Robotics and Science Literacy: Thinking Skills, Science Process Skills and Systems Understanding

Florence R. Sullivan

Department of Teacher Education and Curriculum Studies, University of Massachusetts, Amherst, 813 N. Pleasant St., Amherst, MA 01027

Received 12 July 2006; Accepted 24 July 2007

Abstract: This paper reports the results of a study of the relationship of robotics activity to the use of science literacy skills and the development of systems understanding in middle school students. Twenty-six 11–12-year-olds (22 males and 4 females) attending an intensive robotics course offered at a summer camp for academically advanced students participated in the research. This study analyzes how students utilized thinking skills and science process skills characteristic of scientifically literate individuals to solve a robotics challenge. In addition, a pre/post test revealed that course participants increased their systems understanding, $t\,(21) = 22.47$, $p < .05$. It is argued that the affordances of the robotics environment coupled with a pedagogical approach emphasizing open-ended, extended inquiry prompts the utilization of science literacy-based thinking and science process skills and leads to increased systems understanding.

Keywords: computer science

Science literacy is a key goal of science education (American Association for the Advancement of Science [AAAS, 1993]; National Research Council [NRC, 1996]). The overarching aim of science education in K–12 settings is not to steer all students toward careers in science, but to create a populace knowledgeable enough about scientific ideas, modes of thinking, and scientific practices that they can make informed decisions about science and technology issues of global import. Robotics is an intellectually rich (see Barnes, 2002; Beer, Chiel, & Drushel, 1999; Chambers & Carbonaro, 2003; Flake, 1990; Flowers & Gossett, 2002; Garcia & McNeill, 2002; Klassner, 2002; Kumar, 2004; Miglino, Lund, and Cardaci, in press; Nourbakhsh, 2000; Resnick & Ocko, 1991; Ringwood, Monaghan, & Maloco, 2005; Sargent, Resnick, Martin & Silverman, 1996; Shimbakuro, 1989; Wagner, 1998; Weinberg, White, Karacal, Engel, & Hu, 2005), and popular student activity that has the potential to advance the goal of science literacy in K–12 science education. Despite this potential, little research exists regarding robotics learning in K–12 settings (Penner, 2001).

Correspondence to: F.R. Sullivan; E-mail: fsullivan@educ.umass.edu
DOI 10.1002/tea.20238
Published online 4 February 2008 in Wiley InterScience (www.interscience.wiley.com).

The purpose of this paper is twofold: first, I provide a definition of science literacy that serves as a framework for analyzing the relationship of robotics activity to science literacy skills and knowledge; and second, I report the results of a study focused on how academically advanced students used science literacy skills to solve robotics problems and the learning gains they achieved as a result of participation in the robotics course. In so doing, this research addresses both the gap in the K–12 robotics literature identified by Penner (2001) and it also responds to McGinnis and Stefanich's (2007) call for more research regarding the relationship between talented learners and the types of outcomes these students may achieve in specific science learning programs.

The Goals of Science Literacy

The AAAS (1993) and the NRC (1996) characterize science literacy as a complex, comprehensive, and interdisciplinary understanding of scientific, mathematical, and technological concepts contextualized by knowledge of the historical development of these concepts and the political and social effects thereby engendered. Scientific literacy will vary by individual, but the primary goals of literacy, stressed by the AAAS (1993) and the NRC (1996), are as follows: (1) the development of certain thinking skills including computation and estimation, manipulation and observation, communication, and critical-response skills; (2) the ability to engage in the activity of inquiry including ‘‘making observations, posing questions, planning investigations, reviewing what is already known in light of experimental evidence, using tools to gather, analyze, and interpret data, proposing answers, explanations, and predictions; and communicating the results’’ (NRC, 1996, p. 23); and (3) the development of an understanding of the common themes of science, identified by the AAAS (1993) as: (a) systems, (b) models, (c) constancy and change, and (d) scale.

Robotics and Science Literacy

Robotics learning is strongly linked to these three goals of science literacy. First, robotics study requires utilization of four of the six thinking skills characteristic of scientifically literate people—namely, computation, estimation, manipulation, and observation.[1] Second, students of robotics are engaged in science inquiry through both technological design and computer programming activities. Third, robotics teaches students about systems, one of the common themes in science education.

Thinking Skills. Central activities in robotics study include manipulation of tools, computation using the sensing devices, estimation using the software program modifying variables, and observation of the robotic device as it executes a program. For example, to design, build, and program a robotic device, students engage in manipulating the tools available in the robotics environment. These tools include Lego pieces, the Robolab software, and the sensors that collect and transmit data to the minicomputer at the heart of the robotic device. The tool-rich nature of the robotics environment enables students to take measurements and make computations regarding sensing device settings, estimate time variables, and write programs to test these variables. Students observe the results of these tests and make adjustments accordingly. Manipulation, observation, computation, and estimation represent four of the six thinking skills characteristic of scientifically literate people.

Technological Design and Science Inquiry. One of the primary activities students engage in while studying robotics is technological design. The NRC (1996) has postulated a close epistemic relationship between knowing in technology and knowing in science, asserting that technology as design is parallel to science as inquiry. In line with this assertion, and by way of arguing for the appropriateness of utilizing technological design projects to teach science inquiry, Lewis (2006) has pointed out the strong similarities in the process of designing an artifact to the process of conducting an inquiry in science. He outlines the similarities thusly: both are reasoning processes that bridge the gap from problem to solution; both take place (usually) in ill-structured domains, with uncertainty a characteristic of the starting point in reasoning about the problem; both rely on the cognitive processes of brainstorming and analogical reasoning; both rely on testing and evaluation of solutions; both require a certain amount of content knowledge; both proceed under constraints; both engage in forms of trial and error, reflection, learning from failure, and making adjustments in their hypotheses and in their lines of approach; and both are constrained by paradigmatic thinking.

