Please use the following citation when citing this work:

Adams, K., & Cook, A. (2013). Programming and controlling robots using scanning on a speech generating communication device: A case study. *Technology and Disability 25*, 275–286.

Title: Programming and Controlling Robots Using Scanning on a Speech Generating Communication Device: A Case Study

Kim D. Adams[1,2] and Albert M. Cook[1]

[1]Faculty of Rehabilitation Medicine

University of Alberta

Edmonton, Alberta, Canada

[2]Glenrose Rehabilitation Hospital

Edmonton, Alberta, Canada

Corresponding author:

Kim Adams

Faculty of Rehabilitation Medicine

3-48 Corbett Hall

University of Alberta, Edmonton, Alberta T6G 2G4

E-mail: kdadams@ualberta.ca

Phone: 780-492-0309

Fax: 780-492-1626

Programming and Controlling Robots using Scanning from a Speech Generating Communication Device: A Case Study

Abstract

BACKGROUND: Programming Lego Mindstorms robots is used for problem-based learning in science. Children with physical disabilities and complex communication needs may be limited in their ability to participate.

OBJECTIVES: To involve a 12 year old student with cerebral palsy in programming Lego robots in the classroom by using her speech generating device (SGD). To evaluate the effectiveness and efficiency of using the two-switch scanning mode on the SGD for programming.

METHODS: The participant tested classmates' robot programs using infrared on her SGD, and she accessed the programming software via a customized mouse manipulation page. Her participation in programming activities was measured with Goal Attainment Scaling, descriptive observations, and measures of effectiveness and efficiency.

RESULTS: The participant progressed from observing classmates to independently testing robot programs in the classroom. In individualized sessions she wrote a simple program, with support. Limitations in scanning led to unwanted cursor movements and long task times.

CONCLUSIONS The participant actively participated in the robot programming activity. Actually programming was better suited to individual instruction because of her scanning inefficiency. Using the SGD for robot control affords the potential to also discuss concepts, but this novice user did not yet have the skills to utilize this aspect.

Keywords: augmentative and alternative communication (AAC), speech generating devices (SGD), mouse emulation, Lego Robots, scanning, usability

Programming and Controlling Robots using scanning from a Speech Generating Communication Device: A case study

1. Introduction

Children who have severe physical disabilities and complex communication needs (CCN) participate less than their peers in science activities [1]. One reason for this could be access to the curriculum materials and strategies. Current science pedagogy recommends the use of problem-based learning in hands-on activities while communicating about concepts [2]. However, this approach has challenges – physical and linguistic - for children with disabilities. Children with severe physical disabilities may be limited in their ability to manipulate the educational objects of science instruction. These children may also be unable to speak and therefore rely on alternative or augmentative communication (AAC) methods in order to participate in discussions regarding science topics. One example that incorporates both hands-on activities and communication is the use of robot programming in science education.

Programming was introduced by Seymor Papert as a medium where children could learn to use computers and influence the way they learn about everything [3]. As tools for learning, Papert and colleagues created the programming language Logo and a "turtle" that would draw based on commands given to it - the turtle could be on the screen or on the floor as a simple mechanical robot [4]. This early work led to the development of Lego/Logo where children could build their own robots with moveable gears and sensors and then program them [5]. Currently, Lego Mindstorms (The Lego Group, Billund, Denmark) robot kits are used widely; for example, the FIRST Lego League (www.firstlegoleague.org) is a world-wide competition for children 9-14 years old, who often participate through school teams [6] or sometimes through therapy programs for children who have disabilities [7].

The use of Lego Mindstorms robots by non-disabled elementary school children in problem-based learning activities has been studied. For example, pre-service teachers supervised students in first and second grade who used Lego robots to address a problem that they identified [8]. The head teacher assigned roles to each child based on their strengths (e.g., builder, programmer). Through observations, the authors report that the students learned about physics, technology, and communication skills. In another study, the problem solving strategies of 23 grade 6 students working with Lego robots were examined [9]. Through observations and questionnaires, the authors concluded that the robotic activities assisted students to reflect on the problem solving decisions they made and, with prompting, the students were able to relate their strategies to real-world contexts.

Children who have intellectual disabilities have also used Lego robots for problem-based learning. Karna-Lin, Benarik, Sutinen, and Virnes [10] worked with five groups of children, 8 to 18 years old, programming Lego robots, where some of the children had learning difficulties and mild cognitive delays. The goal of the four month study was to develop methods to support the children with special needs in active learning. Through observations and interviews with teachers, assistants and students, the authors report that group working skills (e.g., asking advice, sharing ideas) increased in all five groups. The children were motivated (e.g., more communicative and active), and had an opportunity to practice problem solving, logical thinking, perseverance, concentration, and tolerance of disappointment. Authors concluded that using the robots was useful in revealing skills and hidden potential of students, and conducive to tailoring to individual needs. Wainer, Ferrari, Dautenhahn and Robins [11] observed seven boys at the higher-functioning end of the Autism Spectrum Disorder (ASD) amoung a number of boys in an after-school computer club for children with ASD. The research goals of the four month study were to determine if interacting with robots over an extended period of time in groups would result in an increase in collaborative behaviours and if children would generalize their collaborative behaviours into another

domain. Through analysis of behavioral data, authors found that the number of potentially collaborative behaviours a child exhibited (group proxemics, shared gaze, robot-related speech, pointing behaviour, and shared positive affect) was related to the amount of enjoyment a child had in the session, measured by a student and parent questionnaire. The amount of time a child spent in a group also affected their collaborative behaviour. Collaborative behaviour also increased in a drawing domain, in three sessions interspersed over the length of the study.

No studies were found where children with physical disabilities programmed robots, but children with physical disabilities have used Lego robots to manipulate objects in play activities [12]. A commercial infrared (IR) remote controller for the RCX Lego robot was adapted so that children could use single switches to control a car-like robot using direct motor commands (e.g., to go forward or turn) or to run pre-stored programs. Direct motor commands were used for activities like taking a princess to a ball (by driving the robot, with a small princess doll on top of it, through a forest of toy trees to go to a castle made of blocks). Pre-stored programs were used by children who did not have the physical or cognitive ability to directly control the robot motors. The programs did things like spin the robot in a circle, with a marker attached for drawing.

