Computers in the Schools

Interdisciplinary Journal of Practice, Theory, and Applied Research

ISSN: 0738-0569 (Print) 1528-7033 (Online) Journal homepage: http://www.tandfonline.com/loi/wcis20

Programming with the KIBO Robotics Kit in Preschool Classrooms

Mollie Elkin, Amanda Sullivan & Marina Umaschi Bers

To cite this article: Mollie Elkin, Amanda Sullivan & Marina Umaschi Bers (2016) Programming with the KIBO Robotics Kit in Preschool Classrooms, Computers in the Schools, 33:3, 169-186, DOI: 10.1080/07380569.2016.1216251

To link to this article: http://dx.doi.org/10.1080/07380569.2016.1216251

Published online: 15 Sep 2016.

COMPUTERS IN THE SCHOOLS
2016, VOL. 33, NO. 3, 169–186
http://dx.doi.org/10.1080/07380569.2016.1216251

Programming with the KIBO Robotics Kit in Preschool Classrooms

Mollie Elkin, Amanda Sullivan, and Marina Umaschi Bers

Tufts University, Medford, Massachusetts, USA

ABSTRACT

KIBO is a developmentally appropriate robotics kit for young children that is programmed using interlocking wooden blocks; no screens or keyboards are required. This study describes a pilot KIBO robotics curriculum at an urban public preschool in Rhode Island and presents data collected on children's knowledge of foundational programming concepts after completing the curriculum. The curriculum was designed to integrate music, literacy, and design with engineering and robotics. Children ($N = 64$) from seven preschool classrooms, ranging in age from 3 to 5, participated in the study. Findings indicated that children as young as age 3 could create syntactically correct programs for the KIBO robot, although older preschoolers (closer to age 5) performed better than younger preschoolers on a standardized programming task. Additionally, all students generally performed better on the programming tasks that required them to manipulate less programming instructions. Implications for designing developmentally appropriate curriculum and scaffolding for young children are addressed.

KEYWORDS
Early childhood education; preschool; programming; robotics

New technologies are increasingly influencing the ways young children are growing, learning, and playing. Digital activities such as playing video games and using an iPad are growing in prevalence among young children under the age of 8. For example, a recent study by Common Sense Media (2013) found that two thirds of children ages 0 to 8 have access to a console video game player at home, and 35% have access to a handheld game player such as a Game Boy, PSP, or Nintendo DS. Additionally, there has been a fivefold increase in ownership of tablet devices such as iPads from 8% of all families in 2011 to 40% in 2013.

As technology has grown increasingly common in young children's home environments in recent years, educational technology in schools has also expanded. This has occurred in part due to federal education programs and private initiatives making computer science and technological literacy a priority for young children (Office of Educational Technology, 2010). Robotics and computer programming initiatives

CONTACT Mollie Elkin elkin.mollie@gmail.com DevTech Research Group, Tufts University, 105 College Avenue, Medford, MA 02155.

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/wcis.

for young children have grown in popularity over the past 5 years as new products for young learners have emerged on the commercial market (i.e., KIBO, Bee-bot, Dot, and Dash).

Prior research has shown that children as young as age 4 can successfully build and program a simple robot (Bers, Ponte, Juelich, Viera, & Schenker, 2002; Cejka, Rogers, & Portsmore., 2006; Perlman, 1976; Sullivan & Bers, 2015; Sullivan, Kazakoff, & Bers, 2013; Wyeth, 2008), but there is still very limited research on what children under age 4 can learn with robotics. Sullivan and Bers (2015) found that preschool students were able to successfully complete basic programming tasks upon completion of a Kids Invent With Imagination (KIWI) robotics curriculum. Similarly, Sullivan et al. (2013) found that with scaffolding, 5-year-old preschool children were able to design, build, and program a LEGO® WeDo robot. Both of these studies emphasized the importance of scaffolding and moving through the curriculum at a slower pace than when working with children in kindergarten through second grade. The present study builds on prior research with preschoolers by looking at what children as young as age 3 can learn about foundational programming concepts and skills when completing a developmentally appropriate curriculum that includes built-in scaffolding and review time. This article presents results from a pilot experience in a preschool robotics program at a public school in Rhode Island.

Literature review

Robotics in early education

From tablet devices to newly designed robotics kits, young children are exploring different types of technology while at school. While access to new technologies is growing, children's understanding of *how* and *why* these tools work the way they do is growing as a new area of research. Robotics and computer programming offer a way to playfully engage students with the process of *how* motors, sensors, and electronics work (much the way they work in automated doors, sinks, and digital toys) through hands-on building projects (Bers, 2008). Young children are naturally inquisitive about how things work and are willing to take risks to uncover solutions (Peel & Prinsloo, 2001). Robotics offers an environment for children to test their hypotheses, engage in problem solving, and make personally meaningful discoveries.

In recent years, there has been a range of new robotic kits on the market for young children. For example, Bee-Bot can be used to teach sequencing, estimation, and problem solving. Children program it by pushing on the directional buttons located on the robot's body (www.be-bot.us). More recently, the makers of Bee-Bot have created Blue-Bot, which functions similarly to the Bee-Bot but can also be programed from a tablet or computer; the program is sent to the robot through a Bluetooth connection (www.terrapinlogo.com/robots.html). Another example is the Dash and Dot robots, created by Wonder Workshop. Children program these robots through iPad and Android applications to navigate a route, as well as use lights and sensors.

Some of the applications target children of all ages, whereas others target children older than age 8 (www.makewonder.com). For the research presented in this study, we have chosen to use the KIBO robotics kit described in the next section.