Educational researchers have empirically validated this theoretical parallel between the process of design and the process of science inquiry (see Crismond, 2001). They have investigated the educational implications of the parallel, particularly as it regards cognitive processes and learning outcomes for students in science. For example, Roth (2001) has shown that the design of technological artifacts in science class prompts a successive chain of gestural, iconic, and discursive representations in the learner, which facilitates the formation of abstract understanding of the science ideas underlying the technological design project. In addition, researchers report that students learn more science content through engagement in science-based design activities than through traditional textbook/lecture/discussion methods (Hmelo, Holtan, & Kolodner, 2000; Penner, Giles, Schauble, & Lehrer, 1997).

Computer Program Debugging and Science Inquiry. In addition to technological design activities, students of robotics are also engaged in computer programming. During the process of debugging a written program, students engage in scientific reasoning through the enactment of science process skills. Science process skills include, but are not limited to: control of variables, hypothesis generation and testing, and the evaluation of solutions. Students may utilize all of these skills while debugging a software program or revising a technological design. For example, the robotics environment is a multivariate one (Suomala & Alajaaski, 2002), and to successfully solve a complex programming challenge, students must learn to hold variables constant while examining the effect of one variable on the execution of the program. If a student changes more then one element of a program at a time, that student will not be able to adequately account for the contribution of any one element to the overall functioning of the program. Therefore, the science process skill of holding variables constant to test the contribution of each independent variable on the dependent variable is a foundational aspect of successfully debugging in computer programming.

Additionally, students engage in hypothetical predictions as a means of debugging the program. Similar to student experiments in Schauble's (1996) study of the development of scientific reasoning in knowledge rich environments, debugging in robotics includes reasoning about the unobservable causes of directly observable phenomena. For example, students design a robotic vehicle and write a program to run on the vehicle. The robotic vehicle's execution of the written program is directly observable, but the underlying cause for errors in the execution of the intended functioning of the vehicle may not be. Unobservable factors that may be influencing the functioning of the vehicle are the interaction between wheel choice and the amount of friction caused by the surface the vehicle is running on, subtle changes to the quality of light in a room that may affect light sensor calibration, or conflicting programmatic commands that result in unintended vehicle motion, such as stuttering.

While debugging, students develop a hypothesis about why the program is not functioning properly; they then engage in conducting tests of their hypothesis and revising the program or the structural design of their robot accordingly. For example, in debugging a light sensor-controlled program, the student may develop the hypothesis that the quality of the light in the room has changed, and therefore the variable modifying the light sensor may no longer be correct. Armed with this hypothesis, the student may then check the light reading in the room. If the hypothesis proves correct the student will then change the modifying variable to reflect the accurate reading.

Finally, the debugging process includes an important feedback loop. Once students have programmed their robot, they test the program, and thereby receive immediate feedback on the quality of their program. This feedback initiates an iterative cycle of observation, hypothesis generation, hypothesis testing and evaluation of the solution. Feedback loops are not commonly viewed as a characteristic feature of science inquiry. However, as the teachers in Weinburgh's (2003) study noted, working scientists engage in an iterative process of inquiry utilizing feedback loops. Robotics activity may well teach something about science that is not usually learned because school science does not typically discuss feedback loops.

Systems Learning. A system is a collection of parts or processes (Penner, 2000). To understand a complex system, one must understand the ''causal interactions and functional relations between parts of the system and other systems'' (Hmelo et al., 2000, p. 248). Robotics is one such complex system, and consists of a minicomputer, sensors, Lego building pieces, motors, a computer, computer programming software, and an infrared device, termed the tower, that sends the computer program from the computer to the minicomputer. In studying robotics, students learn about the parts of the system, the functional relations between the computer program and the output devices (motors and tower), and the causal interaction between the computer program, the input devices (sensors), and the output devices (motors).

In summation, robotics is an activity that involves students in process and content learning activities that comprise three important goals of science literacy as defined by the AAAS (1993) and the NRC (1996). The processes of technological design and computer programming foster the utilization of thinking skills typical of a scientifically literate person and engage students in scientific reasoning and inquiry activities. The content of robotics study includes learning about systems—a common theme in science.

Given the large numbers of students engaged in robotics activity during out-of-school time and the clear relationship between robotics study and the development of science literacy, it is important to have an empirically valid understanding of the actual science literacy activities students engage in and the learning outcomes they achieve while studying robotics. Therefore, two questions were of primary interest in this study: (1) how do students utilize science literacy-based thinking skills and science process skills in solving a robotics challenge? (2) Does students' system understanding change as a result of participation in robotics study as measured by a pre–post test?

Methods

Setting and Participants

The data for this study were collected at one of the Center for Talented Youth's (CTY) summer camp programs for second- through sixth-grade students. CTY is a sponsored program of Johns

Hopkins University. It offers summer programs to students designated academically advanced based on their high scores on standardized tests. Specifically, CTY accepts second- through sixth-grade students who have scored in the 95th percentile or above on nationally normed tests administered by their respective schools. Scores on state tests that are categorized as advanced, distinguished, honors, exceeds, and so forth, may also be used to qualify a student for participation in CTY programs. The students in this study took part in a 3-week long, intensive summer robotics session. The CTY participants completed approximately 100 hours of robotics coursework in this 3-week period. There were two 3-week sessions. Twenty-six 11- and 12-year-old students participated in the study: 22 boys and four girls. Fourteen of the participants were White (12 boys and two girls), nine were Asian American (eight boys and one girl), and three were bicultural students (two White and Middle Eastern—one boy and one girl—and one White and Native American boy). Data on the socioeconomic status of the students' families were not available. However, the camp was expensive and none of the students received financial assistance to attend.