In another study a student used the built-in IR output capability of her speech generating device (SGD) to control a Lego RCX robot to do hands-on educational activities [13]. She did math activities (e.g., rolling a die and moving her marker, the robot, the required number of spaces) and acted out a Greek myth by moving her characters and saying their script. Being able to control the robot through the SGD provided several benefits: 1) the student could practice using her newly acquired SGD and two-switch scanning access method; 2) she could access more robot functions using a full page of direct motor or pre-stored program commands stored in her SGD than she could using her two switches with the adapted remote controller (i.e., two commands maximum); and 3) she was able to engage in the learning

pedagogy of doing hands-on activities while communicating about concepts because she could manipulate the physical objects with the robot *and* communicate with her SGD. In science class, the student's classmates were building and programming Lego robots, but the teacher was unsure how to involve her.

The purpose of the case study reported here was to examine how assistive technology (AT) and adaptations to the activity might increase this student's level of participation in the robot programming activity. The student had skills that could potentially be utilized in the programming activity: 1) she had begun to learn to control the computer cursor for educational computer games using the mouse emulation mode on her SGD; and 2) because of her involvement in the hands-on math and social studies study mentioned above [13], the participant already knew how to make the robot run a program by sending a command from her SGD. The participant could potentially use her SGD to write and test robot programs as well as to talk about programming with her teacher and classmates.

It was expected that using the SGD in mouse emulation mode with scanning for writing programs for the robots might present some challenges for the participant. Scanning is less intuitive than direct access methods such as touch screens [14, 15]. Several cognitive factors make scanning more difficult than direct access [16]: children need to remember their target item for long periods of time thus increasing the load on short term memory; waiting puts a high demand on paying attention to the task; and scanning has additional visual-perceptual demands. Also the SGD's built-in cursor control commands are not optimally located on the mouse commands page for scanning (e.g., the most frequent cursor movements are not in the top left corner where the scan starts). Finally, the robot programming software has many small elements. Thus, the ease of using the mouse emulation mode with scanning for programming was also examined. It has been recommend to apply the International Standards Organization (ISO) definition of usability when examining the ease of use of AT [17]; the usability of a product is defined as the effectiveness, efficiency and satisfaction when using a product to achieve

specified goals [18]. Effectiveness is the ability of the user to complete goals, efficiency is the speed and ease to accomplish the goals, and satisfaction is the user's perceived acceptability of the product [17, 18]. In this study, we examined effectiveness as the ability of the user to control the cursor to perform the required programming tasks and efficiency as the amount of time it took to perform the tasks. Satisfaction was not measured because the student would not have been proficient enough at using her communication device to articulate what she liked and did not like about the process.

The research questions were:

1) To what degree can controlling a robot and computer through an SGD support participation of a student with a severe physical disability and CCN in classroom Lego robot programming activities?

2) What is the effectiveness and efficiency of using the mouse emulation mode with scanning from the SGD to perform Lego robot programming tasks?

2. Method

Since programming of robots by children with physical disabilities and CCN has not been studied before, a descriptive case study research design with quantitative and qualitative measures was used to investigate the research questions. The study began with a baseline phase to establish the level of participation that the student would naturally have in the classroom, and then guidance was given on how to participate by using her SGD. To address research question 1, the student's level of participation in programming activities was tracked with goal attainment scaling (GAS). GAS has been recommended for evaluating outcomes in activities where assistive technology is used [19, 20], and has been used in a previous robot study [21]. It is a criterion referenced, objective measure that allows the identification of multiple, individualized goals for participants [22]. Five potential outcomes for each goal are defined and assigned a value. The expected level of attainment for each goal is given a value of 0 and four other values (+1, +2, -1, -2) represent greater or lesser achievement. The individual's present level of performance is usually

set at level -2. As an initial step of the study, the authors, special education teacher and educational assistant (EA) wrote a goal, scaling the level of participation in the programming activity from observing, to testing, to writing programs, as seen in Table 1. The student agreed with the proposed goal and scaling. The goal level attained in each session was documented, and qualitative observations were also made in order to capture other nuances of participation, especially to examine whether the SGD would be used to talk about the programming activity with the teacher and classmates.

Table 1 Goes Here

It was not known prior to the study what an "ideal" SGD mouse commands page or robot programming software layout would look like, thus, research question 2 was examined while following an iterative technology design process [23]. With this method, adaptations are made to the technology to improve its ease of use during user trials. Having end users inform AT design is recommended [24], but since the user in this study was a student with CCN who would have found it difficult to articulate recommendations, the need for adaptations was identified by observing the student performing the programming tasks [25]. If the participant appeared to be having difficulty controlling the cursor or taking a long time to accomplish tasks then adaptations were made to the SGD page and programming software in an effort to improve the ease of use for the participant.

2.1. Participant

The participant in this case study was a 12 year old girl with CCN who has cerebral palsy with severe physical limitations affecting all four limbs. She used a Vanguard II SGD, with Unity 45 Full vocabulary set Version 4.06 (Prentke Romich Co., Wooster, OH, USA). She had the SGD for seven months prior to the study. She did two-switch step row-column scanning using two Jelly Bean switches (AbleNet,

Roseville, MN, USA) attached to her wheelchair headrest. She was still learning the Unity vocabulary system and typically only generated one word utterances, relying often on cueing from a communication partner to locate words. Except for a few social comments that she would use independently, she did not initiate conversation. She would engage in conversation initiated by others by using frequent non-verbal communication such as smiles, laughter, and head nods.

Because she made vocabulary selections that seemed to be errors, a brief targeting test was performed. This consisted of placing four markers on the screen of the SGD in specific cells of the display. The participant was asked to move her cursor to those markers as quickly as possible, and her accuracy was recorded. Reaching the cell with the marker was recorded as a success, selecting an adjacent cell was an error. Accuracy was determined as correct selections divided by total number of trials. Her target selection was 50% accurate for trial 1 and 100% for trial 2. Thus, it was established that her errors in vocabulary selections were probably the result of randomly selecting cells to get out of rows selected while searching for vocabulary.