Research with robotics in early childhood settings has shown that beginning in preschool, children can learn fundamental programming concepts of sequencing, logical ordering, cause-and-effect relationships, and engineering design skills (Bers, 2008; Fessakis, Gouli, & Mavroudi, 2013; Kazakoff & Bers, 2011; Kazakoff, Sullivan, & Bers, 2013). When children create programs for their robots, they are sequencing commands for their robot to act out. The act of sequencing is foundational for early math, literacy, and planning (Zelazo, Carter, Reznick, & Frye, 1997). Additionally, educational robotics programs, when based in research, child development theory, and developmentally appropriate practices (National Association for the Education of Young Children [NAEYC] & Fred Rogers Center, 2012), can foster student learning of engineering such as design skills and methods (Druin & Hendler, 2000) while engaging in collaboration and other social skills necessary for school success (Clements, 1999; Lee, Sullivan, & Bers, 2013; Svensson, 2000).

The KIBO robotics kit

Although there are now many commercially available robotic kits that teach about programming, the majority described in the previous section are "pre-built" in the sense that children are not involved in any of the construction or design aspects of building a robot. For example, the Bee-Bot robot is designed to resemble a bright and colorful bumblebee, with all motors and design features ready to use, much like any children's toy. In contrast, this study utilized the KIBO robotics kit, which engages young children in both building and programming. This kit was developed by the DevTech Research Group at Tufts University and commercialized by KinderLab Robotics. KIBO is designed for young children ages 4 to 7 to learn foundational engineering and programming content; however, this study examined the hypothesis that it may be developmentally appropriate to use with children as young as 3 years old. KIBO was chosen for this study because of the large and easy-to-manipulate parts, open-ended building and programming possibilities, and the kit's tangible programming language (Sullivan, Elkin, & Bers, 2015). Because KIBO is programmed by putting together wooden blocks, without a computer, tablet, or other form of "screen-time," curricula utilizing the KIBO kit is aligned with the American Academy of Pediatrics' (2003) recommendation that young children have a limited amount of screen time per day.

The KIBO kit contains easy-to-connect robotics materials including wheels, motors, light output, and a variety of sensors (see Figure 1). In addition to these electronic components, the KIBO kit also contains art platforms that can be used for children to decorate and personalize (Sullivan et al., 2015).

KIBO is programmed by using interlocking wooden programming blocks. These wooden blocks contain no embedded electronics or digital components, but each one has a unique barcode. A scanner embedded in the front of the KIBO robot allows users to scan the barcodes on the programming blocks and send a program to

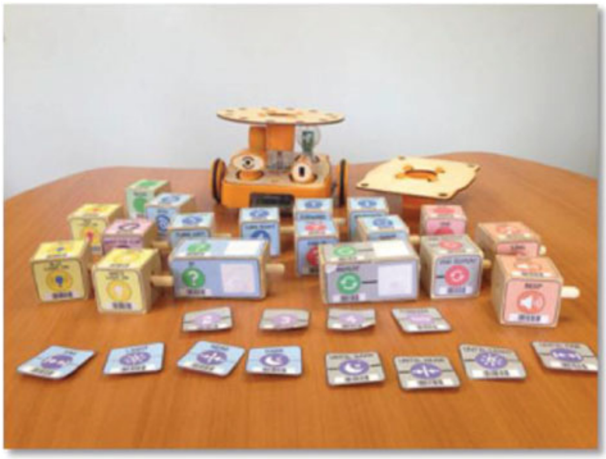

Figure 1. The KIBO robot and the blocks.

their robot instantaneously (Sullivan et al., 2015). Similar to other programming languages, KIBO has specific syntax rules to follow (see Figures 2 and 3). For example, every program must start with a Begin block and finish with an End block. Additionally, in order to create a functional repeat loop, one must use the Repeat block, a parameter (either a number or sensor), and the End Repeat block.

KIBO's developmental considerations

Young children's working memory changes drastically between the ages of 3 and 5 (Shonkoff, Duncan, Fisher, Magnuson, & Raver, 2011), enabling them to effectively learn new content. When children are entering preschool around age 3, most of them can organize themselves to complete tasks that involve following two steps, such as throwing away a napkin and putting away their lunchbox after snack time (Rhode Island Department of Education [RIDE], 2013; Shonkoff et al., 2011). By the time

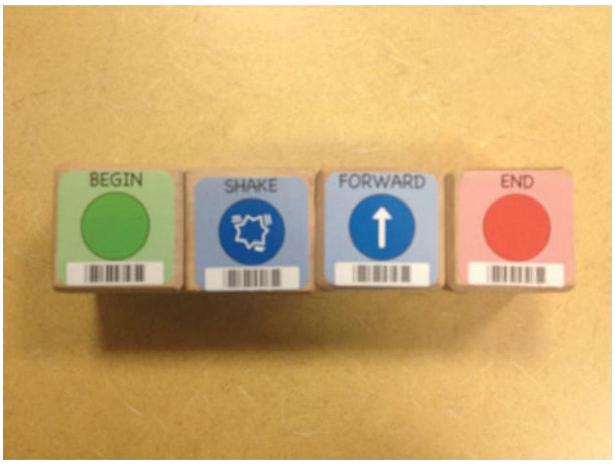

Figure 2. Basic program starting with Begin block and finishing with End block.

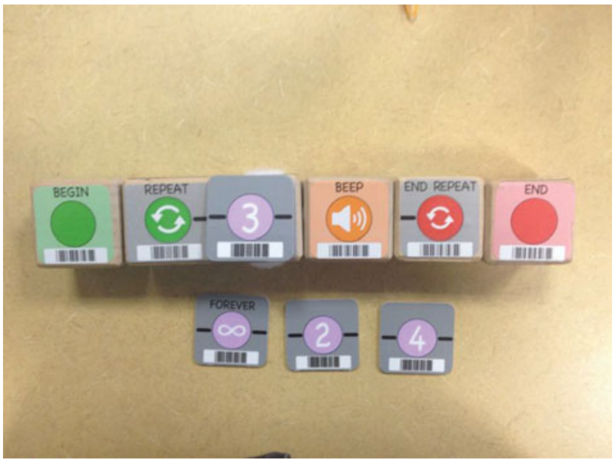

Figure 3. Repeat loops program. *Note*. The program above shows a syntactically correct repeat loops program using the number 3 parameter. This program will tell the robot to beep three times. Pictured below the program are choices of other number parameters that can be used for this type of program.

children are leaving preschool and entering kindergarten around age 5, children can follow multi-step instructions and retell familiar stories in the correct sequence (RIDE, 2013). Using the KIBO robot, children can strengthen their working memory skills by learning to sequence increasingly complex programs and to master all of KIBO's syntax rules.