Materials

The robotics course used Lego Mindstorms construction kits and Robolab software. Lego Mindstorms features the RCX programmable brick, sensors, motors, and building pieces. The RCX has three inputs for sensor devices and three outputs for battery operated motors. Three of the sensing devices that may be connected to the RCX brick used in the class were the light sensor, the touch sensor and the rotation sensor. The light sensor is a device that measures the reflection of light off of a surface and displays the reading on the screen of the RCX. The light sensor may be programmed using either an absolute reference (a specific numerical reading) or a relative reference (darker than or brighter than the initial reading). A touch sensor is a device that responds to physical contact with an object. The touch sensor has two states it may be in: activated or not activated. The rotation sensor counts the rotations of a wheel attached to a motor and the RCX. Measurements may be taken with both the light sensor and the rotation sensor. These measurements are then used as variable modifiers of the corresponding sensor icons in the Robolab program to control the movement of the RCX based on feedback from the environment.

Pedagogical Approach and Curriculum

The investigator and one other full-time instructor taught the course. The class employed a mediated learning approach that included both direct instruction and open-ended, student-directed inquiry (Suomala & Alajaaski, 2002). Direct instruction included short lectures, software demonstrations and lab assignments. Open-ended, student-directed inquiry consisted of students working in pairs to solve problems posed as programming and design challenges. Each challenge required the students to build a structure and program the structure to accomplish a specific task. Students were free to design and build any structure and write any program they wished, provided that together, they met the overall requirements of the challenge. The investigator and the instructor facilitated student learning during open-ended inquiry by being available to answer student questions.

Students working with Robolab learn basic concepts in computer science. The basic computer science concepts covered in the class included the following: input/process/output, procedural flow, conditional statements, iteration, parallel processing, variables and subroutines. Regarding design concepts, students learned about gears, types of gears, and gear ratios. They also learned how to build sturdy structures. The course also emphasized systems learning. Creating and programming a robotic device is tantamount to creating a simple system. For example, one of the challenges the students were given in the course was to create a ping-pong ball-sorting machine. The purpose of the machine was to differentiate between dark- and light- colored ping-pong balls, sort them into different bins, and provide a count of the balls in each bin. This machine is both a mechanical and computerized system. Students utilized the Lego pieces to create a conveyor belt machine, they used the light sensors to determine the color of the ball for sorting purposes, and they wrote a program that counted and stored the values for the content of each of the bins. In addition to learning about systems through solving challenges, systems concepts were addressed in whole class discussions.

Research Design

Both observational and experimental methods were employed to address the research questions. In the last week of both sessions, each student individually engaged in a problem-solving activity that was videotaped. Students were instructed to think aloud during the problem-solving session. The videotaped robotics challenge consisted of three tasks that required the use of the light, rotation, and touch sensors. The tasks (described below) were to be executed sequentially. Pre–post tests of systems understanding were administered on the first and last day of each camp session.

Problem-Solving Challenge

The videotaped problem-solving sessions occurred on days 8 through 13 of instruction. The length of time needed to complete the individual challenge and the number of students in the class caused the sessions to span 3–4 days (3 days for the students in the first session and 4 days for the students in the second session). All of the relevant programming content needed by the students to solve the challenge had been covered by day 5 of the course. Due to time constraints, the problem solving session consisted of solving a robotics challenge that required minimal building. A partially constructed robot vehicle was provided to the students and they completed the construction by adding the appropriate sensors, wheels, and structural supports.

The students were individually asked to write a program that would cause the robot vehicle to follow a black line (that includes a 90 degree turn) on a paper track (see Appendix A), when the vehicle bumped into an object at the end of the track it should reverse motion for six inches, make a 360-degree turn and stop. There are three components to the challenge: the first is the line-following task, the second is the reverse-motion-on-contact task, and the third is the 360-degree turn task. The problem-solving task environment included the written robotics challenge, the Lego Mindstorms construction kit, a paper track with a 6-inch/15-cm ruler inscribed on it, and the Robolab software program. The following instructions were orally given to each student: "Please read the challenge. You may solve the challenge in any way you like. As you are solving the challenge, please say out loud what you are thinking." There was no time limit on the problem-solving session. Each student worked individually and no one received help in solving the challenge.

Problem Solving Data Analysis

Descriptive written logs of student activity during problem solving were created through multiple viewing of the videotapes. All verbalizations were transcribed. The unit of analysis was shift in activity (Chi, 1997; Jordan & Henderson, 1995). A shift in activity refers to an attentional shift from one element or task in the challenge to another element or task. Examples of a shift in activity are: (1) from writing the program to accessing the online context-sensitive help reference; (2) from writing the program to building the robot; (3) from running the program to measuring the quality of light in the room, etc. This unit of analysis was chosen as a means of systematically segmenting continuous student activity. Therefore, the activity log serves as a sequential record of the students' activities during problem solving.

A coding scheme was developed to analyze the segmented student activity. The coding scheme is derived directly from the AAAS (1993) report on science literacy and the NRC's (1996) national science education standards. It consists of the thinking skills described in the AAAS (1993) report, most directly related to robotics activity as follows: computation, estimation, manipulation, and observation. The coding scheme also includes the science process skills discussed by both the AAAS (1993) and the NRC (1996) most directly related to robotics study including: hypothesis generation, control of variables, hypothesis testing, and evaluation of solutions. The coding scheme is presented in Table 1.

All of the student logs were analyzed using the coding scheme. Frequency of usage of the thinking skills and science process skills were calculated from this data and are reported below. Additionally, a narrative description and analysis of one student's complete activity in solving the problem is provided in the results section.