The participant was in an integrated grade six classroom (with students aged 11 to 13 years), however, she was not at the same academic level as her classmates. An educational assistant (EA) provided academic and personal assistance to her and another student. The participant performed individualized reading, writing, and math activities with her EA, and group activities such as acting in drama class with her peers. Prior to receiving her SGD, she had no means for written communication, hence her skills were delayed - she wrote short sentences with assistance from her EA who provided sentence frames for copying. No formal testing was performed as part of this study, but the participant's special education teacher reported that psychological testing done prior to the study indicated that the participant had mild to moderate intellectual impairments. The teacher felt that the impairments were due to reduced opportunities for learning.

2.2. Setting

Sessions were held in a computer lab in the participant's school. The entire grade 6 class and teacher were present in the lab and the participant joined a group of three non-disabled students doing the Lego programming. After six sessions the classmates went on to another science topic, but the special education teacher preferred that the participant continue doing programming. So, for six more sessions the participant did one-on-one programming with author 1, and classmates came by occasionally to observe.

2.3. Materials

The robot used by the class was an infrared-controlled Lego RCX Mindstorms roverbot, a car-like robot (Figure 1)**.** The participant could control the robot by sending direct commands to the robot motors (e.g., forward, backward, left, and right) or by sending a command to start or stop one of five programs stored on the robot. She did this from a page created on her SGD with the robot commands and some vocabulary (e.g., "It's not working", "This is fun", and "This is boring") (Figure 2, left). A Lego remote control unit was used to put the infrared signals into the SGD, according to the SGD manufacturer instructions.

Figure 1 Goes Here

Figure 2 Goes Here

The Robolab software included with the Lego Mindstorms kit was used for programming the Lego robot (Figure 3, left). Students in the classroom accessed the software on standard personal computer stations.

For the participant, the software was installed on a Sahara Slate PC tablet computer (TabletKiosk, Torrance, CA, USA) mounted on a rolling stand. The participant's SGD was connected to the tablet computer with a USB cable, and the built-in mouse emulation mode was used with the mouse commands page for cursor control (Figure 2, right).

Figure 3 Goes Here

2.4. Procedure

The weekly integrated classroom sessions were 40 minutes long. There were three baseline sessions, and then there was a four week period when there were no programming classes since the classmates were writing departmental exams. There were three more integrated classroom sessions, and in these sessions the teacher let the classmates know that they could ask the participant to test their robot programs for them. Running a program on the Lego robot is typically done by pressing a small button on the robot itself or by sending an IR command from the Lego remote control unit corresponding to the location (1-5) of the stored program. The participant ran programs by sending the IR command from her SGD.

There were six weekly one-on-one programming sessions with author 1, 60 to 90 minutes long including breaks. The required tasks for robot programming were as follows (refer to Figure 3 for elements mentioned in this list):

- **Insert** a module icon into the workspace between the start and end icons (i.e., move cursor to the module palette, click on the module, drag it between the green and red lights, and click to release it).
- **Connect** the corners of consecutive modules to each other using the connection tool (i.e., move cursor to the corner of a module, click on it, move to the corner of the next module, click on it).

- **Download** the program to one of the robot program slots 1 through 5 (i.e., move cursor to the Download button in the program menu and click) - author 1 connected the USB cable from the computer to the robot and chose the program slot 1-5.
- **Test** the program (i.e., as described above, typically by pressing a small button on the robot, but the participant ran programs by sending the IR program 1-5 command from her SGD).
- To modify module parameters:
  - **Select a module** (i.e., move cursor to a module in the workspace and click on it)
  - **Select parameter** (i.e., click to bring up a palette of parameter options, move cursor to desired option, e.g. motor direction, and click on it).

Author 1 provided prompts to the participant for cursor control and robot programming tasks as needed. Also, author 1 assisted with cursor control in the following ways:

- clicking to bring up the palette of parameter options (since it was very small target), and then placing the cursor so that the participant only had to move the cursor slightly to her desired choice and then click (called **assisted select**).
- replacing the cursor where it was before it jumped across the screen due to a participant mis-selection on the SGD mouse commands page, but only if the participant responded positively to the question, "Do you want me to put the cursor back?" (called **rescues**).

The participant first wrote a simple program (A and C motors forward for 8 seconds) and then modified motor directions so the robot went backwards, spun in a circle one way and then the other. These were programs that she used in the math and social studies study mentioned above, but were written by author 1 in that study [13]. Table 1 shows all of the programming tasks and activities performed by the participant in the programming sessions.

Table 2 Goes Here

The iterative process of adapting the SGD mouse commands page in order to maximize efficiency and effectiveness led to the modifications circled in Figure 2, right. After programming session 1, the bottom right icon of the SGD page, a shortcut to make the cursor jump to and click on the top left menu item of whatever program is active on the computer display, was relocated from the top left to the bottom right corner because the participant frequently accidentally selected it. After session 2, the three cursor "jump to location" shortcuts along the left side of the SGD page were re-programmed to jump to three relevant locations in the programming interface (i.e., the download button, modules palette, and workspace shown in Figure 3). The participant did not use these shortcuts until pictures of the "jump to locations" were placed on the buttons in session 5 (pictures were made by taking a screen capture and cropping to include only the relevant area of the program screen). Running the program with 640x480 pixels in session 6 made the module icons larger and the distance between them shorter (Figure 3, right).

2.5. Data Collection and Analysis

The GAS level of participation score achieved after each session was determined by author 1 and then verified by members of the participant's assistive technology team. The team verified the scores by looking at artifacts from each session (e.g., observation notes and screen captures of the robot program at the end of each session). This is in accordance with recommendations in Schlosser [19] that the people who evaluate achievement in the GAS goals, be different from the people who created the goals.

Observations of the communication between the participant and her classmates were made by author 1, and included: observing the number of classmates with whom she interacted, the nature of the interaction, e.g., social or academic, the types of statements made, and to what extent the participant received prompting. The observations were summarized and shown to the AT team.

