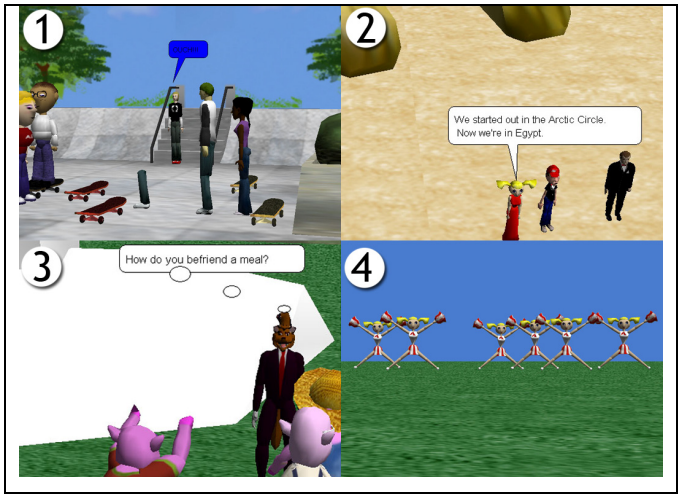


**Figure 6: Examples of programs users created in Storytelling Alice: 1) a romantic relationship story about a boy who is involved with three girls and gets caught, 2) a familial relationship story about a father taking his children on vacation and getting lost, 3) a good vs. evil story about the big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

Good vs. Evil Stories (21%):

9 of the 43 users created stories depicting conflicts between good and evil. In one story, a big bad wolf tries to befriend the three little pigs so that he can eat them later. See Figure 6-2.

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 dance routines created by Generic Alice users.

*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 (scene layout), 2) creating animations that control the motions of 3D objects (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).

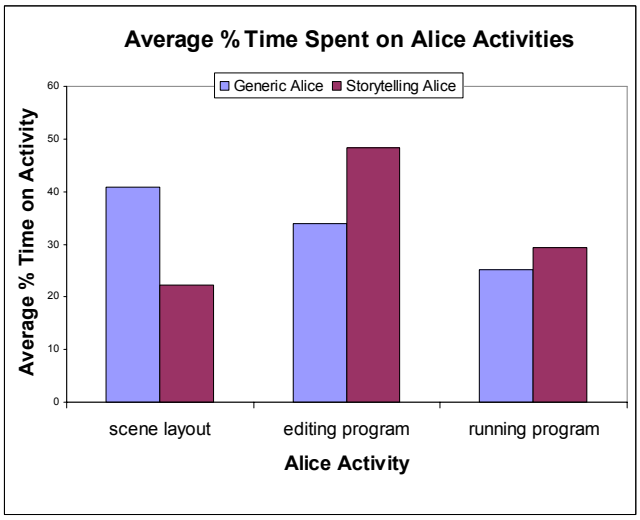


**Figure 7: Average percentage of time users of Generic Alice and Storytelling Alice spent on scene layout, program editing, and running their programs.**

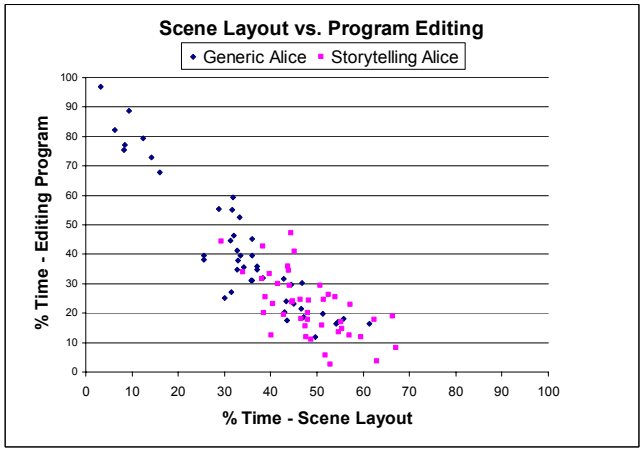


**Figure 8: Percentage of time spent on program editing vs. scene layout for users of Generic Alice and Storytelling Alice.**

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.

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.

| **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.**

**What Participants Say**

We used multivariate analysis to examine the impact of the version of Alice, 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) the 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 programming.

*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 we 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. Thus, 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, $p < .001$). One Storytelling Alice user (2%) shared 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, 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 enjoyed selecting and laying out 3D objects in the virtual world. 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 in both conditions only spent a few hours creating programs. 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

As we continue to develop Storytelling Alice, we would like to focus on three goals: 1) finding techniques that help to keep users engaged with programming in Storytelling Alice over a longer period of time; 2) finding ways to encourage users to explore and master a wider range of programming concepts; and 3) evaluating the impact of Storytelling Alice on girls' pathways into 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 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 an NSF grant. The authors wish to thank Dennis Cosgrove, Jessica Hodgins, and our anonymous reviewers for their helpful comments.