Pre–Post Tests of Systems Understanding

The pre- and posttests consisted of five multipart questions (see Appendix B). The questions covered writing instructions for a robotic device, the flow of information through a device, the concept of input/process/output, the concepts of centralized and decentralized control of a system, and general systems knowledge. For the multipart question related to the concept of input/process/output students were asked to classify whether a described activity was an input, an output, or a processing activity. For the multipart question related to the concepts of centralized and decentralized control of a system, students were asked to classify whether or not a system was centrally controlled or decentralized. These questions had objectively right or wrong answers and

Table 1
Coding scheme

Skill	Type of Skill	Description of Code
Observation	Thinking	Student observes the execution of an algorithm by the robotic device, or the results of a measurement with a sensor.
Evaluation of Solution	Science process	Student makes an evaluative comment about the solution.
Estimation	Thinking	Student makes estimations regarding timing and speed variables included in her solution.
Hypothesis generation	Science process	Student generates an idea about why the robotic device is not functioning properly, based on observation of the functioning of the robot.
Hypothesis testing	Science process	Student tests the hypothesis he has generated.
Control of variables	Science process	Student changes only one element of a program at a time when testing a hypothesis.
Manipulation	Thinking	Student uses available tools to take measurements for use as variable modifiers in solving the robotics challenge.
Computation	Thinking	Student uses mathematical operations to assist in solving the problem.

could therefore be scored accordingly. One point was given for each correct response, and no points were given for incorrect responses in these two sections. There were a total of 20 points possible for these two questions.

The other three questions were open-ended in nature and respectively addressed student's ability to: (1) write programmatic instructions, (2) indicate the flow of information through a remote control device, and (3) describe a system. A rubric was developed to score these answers (see Appendix C). Students could earn a maximum of three points for a completely correct answer in each of the parts of the open-ended questions. There were a total of 15 points possible for these three questions. Two raters scored the responses using the rubric. Interrater reliability is reported below. The raters coded the pretest first and the posttest second.

Results

Thinking Skills and Science Process Skills—Coded Student Activity

I coded all of the student logs and a graduate student trained in the use of the coding scheme coded 25% of the logs, interrater reliability was calculated using Cohen's kappa, $\kappa = .86$, coding disagreements were resolved through discussion. Of the eight skills comprising the coding scheme, evaluation of solution and observation skills were used by all of the students. Twenty-five out of 26 students utilized manipulation, hypothesis generation, control of variables, and hypothesis testing skills; 24 of 26 used estimation skills in solving the problem and 11 of the 26 students used computational skills to solve the problem. In addition, 8 out of 26 students also neglected to control variables at various points during their problem solving session. Table 2 presents the frequency with which students deployed the coded thinking skills and science process skills. Frequency in this table is calculated as a ratio of total skills observed over all student observations. In other words, the table presents how often a skill was used on average by all of the students. The skills are presented in descending order of usage frequency.

In Table 3, I present an illustrative example of the use of each skill from the student activity logs. The examples are presented in descending order of the percentage of students who utilized the skill. Pseudonyms are used in lieu of participants' names in each of the illustrative examples. The left column of the example table is a descriptive notation of the student activity as derived from the videotape record. The right column of the example table is the transcription of the students' comments uttered at the same time as the described activity was being performed. A blank cell in the right column denotes student silence.

Table 2
Frequency of student use of thinking and science process skills

Skill	Type of Skill	% of Total Skill Usage
Observation	Thinking	30.73
Evaluation of solution	Science Process	19.27
Estimation	Thinking	13.01
Hypothesis generation	Science Process	9.76
Hypothesis testing	Science Process	8.94
Control of variables	Science Process	8.86
Manipulation	Thinking	8.13
Computation	Thinking	1.30

Table 3
Illustrative examples of student activity coding

Activity	Verbalizations
Observation—thinking skill	
Ricky runs the program. The robot runs forward on black and it stops when it sees white and it starts to turn to the right very slowly (away from the black line). Ricky watches the robot as it does this for a while.	
Evaluation of solution—science process skill	
Terry tests his program. This time the robot barely moves backward and stops.	That's too short. That is way too short... That's odd, that's really, really odd, .20 is too short and .34 is way too long.
Estimation—thinking skill	
Sam begins writing his program with motors A and C forward. He follows this with a wait for darker light sensor icon. Then he adds motor A reverse and motor C forward. Next he adds the unspecified (?) wait for time icon, but does not modify it. He then adds motors A and C forward, followed by a touch sensor. Next, Sam adds motor A reverse and motor C forward and another unspecified (?) wait for time icon. He then ends the program with the stoplight. Next he modifies the unspecified (?) wait for time icons, the first one he modifies with 0.25 seconds, and the second one with 0.50 seconds.	Ok, so A goes forward, C goes forward until it sees darker and then A goes backward, C goes forward... ummm for... then A goes forward and C until the touch sensor is activated and then A goes backwards, C goes forward for time and... Then let's see so, I'll make that ummm half a second, uh quarter second and half a second.
Hypothesis generation—science process skill	
Jeremy diagnoses the problem (not detecting the black line). He tries the robot program again. This time it stutters and struggles to move forward, finally it catches and moves forward.	Aah. That doesn't, it just keeps on going, so it's not detecting the black line. OK, I can see exactly what its mind is thinking.
Jeremy decides to spread the sensors further apart.	These light sensors are a little too close together, I am going to see if spreading these things apart helps anything.
Hypothesis testing—science process skill	
Jeremy tests the program, the robot is moving back and forth in place on the black line for a while, then it moves forward on the black line until it sees white at which point it turns to the right and follows the black line to the end of the track. J then touches the touch sensor and the robot backs up a bit and then begins to turn to the left for a long time. It is just spinning and then it switches directions and spins around, making several complete circles.	J: Anyway... ok. Does that count? T: That counts. J: OK, back up six inches from where I hit it.
Control of variables—science process skill	
Vanessa puts the robot on the floor and it still moves slowly.	It worked earlier. I'll put it on a higher power level...I'll do a little test to see if it is just...
Vanessa opens a new screen and writes a new tester program. The first thing she does is change the background color of the screen. She then adds motors A and C reverse for 6 seconds, leaving the power level at the default of five.	I personally like doing tests because it helps me make sure later, and if I save them I'll remember them later, just not the big programs.
Vanessa tests her program and observes the movement of the robot.	Yeah, I think it's just the power level is all that's wrong.