Effectiveness was measured by tracking the number of times that the participant made the cursor jump to unwanted locations and the number of times that author 1 positioned the cursor (for rescues). Efficiency was measured as time spent on each programming task (minus time for breaks, distractions and/or technical difficulties). Morae Usability Analysis software (TechSmith, Okemos, MI, USA) was used to obtain the effectiveness and efficiency data. Morae synchronizes computer screen activity, mouse clicks, a video of the participant's face, and manual coding. The cursor jumps and rescues were coded manually. The start and end of each programming task was marked manually, and time on tasks was automatically calculated. To graph the task times, similar tasks were grouped into categories. In other words, inserting a Motor, Wait or Stop module were all called "Insert"; selecting a Motor, Wait, Parameter, or Ok button were all called "Select"; and all tasks that were assisted, were called "Assisted". Connect and download tasks were not grouped. Testing the robot programs, "playing with the robot" (by controlling the robot by direct control of the motors and programs), and writing the name of the program were not shown in the graphs.

3. Results

During the three baseline sessions, the participant observed her classmates do the robot programming. She often did not pay attention to the programming; she was behind the other students and probably could not see what they were doing and they performed tasks quickly without explaining what they were doing. This level of involvement in programming, observing classmates, was a GAS score of -2. In the three sessions after the teacher let the classmates know that the participant could test their programs, her own group and two other groups asked her to do so and she did it without prompting. During these sessions the participant's level of involvement in programming was at GAS score -1, testing programs. In the six programming sessions, the participant wrote a simple program and made variations to it. This level of involvement is at a GAS score of 0. The participant required heavy prompting for robot programming

overall, but she did become independent with some tasks (e.g., she independently download programs once she was told to see if the program works).

It was observed that, through learning to program the participant better understood the difference between direct motor control commands and programs. Early on in the study, the participant discovered a technique to reduce the number of required switch presses to go long distances; she pressed and held her switch on the direct forward command and the output timing of the SGD was such that it would send repeated IR commands as long as the switch was pressed, thus, the robot would continue to move until she released the switch. She also tried this technique of pressing and holding on a program command, for example a program to go forward for 8 seconds, but this caused multiple repetitions of the program resulting in the robot moving forward for over a minute and overshooting the destination. By her last robot programming session she demonstrated an understanding of programs by selecting a program command and then watching the robot until it finished moving before selecting the program command again. She also began to use programs strategically to go long distances instead of pressing and holding on the direct motor control commands.

Observations about communication revealed that the participant did not use her SGD to say anything in the baseline sessions, but used non-verbal communication with several classmates, by smiling and nodding to respond to their comments. The comments were social rather than anything related to robot programming. In the sessions when she tested programs for her classmates she again did not say anything using her SGD, but she responded to classmates' comments by smiling and nodding. In this case the comments were related to robot programming; her classmates made comments such as: "*That's so cool*", "*Where's the infrared?*", and "*Do it again*". One student explained to her exactly how his robot program should have run. When her classmates came by to observe her doing the programming using the mouse emulation mode from her SGD they said things like, “Great job”, and the participant smiled.

The number of times that the participant caused the cursor to make unwanted jumps and the number of times that author 1 positioned the cursor for rescues is shown in Figure 4, bottom. The efficiency of doing the programming tasks is shown in Figure 4, top. Authors noted that the participant became independent in cursor control commands over the course of the study. The teacher reported that the participant's skill at using the mouse commands page in other computer activities improved as a result of the study.

Figure 4 Goes Here

---

4. Discussion

Robots have been proposed as a promising technology for children with disabilities because children can use robots to manipulate items, and also be involved in problem-based learning while programming them [26]. A previous study demonstrated the feasibility of controlling Lego robots from SGDs to manipulate items in various academic activities [13] and this study demonstrated the feasibility of being involved in the problem-based learning activity of programming the Lego robots from the SGD.

The adaptations used in this study helped to increase the participant's involvement in the programming activity. Controlling the robot from the SGD with IR commands to run and test programs was an easy adaptation to the activity and increased her involvement in robot programming in the integrated classroom. The participant went from observing classmates in the baseline (GAS level -2) to testing programs for classmates in sessions 4, 5 and 6 (GAS level -1). Being able to control the computer cursor from the SGD to do programming increased her involvement in programming even further in the one-on-one sessions; she was involved in writing a two-step program (GAS level 0). Due to the heavy prompting required by the participant to do the programming, it was appropriate that the programming

occurred in one-on-one programming sessions rather than in the integrated classroom. Though she was not independent in writing the programs, she was actively involved, and learned the function of programs and used them strategically.

Another way that students can participate in programming activities is by discussing and reflecting on the activity. One proposed benefit to controlling the robot and computer cursor through the SGD, was that the participant had the potential to use the SGD to do the hands-on activities *and* talk about the topic with classmates and teacher. Though the participant engaged in the hands-on aspect of testing robot programs in the class, she did not use her SGD to say anything about programming. She did interact socially with her classmates with non-verbal communication, which is important [27]. The computer lab was a noisy environment and she was a novice user who had not yet built the social skills needed for initiating communication with her SGD in unstructured situations. The participant even needed support to utilize her existing skills; in the baseline sessions, the participant already knew how to send robot programs from her SGD, but she did not offer to test programs for her group. The participant only did the testing after the teacher suggested that she could, and the other students asked her to do it. The SGD afforded the potential, but the participant would have needed some support from a teacher or EA to seize the opportunity for more communicative interaction with classmates.

Though not initiated by her, the participant had positive interactions with classmates when she became the center of attention, rather than being an observer, while she tested their robot programs. Classmates validated her participation with task-related supportive comments. She also received validating comments from her classmates when they came by to observe the one-on-one programming sessions. While supportive of her programming, her classmates seemed more impressed that she could control the robot remotely. Perhaps this is because programming was something they could do themselves, whereas they could not control the robot directly. These results are consistent with previous

robot studies where teachers have perceived students as more capable after seeing them working with the robots [21, 28].

Using mouse emulation mode with scanning from the SGD to perform Lego robot programming was feasible, but not highly effective or efficient in this case study. The participant caused the cursor to jump on several occasions, and although it took her a great deal of time, she usually preferred to move the cursor back to the original position herself rather than letting author 1 "rescue" her. This is shown in Figure 4, bottom, where the number of cursor jumps are generally higher than the number of rescues for each task. On the tasks where the number of rescues is higher than the number of cursor jumps, the additional rescues generally occurred when the computer did something unexpected (e.g., the cursor disappeared and author 1 moved the mouse to make it reappear). Another indication of effectiveness of using the SGD for programming was the number of times that author 1 did an assisted selection. This was influenced by the nature of the software, as it had very small targets. If independence in the task is important then very good user operational skill is necessary, or further customization to the technology could be implemented, for example, macros for program tasks or more "jump to locations" on the SGD mouse controls page.