By building with the robotics manipulatives such as KIBO's motors, sensors, outputs, and wooden programming blocks, children are able to develop fine motor skills and hand-eye coordination. Play that involves the manipulation of physical objects with symbolic meaning (i.e., KIBO's programming blocks that symbolize robotic actions) lets children begin to explore more complex symbolic thinking (Bers, 2008; Piaget, 1952). In addition to these technical manipulatives, children also exercise their fine motor skills through the addition of arts, crafts, and recyclable materials. Specifically, the two art platforms provide a space for exploring the engineering design process to build sturdy creations that are personally meaningful (Sullivan et al., 2015; see Figures 4 and 5). The following section describes the present research evaluating the use of KIBO robotics in a preschool context.

Robotics also engages young children in collaboration and teamwork (Lee et al., 2013). Preschool children are in the developmental process of learning social skills such as how to work with others; the design features of certain types of technology can promote social and prosocial development (Bers, 2012). Unlike many applications and educational software designed for one child working independently, robotics activities lend themselves to more collaborative moments. For example, the KIBO robotics kit used in this study is designed so that small groups of children can work on one robot with each one taking on a very specific role: the programmer, the artist, or the engineer.

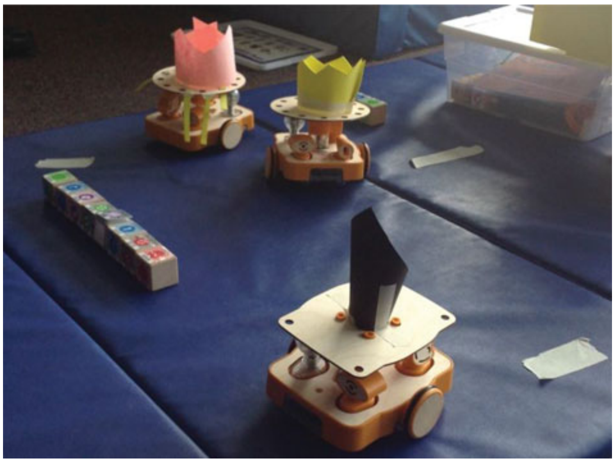

Figure 4. KIBO's static art platform for personalizing projects.

Method

Research overview

This study explores the following research questions:

1. What can young preschool children, ages 3 to 5, learn about foundational programming and robotics content through a short-term educational intervention?
2. What types of errors do young preschoolers make when programming with KIBO?
3. What kinds of programming concepts are easiest for young children to master? Which are more challenging?

To answer these questions, preschool students participated in a nine-hour introductory robotics and programming curriculum. Upon completion of the curriculum, students completed a KIBO programming task (called "Solve-It") to assess their programming knowledge.

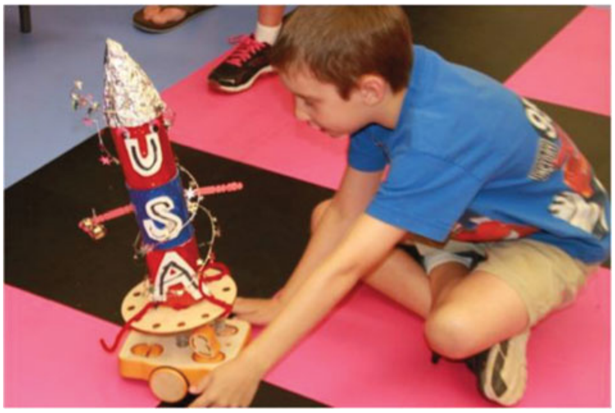

Figure 5. The motorized turntable for personalizing projects.

Participants

Participants in this study were 64 predominantly low-income, Hispanic children drawn from seven classrooms in an urban public preschool in Rhode Island. They ranged in age from 3 to 5 years (mean age = 4.83 at the time of assessment). While students in all of the classes participated in the curriculum, they were not required to partake in the assessment portion of the study; all children were invited to complete the assessment, but if they did not want to, they could choose to do something else. Students who attend this school are 81% Hispanic, 8% Caucasian, 6% African American, 5% mixed race, and 1% Native American. Additionally, 91% of the students were eligible for subsidized lunch.

Procedure

Over the course of 3 months, seven preschool classrooms completed an introductory robotics and programming curriculum taught by students from Tufts University. These undergraduate and graduate students came from a variety of educational backgrounds ranging from the liberal arts to engineering. While some students had no previous experience teaching robotics, all students had experience working with children. Student volunteers were required to attend two trainings: one 8-hour training before the start of the intervention, and one 4-hour training midway through the intervention in order to practice the curriculum and prepare for administering the assessments. Each volunteer practiced administering the assessment on other volunteers before administering the assessment on the children. The children's regular preschool teachers were in the classroom at all times to facilitate behavioral management and assist with small group work. A larger goal for the robotics works at this site was to have classroom teachers learn by observing trained students so they would be able to implement their own robotics curriculum in the future without outside help.

Curriculum overview

The introductory robotics curriculum involved approximately nine hours of work over the course of 6 days. Each day's lesson was divided into two parts: 45 minutes was spent doing an activity with the KIBO robotics kit, and the other 45 minutes was spent doing robotics and engineering-related activities that did not require the use of the KIBO robotics kit. Half of the classes completed the robotics portion first while the other completed the non-robotics portion, and then they swapped (since kits were being shared between classrooms).