(*Continued*)

Table 3
(*Continued*)

Activity	Verbalizations
Manipulation—thinking skill	
Don then took a reading with the light sensor of the black line (41) and the white field (45). He took note of the readings on the Mindstorms track.	The black is 41 to 40 and the white is 45.
Computation—thinking skill	
Derek asks me for a ruler.	Derek: May I have a ruler and a pencil? T: Yes, you can use this piece of paper and. . .Derek: May I have a ruler? T: A ruler? Here's one right here.
Derek takes a wheel off of the robot and uses the ruler to measure the diameter of the wheel.	Must go six inches, so that is about 1 and 1/4 inches.
Derek writes down the formula for calculating the circumference of the wheel.	OK. . . wait, 16 isn't
Derek explains to me why he is using this formula.	Right now I am using the calculator, this is to, I am figuring out the circumference so that one rotation of the wheel. I need to learn how long that is. I need to know how long one rotation of the wheel is, so when I back up six inches, I need to know what the six inches is.
Derek uses the computer's scientific calculator to perform the calculation.	There should be a calculator here somewhere, normally these calculators, scientific calculators. . . 1 and 1/4 times pi, so one rotation will almost do it.

Narrative Description and Analysis of Student Activity

To show how students used the thinking skills and science process skills in solving the robotics challenge, I present an analysis of one student's activity in solving the problem. My analysis is italicized within the activity narrative and the use of each skill is labeled. To select a student case for closer analysis, I compiled a list of students who had used all of the eight thinking and science process skills of interest in this study. Seven students utilized all of the skills; they included one bicultural boy (White and Middle Eastern), one White girl, two Asian-American boys, and three White boys. I randomly selected one of these students by placing all of their pseudonyms in a container and then pulling one pseudonym out. I pulled the pseudonym of one of the White boys, Adrian. The following narrative describes and analyzes Adrian's activity, demonstrating how he used the thinking skills and science process skills to solve the problem. The narrative is organized into three segments, following Adrian's approach to solving each of the three tasks that comprise the challenge. Adrian begins his problem solution by tackling the second task. Therefore, his activity in solving the reverse-motion-on-contact is presented first, his activity in solving the line-following task is presented second, and his activity in solving the 360-degree turn task is presented in the third segment. Two figures are presented which graphically illustrate the narrative description of the changes Adrian made to his Robolab program as he worked toward a solution to the problem.

Reverse-Motion-on-Contact-Task. Adrian begins his problem-solving activity by completing the design of the partially constructed robot. He adds the touch sensor to the front of the RCX and the light sensor to the back of the RCX. Next, he turns his attention to programming the robot. The first thing he decides to do is to figure out how long it will take the robot to back-up 6 inches. Adrian states, "I am going to see how far 6 inches is." Adrian writes a program to have the robot back up for 4 seconds (*estimation*). He transfers this program to the robot and lines the robot up with the 6-inch ruler that is inscribed on the challenge track (*manipulation*). Adrian runs the program and watches the robot (*observation*), it moves far beyond 6 inches. *Here Adrian is using three of the thinking skills characteristic of scientifically literate people to develop a solution to one of the challenge tasks—estimation, manipulation, and observation. His plan is to see how far the robot will move when programmed to back up for 4 seconds. His observational activity centers on the use of the inscribed ruler on the track—this ruler is a tool that allows Adrian to experiment with a solution and it is his main reference for data collection.*

Next, Adrian picks up the robot and returns to his program. He revises the time variable to 1 second (*estimation*). Adrian sends the program to the robot; he aligns the robot with the ruler on the track (*manipulation*). Next he runs the program and watches the movement of the robot (*observation*). The robot moves further than 6 inches and longer than 1 second. Adrian retrieves the robot. He adds an all motors stop icon to his program. He then sends this program to his robot, aligns the robot with the ruler (*manipulation*), and runs the program (*observation*). The robot moves about one foot and stops. *In this sequence of moves, Adrian continues to use the three thinking skills of estimation, manipulation, and observation to assist him in developing a solution. Realizing through observation that the 4-second program is too long, he tries a 1-second estimate. Having observed that the robot did not stop after 1 second, he realized he needed to add an all motors stop command, and so he adds it. Through observation and the use of the ruler tool he is able to make increasingly accurate estimations, moving from a 4-second estimation to a 1-second estimation of the time variable which causes the robot to move about one foot.*

Adrian retrieves the robot. He revises the time variable of his program from 1 second to .50 seconds (*computation*). Adrian transfers his program to the robot, aligns the robot with the ruler (*manipulation*), and runs the program (*observation*). The robot moves just beyond the 6-inch ruler. Adrian revises his program one more time by changing the timing element from .50 seconds to .40 seconds (*estimation*). He sends this program to the robot, aligns the robot with the ruler (*manipulation*), and runs the program (*observation*). The robot moves backward the length of the 6-inch ruler and stops. *At this point, Adrian has solved the reverse-motion-on-contact portion of the overall challenge. He has done this by combining the use of four thinking skills—estimation, computation, observation, and manipulation. He started by selecting a time to test (4 seconds), and he observed the movement of the robot, measuring its trajectory with the ruler tool. When his first test proved that 4 seconds was too long, he estimated a 1-second time interval, he then observed that this 1-second program moved the robot about 1 foot, from this observation he used computation skills to reason that if he halved the timing element, the robot would move half as far. So, because the robot went about a foot in 1 second, it would go about 6 inches in half a second. Adrian revised the 1-second timing element to .50 seconds, this half-second program was very close to correct leading him to his final estimate of .40 seconds, which, when tested with the ruler, proved to be the correct time variable*. Figure 1 illustrates the successive changes Adrian made to his Robolab program described in this section of the narrative.