The participant’s efficiency in accomplishing programming tasks generally did not improve over sessions as would be expected with repetition. Rather, it was variable, and ranged from one to fifteen minutes in the tasks that she did independently. In Sessions 2, 3 and 5, spikes in the amount of time correspond to spikes when the participant caused the cursor to jump to unwanted locations (indicated with circles on Figure 4, top and bottom). Though user operational skill was a factor here, there were other factors that affected time to complete tasks that were out of the participant's control. Other tasks had high time values because of the nature of the software and task, for example, in Session 3 the participant needed to connect very small targets requiring a lot of small cursor movements. Tasks in Session 6 have

shorter times, around 1 minute, because many assisted selections were necessary while the participant modified existing module parameters, requiring clicking on very small targets. This data shows that researchers and clinicians should be careful when using time to track efficiency of technology when users have severe physical disabilities. There are many components that contribute to accomplishing a task, including user skill, technology demands, and task demands, and they should all be considered in the final assessment of efficiency.

The modifications made to the SGD display (moving and creating "jump to" location shortcuts) and the program (running the program in 640x480 mode) were made in an effort to reduce the participant's mis-selections and to increase efficiency of doing tasks. Figure 4 reflects this in some cases, for example, after the shortcut to "jump to the download button" was added, it took the participant less than one minute to accomplish the download task, compared to five minutes or more previously. Also, selection times of only one minute were facilitated by using the "jump to" palette and workspace shortcuts. However, the participant continued to make several mis-selections during the sessions, likely due to her use of the strategy to randomly select cells to get out of rows selected while searching for items on her SGD display. Since most SGDs require some customization when purchased, it is expected that teachers, EAs and AT specialists would be able to make these sorts of modifications.

There were some limitations to the study. Since it was a case study, no inferences regarding causality can be made. Also, inter-rater reliability of the observations made in the classroom was not performed. Classroom observations were not used as a dependent variable, but they were used to more richly describe the concept of participation, in ways that the rigid GAS goal about level of involvement in programming tasks could not capture. The behaviors observed could be helpful in establishing GAS goals or other measures in future studies. For example, we established that interactions do occur, so in future studies the actual number of interactions could be counted. Or, since the classmates comments

indicated they were impressed by the participant's capabilities, a measure of perceived competence could be administered in future studies. Also, inter-rater reliability of the number of cursor jumps and rescues was not performed. However, the Morae software clearly shows when cursor jumps occurred, and the audio recording in the video makes it clear who was performing a specific task.

5. Conclusion

AT and adaptations to the activity increased the participation in a Lego robot programming activity of a student who had severe physical disabilities and CCN. Controlling the robot from the SGD with IR commands to run programs was an easy adaptation which enabled her to have a central role in the classroom while she tested programs for classmates. Being able to control the cursor to do programming increased her level of participation in programming even further, but due to the noisy classroom environment and heavy prompting required by the participant, this was more appropriate to implement in one-on-one sessions rather than the integrated classroom.

Though the participant engaged in the hands-on aspects of the activity, manipulating the robot by sending program commands and manipulating the computer cursor for programming from her SGD, she did not use her SGD to say anything. The novice SGD user would have needed some support to also communicate with classmates about programming. However, controlling the robot and computer cursor through the SGD gave the potential to begin to address the current science pedagogy of problem-based learning in hands-on activities while communicating about concepts.

Using mouse emulation mode with scanning from the SGD to perform Lego robot programming was feasible, but since the participant was a novice user she had operational limitations which manifested in considerable time to accomplish the tasks, and she was not able to do them without assistance. The technology itself posed some limitations, some of which were addressed with modifications to the software and SGD interface. Further customization could be done to increase independence.

Future studies will involve more students with disabilities in problem-based learning with non-disabled peers, with measures to establish if performance in and understanding of programming improves. Methods to control the Lego Mindstorms NXT robots through SGDs should be implemented, for example by custom programming in [29]. Other methods to access programming should also be investigated, for instance computer-based SGDs are now available, so more intuitive cursor control methods could be used, while still having access to language. With innovations such as these, children with disabilities will be better situated to more fully engage in hands-on robot programming in the classroom and reflect on the concepts being learned.

6. Acknowledgements

We kindly acknowledge Elaine Holtham at Aroga in Vancouver, B.C., for the loan of a Vanguard for use during the study.

7. References

[1] Eriksson L, Welander J, Granlund M. Participation in everyday school activities for children with and without disabilities. Journal of Developmental and Physical Disabilities. 2007;19:485–502.

[2] National Science Teachers Association. NSTA position statement: Elementary school science. http://wwwnstaorg/about/positions/elementaryaspx. Arlington, Virginia 2002.

[3] Papert S. Mindstorms: Children, computers, and powerful ideas. New York: Basic Books; 1980.

[4] Papert S. Teaching children thinking. Innovations in Education & Training International. 1972;9(5):245-55.

[5] Resnick M, Ocko S, Papert S. Lego, Logo, and design: Children and interactive electronic environments. Children's Environments Quarterly. 1988;5(4):14-8.

[6] Portz SM. Lego league: Bringing robotics training to your middle school. Tech Directions. 2002

May:17-9.

[7] Redepenning S, Mundl J. Robots. Closing the Gap Solutions: Assistive Technology Resources for Children and Adults with Disabilities. 2012 December 2012/January 2013:18-9.

[8] Bers MU, Ponte I, Juelich K, Viera A, Schenker J. Teachers as designers: Integrating robotics in early childhood education. Information Technology in Childhood Education. 2002(1):123-45.

[9] Castledine A-R, Chalmers C. Lego robotics: An authentic problem solving tool? Design and Technology Education. 2011;16(3):19-27.

[10] Karna-Lin E, Pihlainen-Bednarik K, Sutinen E, Virnes M, editors. Can robots teach? Preliminary results on educational robotics in special education. Proceedings of the Sixth International Conference on Advanced Learning Technologies (ICALT'06); 2006; Kerkrade, The Netherlands.