During non-robotics time, children spent a portion of the class participating in a full group activity, and then spent the remainder of time participating in an activity of their choice. According to the school's principal, Rhode Island requires preschool children to spend at least half of their school day engaged in self-directed activities. The KIBO curriculum was therefore designed to include multiple activity choices

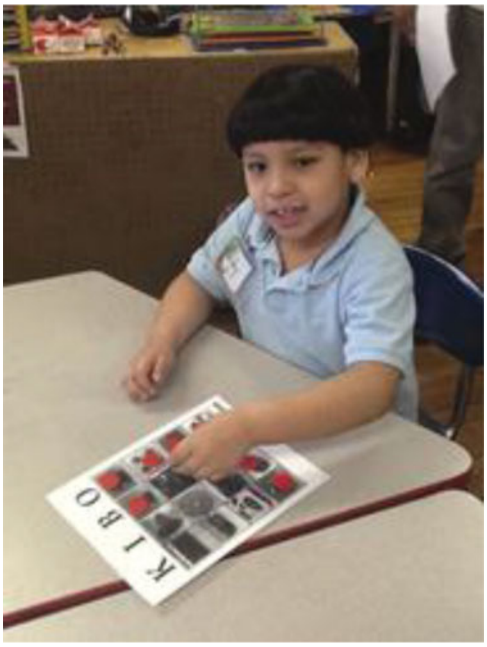

Figure 6. KIBO BINGO.

related to engineering and programming. During the group activity time, children learned songs (such as one that teaches about the different parts of the KIBO robot) and listened to picture books being read aloud (reinforcing fundamental engineering concepts like the engineering design process). During free-choice time, children could choose an activity related to KIBO. For example, one activity choice was KIBO BINGO (see Figure 6), which is played like the traditional BINGO game. The teacher showed a part of the robot (such as a wheel or a body), and students covered up that picture on their game board. The goal of this game was to teach children the different parts of the robot. Another activity choice was KIBO Says (see Figure 7), an

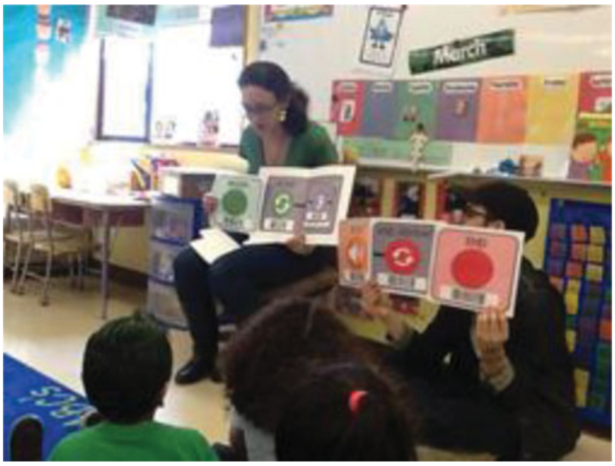

Figure 7. Simon Says with programming commands.

Figure 8. Small group work with robot.

adaptation of the Simon Says game, where children follow the directions on a large print-out version of the KIBO blocks if KIBO (the teacher) told them to do so. In addition to games, children could choose to create decorations for their robots or draw in their engineering design journals.

During robotics time, children were given a task to complete involving their robot. Each classroom had one robot for approximately three children; the student volunteers, as well as the classroom teachers, assisted the different groups. For example, during Session 2, children were asked to work in small groups to program their robots to dance the Hokey Pokey (see Figure 8). The Hokey Pokey involves children sequencing seven programming blocks in order to make the well-known song. This activity focuses on strengthening young children's working memory through trial and error and iterative programming (Shonkoff et al., 2011). It also works on their ability to understand sequence and order, which is a foundational early math and literacy component (Kazakoff & Bers, 2011). During Session 5, children worked in small groups and programmed their robots to travel along differently shaped paths using the Repeat and End Repeat blocks. The repeat loop required children to practice sequencing, order, counting, and estimation to select the correct number parameter that would make their robot travel the correct distance. See Table 1 for a breakdown of the types of activities completed each day.

On the final day of the curriculum, each class was given a KIBO robot kit to build and program together. Prior to this session, children had learned about different dances from around the world, and as a class, they selected one dance that they wanted their robot to perform. During robotics time, each class created a dance program for their robot. For example, one class programmed their robot to dance the Hula, which resulted in a program with the robot repeating the motions of moving left and right. Another class wanted their robot to move like a Salsa dancer, so they included many Spin blocks in their program. During non-robotics time, students created decorations for the robot itself as well as a "stage" for the robot to dance on. At the end of the curriculum, students presented their dancing robots to special

Table 1. Overview of the curriculum.

Session	Focus	Robotics activity	Non-Robotics activity
Session 1	Introduction to engineering and robotics	Discussion about what is a robot, play a game to learn the difference between a robot, learn the "KIBO Robot Parts" song	Discussion about what is an engineer, learn the "Engineering Design Process" song, complete sturdy building activity with non-robotic materials
Session 2	Introduction to what is a program	Introduction to KIBO's programming blocks, program the robot to dance the Hokey Pokey	Play Simon Says with KIBO commands, review the engineering design process
Session 3	Introduction to sensing and sensors	Review parts of KIBO robot with the "KIBO Robot Parts" song, review the different blocks with KIBO Bingo, free exploration with the robot	Sing "Engineering Design Process" song, read a book about the five senses, go on a sensor walk
Session 4	Sensing and introduction to repeats	Review what is a sensor, program a robot to dance to "If You are Happy and You Know it"	Talk about the meaning of the word "repeat," review KIBO's commands with KIBO Bingo
Session 5	Repeat loops with numbers	Review sound sensor, program robot to travel along different maps using repeats	Learn about dances from around the world through watching different videos, create decorations for the robot as well as a stage area
Session 6	Final projects	Create a dance for the KIBO robot based on a dance from around the world (as a whole class)	Create sturdy decorations for the robot and a stage area, plan the robot's dance

guests such as the principal and other administrators in order to celebrate the end of the unit.

Assessment

After curriculum implementation was complete, the Solve-It assessment was administered to students to assess their programming knowledge. The assessment combines KIBO's programming language and playful stories to evaluate children's mastery of different programming concepts. Because children worked in small groups during the curricular activities, it was important to implement individual assessments to see what types of tasks children could solve on their own.