Line-Following Task. Having devised a solution to the reverse-motion-on-contact task, Adrian focuses on the line-following task portion of the challenge. He writes a program utilizing both the absolute and relative methods for taking light readings with the light sensor (*manipulation*). He modifies the absolute wait for dark sensor with 50 (*estimation*), and he follows this with the wait for brighter icon. Next, Adrian adds a task split to the beginning of the

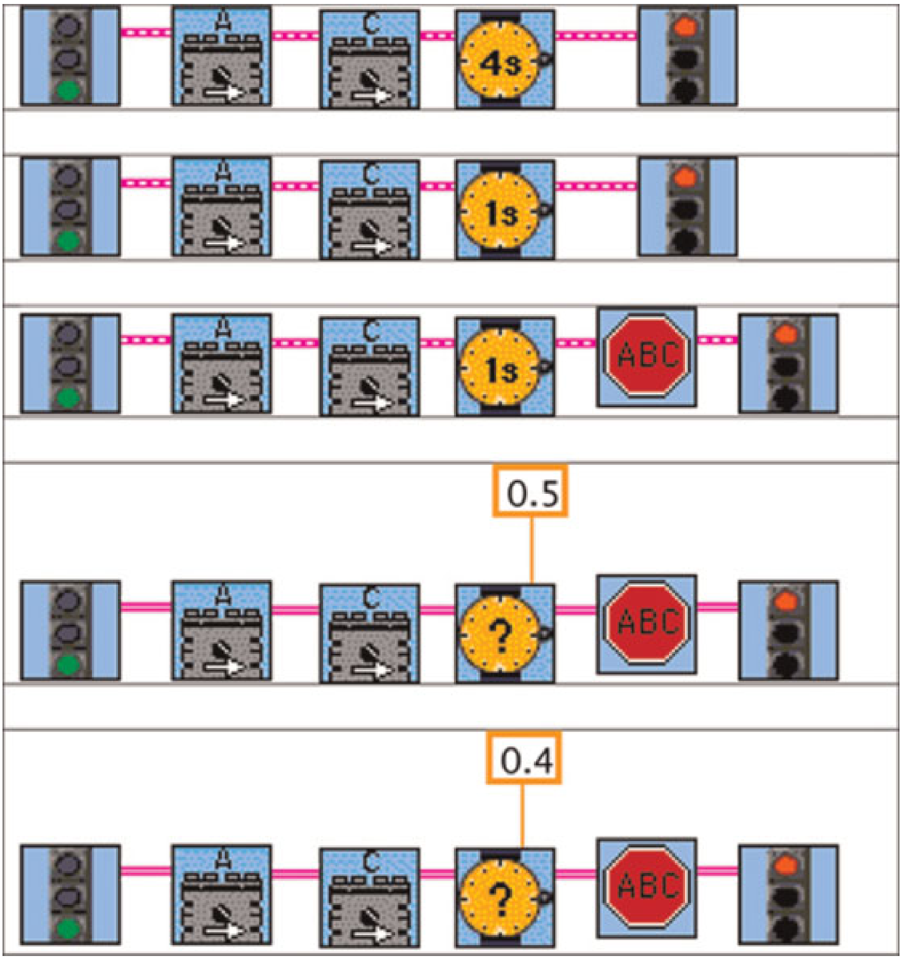

Figure 1. Adrian's successive changes to reverse-motion-on-contact task solution.

program and he attaches the line-following program to the bottom branch of the split, he attaches the reverse-motion-on-contact task program to the upper branch of the split. He then adds instructions after the reverse-motion-on-contact task program for the robot to turn for 4 seconds to the right (*estimation*). These last instructions refer to the third challenge task, which is for the robot to make a 360-degree turn. Adrian tests this program. The robot moves forward on the black line and makes a 90-degree turn about 3 to 4 inches beyond where the black line turns on the track (*observation*). *In this sequence, Adrian is again using the three thinking skills of estimation, observation, and tool manipulation. In this case, the tool that Adrian is manipulating is the light sensor. He uses an estimate of 50 for the reading of the reflection of light off of the track surface—the possible settings of the light sensor range from 1–100, so Adrian has selected the median setting for his first test. He observes the movement of the robot and notes that it is not functioning as he intended.*

Adrian lowers the power level of the first set of motors in the line-following task from full power (power level five) to power level three (*hypothesis generation and control of variables*). Adrian tests this new program (*hypothesis test*). *Here, Adrian has been able to observe that the robot did not follow the black line, yet, the reason for this error is not directly observable. There could be more than one reason why the robot did not function properly. For example, the variable modifying the light sensor may be wrong or the light sensor could be incorrectly positioned on the robot or the light sensor could be connected to the wrong input port, etc. At this juncture, Adrian utilizes the science inquiry skill of hypothesis generation regarding the nature of the problem to help him solve the problem. He hypothesizes that the robot was moving too fast for the light sensor to function properly, so he adjusts the speed of the motors powering the robot and tests his hypothesis by running the program again. By only changing one element of his program—the speed of the motors—Adrian is demonstrating the use of another inquiry skill: control of variables. Controlling the variables in this instance allows Adrian to either eliminate or isolate the source of the problem.*