[11] Wainer J, Ferrari E, Dautenhahn K, Robins B. The effectiveness of using a robotics class to foster collaboration among groups of children with autism in an exploratory study. Personal Ubiquitous Computing. 2010;14(5):445–55.

[12] Cook A, Adams K, Volden J, Harbottle N, Harbottle C. Using Lego robots to estimate cognitive ability in children who have severe physical disabilities. Disability and Rehabilitation: Assistive Technology. 2011;6(4):338-46.

[13] Adams K, Yantha J, Cook A. Lego robot control via a speech generating communication device for play and educational activities. Proceedings of the RESNA Conference; Washington, DC 2008.

[14] Higginbotham DJ, Shane H, Russell S, Caves K. Access to AAC: Present, past and future. AAC Augmentative and Alternative Communication. 2007;23(3):243-57.

[15] Light J, Drager K. AAC technologies for young children with complex communication needs: State of the science and future research directions. AAC Augmentative and Alternative Communication. 2007;23(3):204-16.

[16] Ratcliff A. Comparison of relative demands implicated in direct selection and scanning: Considerations from normal children. Augmentative and Alternative Communication. 1994;10:67-74.

[17] Arthanat S, Bauer SM, Lenker JA, Nochajski SM, Wu YWB. Conceptualization and measurement of assistive technology usability. Disability and Rehabilitation: Assistive Technology. 2007:1-14.

[18] ISO 9241-11 ergonomics requirements for office work with visual display terminals (VDTs) - part 11: Guidance on usability. Geneva: International Standards Organization (ISO); 1998.

[19] Schlosser RW. Goal attainment scaling as a clinical measurement technique in communication disorders: A critical review. Journal of Communication Disorders. 2004;37:217–39.

[20] Ottenbacher KJ, Cusick A. Goal attainment scaling as a method of clinical service evaluation. American Journal of Occupational Therapy. 1990 Jun;44(6):519-25.

[21] Cook AM, Bentz B, Harbottle N, Lynch C, Miller B. School-based use of a robotic arm system by children with disabilities. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2005;13(4):452-60.

[22] Kiresuk TJ, Smith A, Cardillo JE. Goal attainment scaling: Applications, theory and measurement. Kiresuk TJ, Smith A, Cardillo JE, editors. Hillsdale, NJ: Erlbaum; 1994.

[23] Green WS, Klein D. User trials as a design directive strategy. Human factors in product design. London: Taylor & Francis; 1999. p. 92-102.

[24] Blackstone SW, Williams MB, Joyce M. Future AAC technology needs: Consumer perspectives. Assistive Technology. 2002;14:3-16.

[25] Hanna L, Risden K, Alexader KJ. Guidelines for usability testing with children. Interactions. 1997;September and October:9-14.

[26] Cooper M, Keating D, Harwin W, Dautenhahn K. Robots in the classroom - tools for accessible education. Proceedings of the AAATE Conference, The 5th European Conference for the Advancement

of Assistive Technology; Dusseldorf, Germany 1999.

[27] Raghavendra P, Olsson C, Sampson J, Mcinerney R, Connell T. School participation and social networks of children with complex communication needs, physical disabilities, and typically developing peers. AAC: Augmentative & Alternative Communication. 2012;28(1):33-43.

[28] Cook AM, Howery K, Gu J, Meng M. Robot enhanced interaction and learning for children with profound physical disabilities. Technology and Disability. 2000;13(1):1-8.

[29] Chung Y, Simpson R. The development of a robot controlled by speech generating devices for children. Proceedings of the RESNA Conference; Toronto, ON 2011.

Tables

Table 1: Goal attainment scale levels for participant's level of participation in robot programming

Goal Attainment Scale (GAS)	**Expected participant performance**
-2 (much below expected)	Participant observes classmates programming.
-1 (somewhat below expected)	Participant tests programs for classmates by sending the appropriate program command from her SGD.
0 (expected status)	Participant involved in writing a simple (i.e., 2 steps) program for the robot (gets it downloaded, perhaps by another student) and tests it.
+1 (somewhat above expected)	Participant involved in writing a longer program (3-4 steps) for the robot, (gets it downloaded, perhaps by another student) and tests it.
+2 (much above expected)	Participant involved in writing a more complex program (with conditional statement or sensor), (gets it downloaded, perhaps by another student) tests it, and demonstrates understanding of fixing program errors.

Table 2: Robot programming tasks performed by the participant

Session	Task Number and Programming Task
1	**Program robot go to forward for 8 seconds:** 1) Insert Motor C, 2) Insert Motor A
2	3) Reinsert Motor A, 4) Reinsert Motor C, 5) Insert Wait, 6) Select Wait 8s, 7) Bring up change palette (Assisted), 8) Insert Stop, 9) Connect Start to Motor A, 10) Connect Motor C to Wait
3	11) Select extraneous module (Assisted), 12) Delete module (Assisted), 13) Connect Motor C to Wait, 14) Connect Wait to Stop Sign, 15) Connect Stop Sign to Stop
4	16) Download, 17) Repeat Download (Assisted), 18) Test
5	19) Play with robot by sending direct commands and programs (called Play with robot) **Program robot to go in a circle:** 20) Select Motor C, 21) Select Motor C backwards (Assisted), 22) Download (but robot was off) , 23) Press OK Dialog Button, 24) Repeat download, 25) Test, 26) Write program name, 27) Test
6	28) Play with robot **Program robot to go backwards:** 29) Select Motor A, 30) Select Motor A backwards (Assisted), 31) Select Motor C, 32) Select Motor C backwards (Assisted), 33) Select Download button (Assisted), 34) Test 35) Play with robot **Program robot to go in a circle the other direction:** 36) Select Motor C, 37) Select Motor C forward (Assisted), 38) Select Motor A, 39) Select motor A backwards (Assisted), 40) Download, 41) Download again (Assisted), 42) Test

Figure Captions

Figure 1. Lego RCX Mindstorms roverbot, a car-like robot

Figure 2. (Left) SGD robot page layout showing robot directional commands, program 1-5 and some vocabulary. (Right) The SGD mouse commands page, with modifications indicated by circles. Screen captures made with PASS software (www.prentrom.com).