The Solve-It assessment was developed to target areas of foundational programming ability and basic sequencing skills. These tasks capture student mastery of programming concepts, from basic sequencing up through repeat loops. The assessment was verbally administered one-on-one to each student by one of the volunteers who taught the robotics curriculum. The assessment required children to listen to a series of stories being read aloud to them about a robot. Then, children attempted to create the robot's program using paper versions of the KIBO programming icons provided for them (see Figure 9 for a student example and Table 2 for the story prompts). Four Solve-It tasks were administered to address the following concepts: Easy Sequencing, Hard Sequencing, Easy "Wait for" Command, and Easy Repeat Loops with Number Parameters. Tasks were called easy or hard based on how many commands children needed to sequence (i.e., easy tasks had fewer blocks for children to sequence than hard tasks).

Table 2. Solve-It story prompts and correct answers.

Solve-It number	Story prompt	Correct answer
Solve-It 1 (Easy Sequencing)	"This story is about a robot that is a car. Have you ever heard a car honk its horn? First, I want my car robot to turn on. Next, I want the car robot to honk the horn—Beep! Beep!—to warn people that it's about to move. Then I want my car to drive straight ahead, and then stop. So in this story, my robot turns on, beeps, goes forward, and then stops. Can you make a program that matches this story?"	Begin, Beep, Forward, End
Solve-It 2 (Hard Sequencing)	"This story is about a robot that drives into a puddle. I want you to make a program that lets my robot dry itself off after it accidentally moves into a puddle. First, my robot will turn on, and then it will move straight ahead—but OOPS! My robot is in a puddle! It's going to make a noise—Beep!— as if it is saying 'Oh no!' Then, I want the robot to shake itself dry—shake!—and finally, turn off! So my robot will turn on, go straight ahead, beep, shake, and then stop. Can you make a program that matches this story?"	Begin, Forward, Beep, Shake, End
Solve-It 3 (Easy "Wait for" Command)	"This story is about a dancing robot. This robot is in a dance competition. The robot has stage fright though, so it is going to wait to start dancing until it hears a clap from the audience. Once it hears a clap, it will shake and keep shaking until the end of the song. Then it will stop. So, for this story, my robot will turn on, wait to hear a clap, then shake, and then stop. Can you make a program that matches this story?"	Begin, Wait for Clap, Shake, End
Solve-It 4 (Easy Repeat)	"In this story, my robot is going to sleep. I want my robot to say goodnight to everyone in the house. It has a brother, a sister, and a mommy, so it will say goodnight to three people. First, I want my robot to turn on. Next, I want the robot to make a noise—Beep!—where it is telling us 'Goodnight!' I want the robot to say goodnight to three people, so it has to beep three times. Then, I want the robot to stop beeping, and last, to turn off." So my robot will turn on, repeat the beep sound three times, stop beeping, and then stop. Can you make a program that matches this story?"	Begin, Repeat (3), Beep, End Repeat, End

Note. Solve-It tasks are numbered based on overall difficulty of concept. For example, both the "easy" and "hard" sequencing tasks (tasks 1 and 2) are typically easier for children than the easiest "repeats" task (task 4). Within each category, there may be easy and hard tasks (for example, an "easy sequencing" and a "hard sequencing" task). Both target the same conceptual understanding, but the more difficult task has more actions to sequence.

Each of the four Solve-It tasks described was scored on a 0–6 rubric based on how close the children's program came to being completely correct (a score of 6). The scoring rubric was developed and piloted by the DevTech Research Group (Strawhacker & Bers, 2015; Strawhacker, Sullivan, & Bers, 2013; Sullivan & Bers, 2015).

Each question received two sub-scores based on separate criteria, including placement of Begin and End blocks (worth up to 3 points) and relative order of action blocks (worth up to 3 points). The scoring rubric was developed after a pilot assessment was administered to identify incorrect answer patterns that could demonstrate developmental level rather than programming comprehension. Inter-scorer reliability tests during the development of the assessment showed precise agreement (two items; $K = 0.902$, $p < 0.001$; Strawhacker & Bers, 2015).

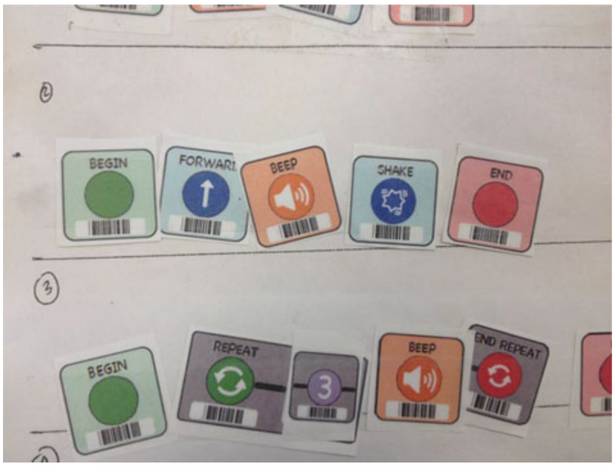

Figure 9. Sample Solve-It task.

Results

For all tasks on the Solve-It assessment, basic descriptive statistics were calculated. On average, the children in this study were highly successful at mastering basic programming concepts after completing the curriculum. After children's Solve-It data were examined for general trends and coded for the types of mistakes, the data were divided into two groups in order to compare programming performance between younger and older preschoolers. Detailed analysis is presented in the following sections.

Solve-It errors

Solve-It tasks were analyzed to look at students' knowledge of various KIBO programming concepts and the types of errors students made. By looking at the types of mistakes, we can better understand how to create developmentally appropriate curricula that provide children the opportunity to master sequencing and other programming concepts.

Sixty-four (64) students elected to participate in the Solve-It tasks (if a student was asked to participate and he or she answered no, there was no force to complete the activity), but only 61 students completed all four tasks. Each Solve-It was scored on a scale from 0–6, with 3 points awarded for the placement of the Begin and End blocks, and another 3 points awarded for the relative order of the action blocks. For this analysis, overall scores as well as sub-scores were analyzed to look at the types of mistakes that students made on each of the Solve-Its.