Adrian tests this revised program, the robot executes the algorithm at a slower speed, but with the same result—the robot moves forward on the black line and makes a 90-degree turn approximately 3 to 4 inches beyond the place where the black line turns on the track (*observation*). Adrian says, ‘‘I think this thing [the light sensor] has to go on front (*hypothesis generation*).’’ He moves the light sensor from the back of the robot to the front of the robot (*control of variables*). Adrian tests his program again (*hypothesis test*). The robot now completely follows the black line on the track making the 90-degree turn in the correct spot, the robot then bumps into the wall, backs up 6 inches and makes a less than 180-degree turn. *Adrian has now solved the line-following portion of the challenge. Having eliminated the speed of the robot as the cause of the turn problem, and through observing the movement of the robot, Adrian develops a second hypothesis. This new hypothesis is that the light sensor is positioned incorrectly on the robot. Adrian adjusts the position of the sensor and tests his hypothesis by running the program again. Adrian's second hypothesis proves correct as the robot follows the black line as intended. In this segment, Adrian has again used the science inquiry skills of hypothesis generation, control of variables, and hypothesis testing to help him solve the problem.*

360-Degree Turn Task. Upon solving the line following task, Adrian turns his attention to the third task, which is to have the robot make a 360-degree turn and stop. Adrian observed that the initial 4-second time variable estimate resulted in a less than 180-degree turn; he revises the time variable from 4 seconds to 6 seconds (*estimation*). Adrian tests this program. The robot makes an approximately 180-degree turn (*observation*). He states, ‘‘It is not turning long enough, it is only turning about 180 degrees’’ (*evaluation of solution*). Adrian decides to double the time variable; he changes the program from 6 seconds, to 12 seconds (*computation*). Adrian tests this program. The robot executes the algorithm correctly by following the black line, backing up 6 inches when it bumps into an object and then makes a 360-degree turn and stops (*observation*). *Adrian has now completely solved the challenge. In this sequence, Adrian returned to his use of the thinking skills of estimation, computation, observation, and manipulation, and he also utilized the skill of evaluating his solution. After observing that the initial 4-second time variable was too short, he estimated a 6-second time variable. He observed the movement of the robot based on this estimation and evaluated his solution—noting that the robot was making about a 180-degree turn. Using this information, he reasoned that doubling the amount of time of the turn would double the distance of the turn. Therefore, he changed the timing variable from 6 seconds to 12 seconds. Adrian then tested this program and found that his reasoning was correct.* Figure 2 illustrates these successive changes Adrian made to his Robolab program in solving the challenge.

Figure 2. Adrian's successive changes to fall challenge solution.

Pre–Post Test of Systems Understanding

The pretest was administered on the first day of each session. All of the students completed the pretest. The posttest was administered on the last day of each session, four students did not complete the posttest; due to this, they were excluded from analysis. Two raters were used to score the tests. I scored all of the tests and a graduate student trained in the use of the open-ended questions rubric (see Appendix C) scored 25% of the tests. Interrater reliability for the open-ended questions rubric was achieved at $\kappa = .80$. Scoring disagreements were resolved through discussion. The means for the tests were calculated and a one-sample *t*-test was conducted to compare the means from pre to post. The mean score on the pretest of systems understanding ($M = 23.09$, $SD = 4.14$) was significantly different than mean score on the posttest ($M = 25.82$, $SD = 4.04$), $t\,(21) = 30.04$, $p < .05$. Further analysis revealed a medium to large effect size for the difference in understanding from pre to post, Cohen's $d = .67$, the observed power was .930.

Discussion

The results of this study indicate two main findings. First, the study demonstrates how students participating in this robotics curriculum utilized the thinking skills and science process skills associated with scientifically literate people to solve a robotics problem. Second, the study found that students' systems understanding improved as a result of participation in this robotics course. Regarding the first finding, it is important to note that all or nearly all of the students utilized seven out of the eight thinking skills and science process skills described in the coding scheme. All of the students made observations and evaluated their solutions; all but one of the students used tools in the environment, generated and tested hypotheses while controlling variables, and all but two of the students used estimation skills in solving the robotics problem. The only thinking skill that was not used by the majority of the students was computation. Both the line-following task and the reverse-motion-on-contact task could have been solved using computation. They could also be solved using estimation. It is plausible that students found the estimation skill, as demonstrated by Adrian's successive approximations in the reverse-motion-on-contact task, to be a simple and successful strategy in solving distance-traveled problems. It is arguably easier to estimate distance traveled with time variables than it is to calculate distance traveled using the circumference of the wheel as done by Derek (see Table 3). Such a computational method is more accurate and certainly more sophisticated, but it also requires the use of the rotation sensor. The estimation method also works and it does not require the use of a sensor, making it an effective, yet simpler, skill to deploy.

In addition to using thinking skills and science process skills to solve the robotics challenge, students who took part in this robotics course also improved their systems understanding. The pre- and posttests results show a clear gain in systems understanding. The effect size of the change in understanding is medium to large (Cohen, 1988), and the power is very good. However, a practice effect cannot be ruled out, as the same test was used for pre- and post assessment. Another limitation of these findings is the fact that the participants in the study are not a representative sample of the U.S. middle school population. The students who took part in this study were primarily male, labeled gifted, and from financial backgrounds that enabled them to attend an expensive summer camp. Future studies with a more representative sample to verify the results of this study are warranted.