Figure 3: Robot programming environment (Left) at first session and (Right) after last session. The download button, module palette and workspace are indicated with arrows.

Figure 4. (Top) The amount of time that it took to do each programming task (i.e., insert module, connect module, download program, select module, assisted selection). (Bottom) The number of times that the cursor jumped and the number of times the investigator repositioned the cursor (rescues). Spikes in the time data which correspond to high number of cursor jumps are indicated by circles on each graph.

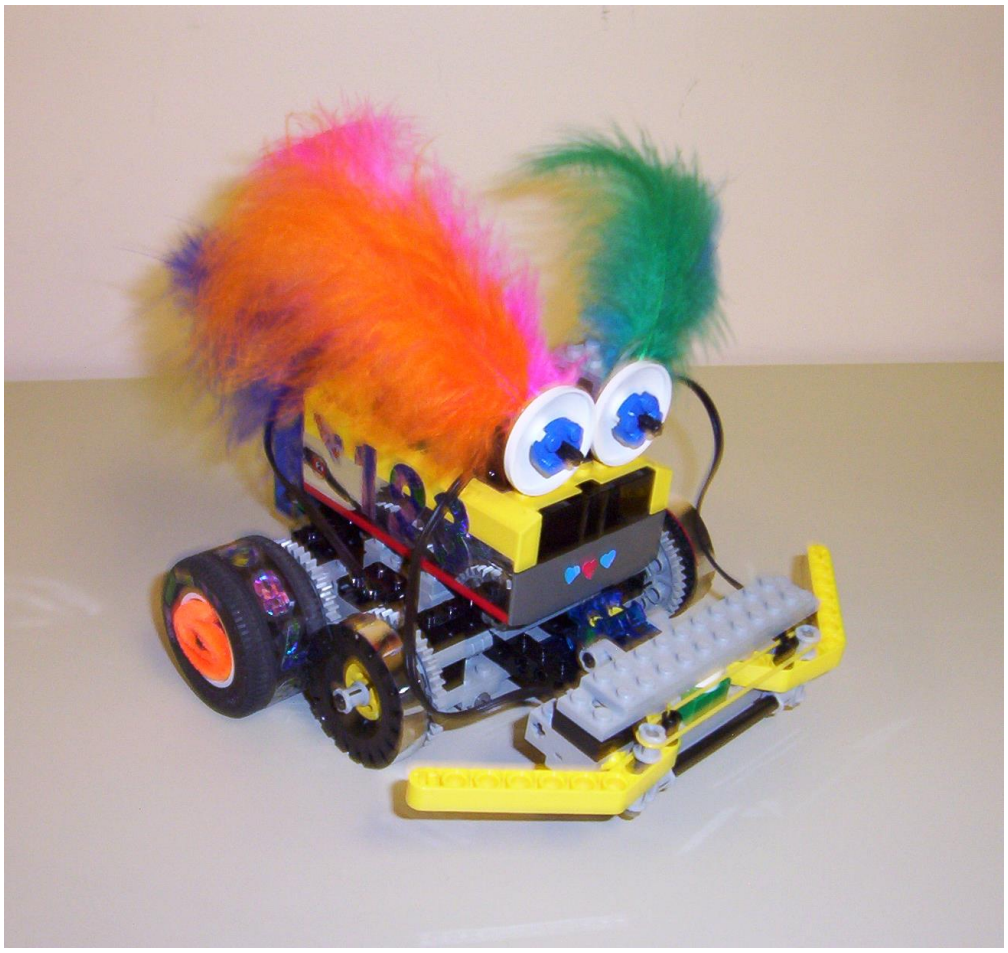

Figure 1. Lego RCX Mindstorms roverbot, a car-like robot

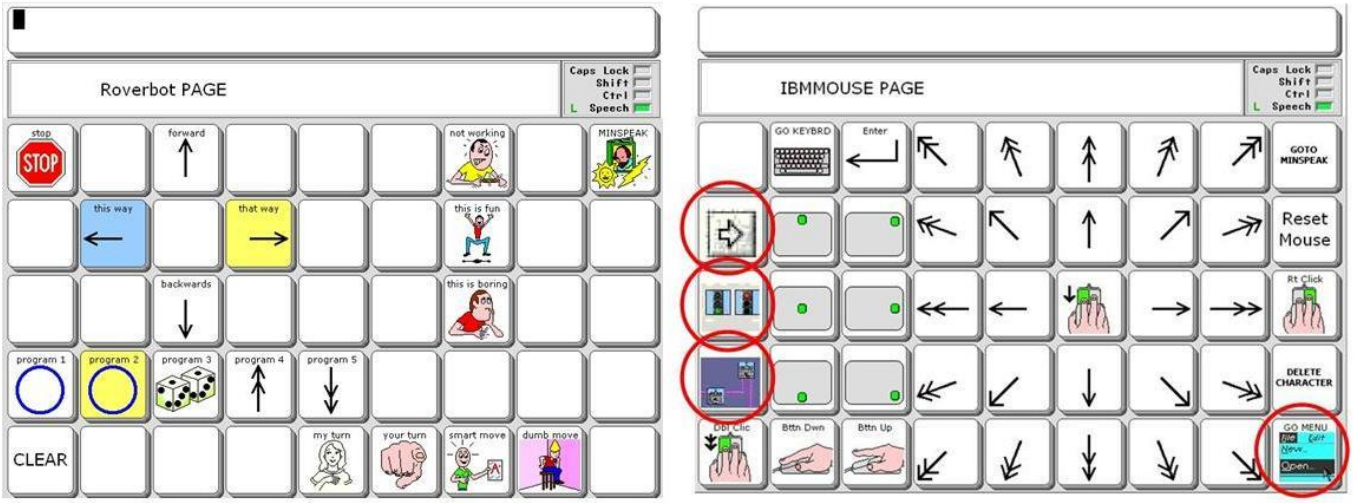

Figure 2. (Left) SGD robot page layout showing robot directional commands, program 1-5 and some vocabulary. (Right) The SGD mouse commands page, with modifications indicated by circles. Screen captures made with PASS software (www.prentrom.com).