Children made a variety of different types of mistakes on the Solve-Its—some of which were syntactical (i.e., the program had a logical programming error and would not work on a real robot) and some of which were story related (i.e., the program made syntactical sense but did not match the sequence of the story). Aside from Solve-It 4, most students were able to create syntactically correct programs,

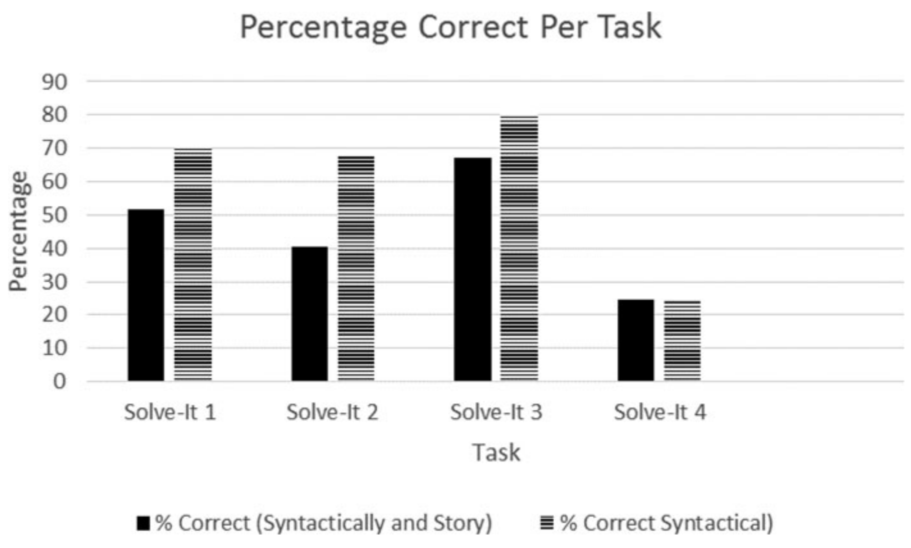

Figure 10. Percentage correct on each Solve-It task.

even if they did not match the story that they heard (See Figure 10). On three of the four Solve-Its, more than 65% of the children correctly placed Begin and End blocks.

For Solve-It 1 (Easy Sequencing), 70.4% of students made a syntactically correct program (with 51.6% of students creating a syntactically correct program that also matched the story). Interestingly, 14% of students sequenced the action instructions correctly but misplaced the Begin and/or End block. The average score, on a scale from 0–6, was 4.67 (*SD* = 1.653), with students most frequently scoring 6. The second most frequent score was 4 (26.6%), and only one student received a score of 0, which indicates that he or she not only misplaced the Begin and End blocks, but also did not order the action instructions correctly.

For Solve-It 2 (Hard Sequencing), a slightly lower percentage of students (67.8%) created a syntactically correct program, with 40.3% also making a program that matched the story. The average total score was 4.08 (*SD* = 2.043), with students most frequently scoring 6. Seven students (11.3%) received a score of 0.

Solve-It 3 (Easy "Wait for" Command) had the highest percentage (80.3%) of students who created a syntactically correct program, as well as the highest percentage (67.2%) of students whose programs also matched the story. The percentage of students who misplaced the Begin and/or End blocks was 13.1; also 13.1% of students swapped the Wait for Clap and Shake instructions. The average total score was 4.84 (*SD* = 1.827), with students most frequently scoring a perfect score of 6. Three students received a total score of 0.

The lowest scores were seen on Solve-It 4 (Easy Repeat). Only 24.6% of students created a functional program that also matched the story. The most frequent mistake (16.4%) observed was an empty repeat loop, where no action block was placed between the Repeat and End Repeat blocks. Other mistakes included swapping the End and End Repeat blocks, as well as placing the incorrect action block inside the

Table 3. Performance on Solve-It programming tasks.

Solve-It number	Concept addressed	Mean score (out of a maximum of 6)
1	Easy sequencing	4.67 (*SD* = *1.653)*
2	Hard sequencing	4.08 (*SD* = *2.043)*
3	Easy "wait-for" command	4.84 (*SD* = *1.827)*
4	Easy repeats with numbers	3.72 (*SD* = *1.845)*

repeat loop. The average total score was 3.72 ($SD = 1.845$), with 3 being the most common score. Six students received a score of 0.

Overall, the preschool students were successful in their performance on the Solve-It tasks, particularly the sequencing tasks that did not involve repeat loops. Each Solve-It was scored on a scale from 0 to 6, and the average score for Solve-Its 1–3 was above 4 (See Table 3). On all Solve-Its, students scored on average higher on their Begin/End sub-score than their Sequence/Repeat sub-score. Of the four tasks, students performed best on Solve-Its 1 and 3, which required students to sequence four instructions. They performed worse on Solve-It 2 (Hard Sequencing), which required the sequencing of five instructions, and even lower on Solve-It 4 (Easy Repeat), which required the sequencing of seven instructions.

Solve-Its by age

A one-way independent samples t test was performed to determine if there were significant differences between older and younger children's mean scores on each of the Solve-It tasks. Group placement was determined by a median split (median = 4.91). For this analysis, we had 59 ($N = 59$) students because five students did not provide their birthdays on the consent forms; 29 students were placed in the "younger" group and 30 students were placed in the "older" group. On average, the older children performed better on all Solve-It tasks. Statistically significant differences between the two groups were found on the easy ($t(57) = -2.030$, $p < .05$). Cohen's effect size value ($d = -0.54$) suggested a moderate level of practical significance. Statistically significant differences between the two groups were also found on the hard sequencing tasks ($t(55) = -2.813$, $p < .05$). Further, Cohen's effect size value ($d = -0.76$) suggested a moderate to high practical significance. There were no significant differences found between the groups on the Repeat and "Wait for" command tasks (See Table 4) indicating that both groups demonstrated the same mastery of Repeats and "Wait for" programming concepts.