## REFERENCES

1. AAUW, *Shortchanging Girls, Shortchanging America* American Association of University Women, Washington DC, 1990.
2. AAUW, *Tech-Savvy: Educating Girls in the New Computer Age*. American Association of University Women Educational Foundation, Washington, DC, USA, 2000.
3. Alice 2.0. http://www.alice.org
4. Begel, A. LogoBlocks: A Graphical Programming Language for Interacting with the World. EECS, MIT Boston, MA, 1996.

5. Bruckman, A. MOOSE Crossing: Construction, Community, and Learning in a Networked Virtual World for Kids. MIT Media Lab. Boston, MA., 1997.
6. Bruckman, A., Jenson, C., and DeBonte, A. Gender and Programming Achievement in a CSCL Environment. *In Proc. CSCL 2002.* (2002), 119-227.
7. Dossey, J., Mullis, I., et al. *The mathematics report card: Are we measuring up?* Educational Testing Service, Princeton, NJ, 1988.
8. Duplantis, W., MacGregor, E., Klawe, M., and Ng, M. Virtual Family: an approach to introducing Java programming. In *Proc. SIGCSE 2002.* ACM Press (2002), 40-43.
9. Fennema, E. and Sherman, J. "Sex Related Differences in Math Achievement, Spatial Visualization and Affective Factors." American Educational Research Journal 14 (1977), 51-71.
10. Fenton, J. and K. Beck. Playground: an object-oriented simulation system with agent rules for children of all ages. In *Proc. OOPSLA.* ACM Press (1989).
11. Fernaeus, Y., Kindborg, M., and Scholz, R. Rethinking Children's Programming with Contextual Signs. In *Proc. IDC 2006.* ACM Press (2006), 121-128.
12. 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.
13. Furger, R. *Does Jane Compute?: Preserving Our Daughter's Place in the Cyber Revolution.* Warner Books, Inc. New York, NY, 1998.
14. Gill, J. Shedding some new light on old truths: student attitudes to school in terms of year level and gender. In *Proc. of the American Educational Research Association.* (1994)
15. Harel, I. *Children Designers*. Ablex Publishing Norwood, N.J., 1991.
16. Hoyles, C., Noss, R., and Adamson, R. Rethinking the Microworld Idea. *Journal of Educational Computing Research. 27, 1-2* (2002), 29-53.
17. Kafai, Y. *Minds in Play: Computer Game Design as a Context for Children's Learning.* Lawrence Erlbaum Associates, Hillsdale, N.J., 1995.
18. Kahn, K. Drawings on napkins, video-game animation, and other ways to program computers. *Communications of the ACM 43,3* (1996), 104-106.
19. Kay, A. Etoys and Simstories in Squeak. http://www.squeakland.org/author/etoys.html
20. 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), 165-172.
21. 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.
22. Kelleher, C. and Pausch, R. Stencils-based tutorials: design and evaluation. In *Proc CHI 2005.* ACM Press (2005), 541-550.
23. Linn, M., Fostering Equitable Consequences from Computer Learning Environments. *Sex Roles 13, 3/4* (1985). 229-240.
24. Lionet, F. and Lamoureux, Y. Klik and Play, Maxis, 1994.
25. Logo Computer Systems, Inc., My Make Believe Castle, 1995.
26. Logotron, Magic Forest, 2002.
27. 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.
28. Moskal, B., D. Lurie, et al. *Evaluating the Effectiveness of a New Instructional Approach.* In *Proc SIGCSE 2004.* ACM Press (2004), 75-79.
29. Nelson, M. Robocode. IBM Advanced Technologies, 2001.
30. 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.
31. Papert, S. *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books New York, NY, 1980.
32. Resnick, M. StarLogo: an environment for decentralized modeling and decentralized thinking. In *Ext. Abstracts CHI 1996.* ACM Press (1996), 11-12.
33. Smith, D., Cypher, A., and Tesler, L. Programming by example: novice programming comes of age. *Communications of the ACM 43, 3* (2000), 75-81.
34. StarLogo TNG. http://education.mit.edu/starlogo-tng/index.htm
35. Tanimoto, S. and Runyan, M. Play: An Iconic Programming System for Children. Visual Languages. In S. K. Chang, T. Ichikawa and P. A. Ligomenides, Plenum Publishing Corporation (1986). 191-205.
36. Vegso, J. Drop in CS Bachelor's Degree Production. *Computing Research News 18, 2* (2006).
37. Zimmer, L. and Bennett, S. Gender Differences on the California Statewide Assessment of Attitudes and Achievement in Science. *Proceedings of the Annual Meeting of the American Educational Research Association.* (1987).