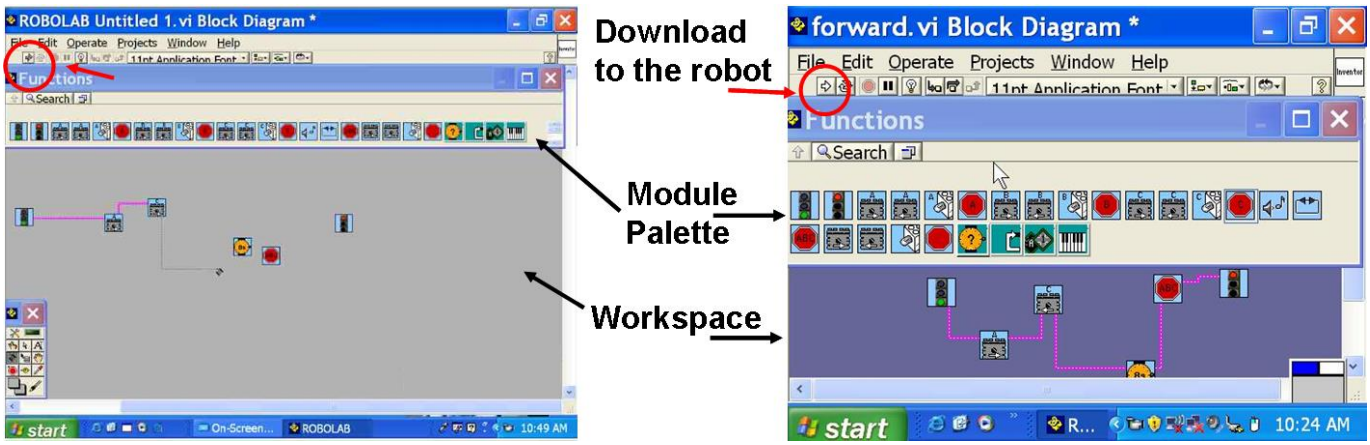

Figure 3: Robot programming environment (Left) at first session and (Right) after last session. The download button, module palette and workspace are indicated with arrows.

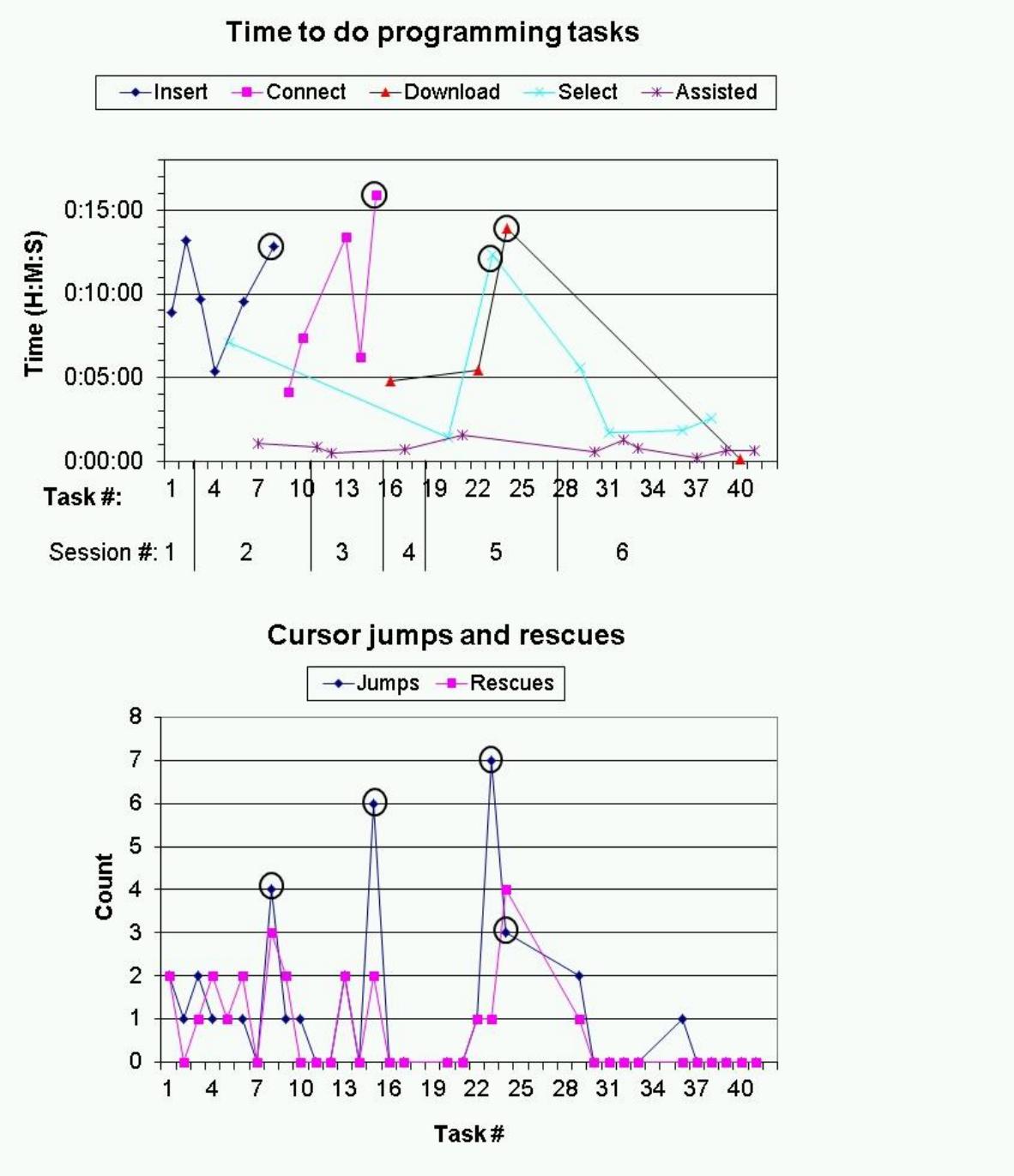

Figure 4. (Top) The amount of time that it took to do each programming task (i.e., insert module, connect module, download program, select module, assisted selection). (Bottom) The number of times that the cursor jumped and the number of times the investigator repositioned the cursor (rescues). Spikes in the time data which correspond to high number of cursor jumps are indicated by circles on each graph.