Table 4. Differences between younger and older preschoolers' Solve-It scores.

Solve-It Task	Mean younger	Mean older	*t*	*df*	*p* Value
Solve-It 1 (Easy Sequencing)	4.14 (*SD* = *1.827*)	5.30 (*SD* = *1.291*)	−2.83	57	$p = .006$*
Solve-It 2 (Hard Sequencing)	3.61 (*SD* = *2.299*)	4.76 (*SD* = *1.504*)	− 2.246	55	$p = .029$*
Solve-It 3 (Easy "Wait-for" Command)	4.56 (*SD* = *2.063*)	5.14 *V(SD* = *1.432*)	− 1.941	54	$p = .058$
Solve-It 4 (Easy Repeat Loops)	3.33 (*SD* = *1.961*)	4.24 (*SD* = *1.527*)	− 1.234	54	$p = .222$

* Significant p value less than .05.

Discussion

The results from this study suggest that the KIBO robot and some aspects of the KIBO programming language are appropriate for children as young as age 3, despite the fact that it was designed for children ages 4 and older. The introductory robotics curriculum used in this study focused on rudimentary programming skills for KIBO, including sequencing and an introduction to repeat loops. Preschool children in the study, ages 3 to 5, were able to successfully master sequencing a syntactically correct program. However, the more instructions the students were asked to sequence, the more difficult it was for them to correctly create a program. This is consistent with the findings of Sullivan and Bers (2015), who found that prekindergarten students were more successful on an easy-sequencing Solve-It task (ordering four blocks) than a hard-sequencing Solve-It task (ordering five blocks). This is also consistent with the literature showing that younger children do not have enough working memory to hold five instructions simultaneously in their minds until they are several years older (Shonkoff et al., 2011).

The students' success on sequencing shorter programs may be due to their working memory and the capacity to remember all the parts of a longer story at a given time. Working memory is described as the ability to simultaneously hold and manipulate information internally over a short period of time (Shonkoff et al., 2011). During the Solve-It tasks, students had to simultaneously process the story being told, remember the programming instructions they had learned, and connect the instructions to the story. All of these elements in their mind may have been too heavy of a cognitive load for the young children in this study, even if the programming concepts were manageable.

Regardless of age, students in this study scored much lower on Solve-It 4. This suggests that programming repeat loops may be a challenging concept for very young children, either due to the number of blocks needed or their conceptual understanding of the repeat loop. The types of mistakes that students made on this Solve-It suggest that most children misunderstood the syntactical rules of creating repeat loops. Additionally, many students swapped the positions of the End and End Repeat blocks, suggesting that students had trouble distinguishing between the End block (which ends the whole program) and End Repeat block (which ends the repeat loop). Physically, both blocks include the word "end" and use the color red. However, when manipulating the blocks, the End and End Repeat blocks have different physical features (such as the absence of a peg on the End block); these features were not present when using the paper version of the blocks for the Solve-It assessment. This may have added to the difficulty for students to fully differentiate between the blocks when they needed to be used in one program. Future work may want to include a block-identification task to better understand if these young preschoolers can identify different blocks, and then using the Solve-It assessment to see if they can apply what they know about the blocks to create a syntactically correct program.

Given that less than a quarter of the children were able to create a functional program that included repeats, it may be worthwhile when introducing such young children to programming to focus on basic sequencing, or to provide more scaffolding when teaching repeat loops. Repeat loops involve more than just new blocks; they also introduce a new piece of KIBO syntax (i.e., creating a "parenthesis" with the Repeat and End Repeat block to separate a series of commands from the rest of the program). In addition to holding this new piece of syntax in their working memory, repeat loops also require children to estimate and mathematically reason with number parameters. This may be too much of a cognitive strain for children beginning to program. This connects with previous findings that preschool students spend more time than kindergarten through second-grade students on basic robotics concepts, and move through an introductory robotics curriculum at a slower pace (Sullivan & Bers, 2015; Sullivan et al. 2013).

Results also indicate that older preschoolers (around 5 years old) demonstrated a higher mean level of mastery on all programming concepts assessed than younger preschoolers (under 5 years). This may be due to a variety of factors including increased working memory, attention span, and ability to plan (Shonkoff et al., 2011). These results suggest that, even in a preschool setting, teachers should consider offering differentiated learning opportunities for students based on cognitive and social development. Additionally, when given more time, teachers may find that preschoolers, particularly the older students, may be able to master more programming concepts beyond those introduced in this study. Because KIBO programming concepts build on one another, children can easily continue to explore and master more complicated programming concepts including repeat loops with sensor parameters and conditional branching. This makes the kit ideal for preschool settings with a range of student abilities.

Limitations and recommendations for future research

The primary limitation of this study was the availability of robotic materials at the school. In order to keep the ratio of three children to one robot, robotics kits needed to be shared between classrooms. As a result, of the 9 hours in which students participated in the robotics curriculum, only half the time was spent with the robot itself. While the non-robotics time was a great opportunity for students to play games and read stories reinforcing what they learned during robotics time, students did not have a lot of time to engage in hands-on programming. Given that the assessment measured programming ability, students may have done better if they had received more practice with the curriculum.

Another limitation of this study correlates with the scheduling and logistics of working with students of all ages. Due to the schedules of both the preschoolers and the student volunteers from Tufts University, the six sessions were spread out with lengthy gaps between each one. Each lesson built off of the previous sessions, so students were asked to recall information that they had been taught a while ago. If implemented again, a longer intervention with more consistent timing may be helpful.

Furthermore, the way in which the Solve-It assessments were administered may have hindered student performance. The curriculum allowed time for students to create programs for their robot using tangible blocks, scan the program onto the robot, and then see the robot move. In this way, students were able to directly check if the program they had created made the robot move like they had intended it to. Additionally, the curriculum was designed to be open-ended, so children could choose whichever instructions they wanted for their robot to act out. Conversely, on the Solve-It assessment, students were given a specific story that they needed to recreate using paper representations of the blocks. Students did not have the opportunity to see the robot act out the program and decide whether they needed to change any instructions. Future studies may want to develop an assessment that gives students the opportunity to use the tangible blocks and check their robot's program for each story.

Conclusion

Robotics offers preschool children and teachers a playful new way to learn foundational engineering and programming concepts. This study demonstrated that it is possible to teach preschool children as young as 3 years of age fundamental programming concepts such as sequencing and repeat loops. As results from this study show, with proper scaffolding and time to explore engineering concepts through robotic and non-robotic activities, students as young as 3 can successfully build and program a KIBO robot.

References

American Academy of Pediatrics. (2003). Prevention of pediatric overweight and obesity: Policy statement. *Pediatrics*, *112*, 424–430.

Bers, M. (2008). *Blocks to robots :Learning with technology in the early childhood classroom*. New York, NY: Teachers College Press.

Bers, M. U. (2012). *Designing digital experiences for positive youth development: From playpen to playground*. Cary, NC: Oxford.

Bers, M. U., Ponte, I., Juelich, K., Viera, A., & Schenker, J. (2002). Teachers as designers: Integrating robotics into early childhood education. *Information Technology in Childhood Education*, *1*, 123–145.

Cejka, E., Rogers, C., & Portsmore, M. (2006). Kindergarten robotics: Using robotics to motivate math, science, and engineering literacy in elementary school. *International Journal of Engineering Education*, *22*(4), 711–722.

Clements, D. H. (1999). Young children and technology. In G. D. Nelson (Ed.), *Dialogue on early childhood science, mathematics, and technology education*. Washington, DC: American Association for the Advancement of Science.

Common Sense Media. (2013). *Zero to eight: Children's media use in America 2013*. San Francisco, CA: Author.

Druin, A., & Hendler, J. (Eds.). (2000). *Robots for kids: Exploring new technologies for learning experiences*. San Francisco, CA: Morgan Kaufman/Academic Press.

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5–6 sources old kindergarten children in a computer programming environment: A case study. *Computers & Education*, *63*, 87–97.

Kazakoff, E., Sullivan, A., & Bers, M. U. (2013). The effect of a classroom-based intensive robotics and programming workshop on sequencing ability in early childhood. *Early Childhood Education Journal*, *41*(4), 245–255. doi:10.1007/s10643-012-0554-5.

National Association for the Education of Young Children (NAEYC), & Fred Rogers Center (2012). Technology and interactive media as tools in early childhood programs serving children from birth through age 8. Retrieved from http://www.naeyc.org/files/naeyc/file/positions/PS_technology_WEB2.pdf

Kazakoff, E. R., & Bers, M. U. (2011, April). *The impact of computer programming on sequencing ability in early childhood*. Paper presented at the American Educational Research Association Conference (AERA), New Orleans, LA.

Lee, K., Sullivan, A., Bers, M. U. (2013). Collaboration by design: Using robotics to foster social interaction in kindergarten. *Computers in the Schools*, *30*(3), 271–281.

Office of Educational Technology. (2010). *Transforming American education: Learning powered by technology*. Washington, DC: U.S. Department of Education. Retrieved from http://www.ed.gov/technology/netp-2010

Peel, S., & Prinsloo, F. (2001). *Neuro- integration movements workshop: Module 1*. South Africa: Brain Gym Edu-K (Educational Kinesiology).

Perlman, R. (1976). *Using computer technology to provide a creative learning environment for preschool children* [Logo memo no. 24]. Cambridge, MA: MIT Artificial Intelligence Laboratory Publications 260.

Piaget, J. (1952). *The origins of intelligence in children*. New York, NY: International Universities Press.

Rhode Island Department of Education (RIDE). (2013). *Rhode Island early learning and development standards*. Retrieved from http://www.ride.ri.gov/InstructionAssessment/EarlyChildhoodEducation/EarlyLearningandDevelopmentStandards.aspx#1669797-literacy-l

Shonkoff, J. P., Duncan, G. J., Fisher, P. A., Magnuson, K., & Raver, C. (2011). *Building the brain's "air traffic control" system: How early experiences shape the development of executive function* (Working Paper No. 11). Retrieved from http://www.developingchild.harvard.edu

Strawhacker, A., Sullivan, A., & Bers, M.U. (2013). *TUI, GUI, HUI: Is a bimodal interface truly worth the sum of its parts?* Proceedings of the 12th International Conference on Interaction Design and Children (IDC ″13; pp. 309–312). New York, NY: Association for Computing Machinery (ACM).

Strawhacker, A. L., & Bers, M. U. (2015). "I want my robot to look for food": Comparing children's programming comprehension using tangible, graphical, and hybrid user interfaces. *International Journal of Technology and Design Education*, *25*(3), 293–319.

Sullivan, A., & Bers, M. U. (2015). Robotics in the early childhood classroom: Learning outcomes from an eight-week robotics curriculum in pre-kindergarten through second grade. *International Journal of Technology and Design Education*, *36*(1), 3–20.

Sullivan, A., Elkin, M., & Bers, M. U. (2015). *KIBO robot demo: Engaging young children in programming and engineering*. Proceedings of the 14th International Conference on Interaction Design and Children (IDC ″15; pp. 418–421). Boston, MA: Association for Computing Machinery.

Sullivan, A., Kazakoff, E. R., & Bers, M.U. (2013). The wheels on the Bot go round and round: Robotics curriculum in pre-kindergarten. *Journal of Information Technology Education: Innovations in Practice*, *12*, 203–219.

Svensson, A. K. (2000). Computers in school: Socially isolating or a tool to promote collaboration? *Journal of Educational Computing Research*, *22*(4), 437–453.

Wyeth, P. (2008). How young children learn to program with sensor, action, and logic blocks. *International Journal of the Learning Sciences*, *17*(4), 517–550.

Zelazo, P. D., Carter, A., Reznick, J. S., & Frye, D. (1997). Early development of executive function: A problem-solving framework. *Review of General Psychology*, *1*(2), 198–226.