To what may we attribute the learning outcomes and gains demonstrated by the students in this study? I contend that it is the pedagogical approach coupled with the design of the robotics environment itself that leads to specific activity structures and modes of participation which, in turn, engender the use of these thinking and science process skills, resulting in increased systems understanding. Gibson (1986) has developed an ecological theory of perception that argues that specific perceivable environmental factors make possible, and indeed encourage, specific activity. For example, a table makes possible the action of lying things upon it. A window makes possible the activity of opening or closing it. Gibson (1986) terms these factors affordances. The design of the robotics environment affords certain activity to be undertaken by students participating in robotics study including the use of manipulation and observation thinking skills, and the use of hypothesis generation, hypothesis testing, and evaluation of solution science process skills. This is particularly true when the instructional design of the robotics activities allows for open-ended, extended inquiry on the part of the student. In other words, the structure of the robotics environment combined with specific pedagogical approaches fosters the use of these skills. In addition, while the use of computation and estimation thinking skills and the science process skill of control of variables are not directly afforded by such participation in robotics study, they are encouraged, as is greater systems understanding.

There are two design aspects of the robotics environment and one instructional design aspect of the pedagogical approach utilized in this study that afford activity structures and modes of participation that result in the use of thinking skills and science process skills as follows: (1) the tool-rich nature of the environment, (2) the immediate feedback built into the system, and (3) the open-ended and extended nature of student inquiry. Further, the iterative nature of debugging activity results in an emergent understanding of the science process skill of control of variables. Using Adrian's activity as an example of these affordances, I discuss each in turn.

Tool-Rich Environment

The most basic activity in the robotics environment requires tool use. Although this study has focused primarily on the use of the sensors as tools in the environment, there are a number of other elements in the environment that may be thought of as tools. For example, the Robolab software program is a tool for writing computer programs, and the RCX is a tool for executing the computer program. The tower is the tool that sends the computer program from the computer to the minicomputer. Students participating in the most basic robotics activity of writing a program, sending the program to the robotic device, and executing the program on the device must learn to manipulate all of these tools, and they observe the results of these manipulations.

In addition to this most basic activity, students may also use the sensors. As aforementioned, the sensors are the primary measurement tools. The light sensor and the rotation sensor not only allow students to take measurements, gather readings, and engage in computation, but they also give students an appreciation of specific instrumentation for specific types of inquiry, for example, to program a robotic vehicle to read the difference between a white line and a black line, the best tool to use is a light sensor. While using the tools in the environment, students observe the effects of the tools and the program they have written. Students make decisions about when to use tools and when to use estimation skills in solving the problem.

This type of tool use and decision making is clearly illustrated in the analysis of Adrian's activity. Adrian utilized two tools in developing his solution: the ruler and the light sensor. He used the ruler in conjunction with a computer program featuring a time variable to arrive at a solution to the reverse-motion-on-contact task. Through observation and the use of the ruler tool, Adrian was able to make better and better estimations of the time variable. Adrian made the decision to use the ruler and the time variable instead of the ruler and the rotation sensor. He may have done this because the time variable estimation was a simple, yet effective, means of developing a solution. Adrian also utilized the light sensor as a tool for solving the line-following task. The light sensor is the most appropriate and precise tool for solving such a task. In selecting the light sensor, Adrian demonstrated an awareness of using the right tool for a specific task.

Immediate Feedback

The immediate feedback built into the system refers to two elements. First, to send a program from the computer to the minicomputer via the tower, there must be no language errors. In other words, if a needed element is missing, the Robolab program will indicate that there is a programming error and will give specific instructions as to where the error is located. For example, a common error occurs with the wiring of the program. The metaphor used in the Robolab software is one of electrical connections; each icon is connected to the next by means of a ''wire.'' The wire must connect the icons from a specific region of the icon (upper right quadrant) to a specific region of the next icon (upper left quadrant), if the wire is connected from the wrong region or to the wrong region the connection is not made (even though it visibly appears to be connected on the computer screen). If a student attempts to send a program with wiring problems from the computer to the minicomputer, she will receive an error message with instructions regarding the location of the error, at which point she may correct the error. This feedback leads to student evaluation of their solution.

The second, and more important, aspect of the immediate feedback built into the system is the fact that a program written in Robolab, that is free of language errors, may be immediately sent to the minicomputer to be executed. Robolab computer code does not need to be compiled. This capability allows for immediate feedback on the efficaciousness of one's computer program. Students observe the execution of their program and, if the program is not functioning properly, they evaluate their solutions against the behavior of the robotic device. This behavioral feedback and evaluation then leads to the generation of a hypothesis as to why the program is not working, a revision of the program and hypothesis testing.

This feedback loop of observation, evaluation, hypothesis generation, and hypothesis testing is clearly demonstrated in Adrian's work. For example, in solving the line-following task Adrian wrote a program and observed the functioning of the robot. When the robot did not function as intended on the first try, Adrian hypothesized that the robot may have been moving too fast for the light sensor to function properly, he adjusted the speed of the robot and tested his hypothesis. Observing the same results on the second trial allowed Adrian to eliminate robot speed as the cause of the problem. Also, by watching the slowed movement of the robot, Adrian was able to develop a second hypothesis regarding the placement of the sensor on the robot. Adrian repositioned the sensor on the robot and tested his hypothesis. This time his hypothesis was correct. The immediate feedback made it possible for Adrian to develop and test a couple of hypotheses until he arrived at the solution.

Open-Ended and Extended Inquiry

All of these activities afforded by the robotics environment are enabled by a pedagogical approach that emphasizes open-ended and extended inquiry. Certainly it is possible for a teacher to prescribe robotics activity that does not require the students to think. For example, step-by-step instructions that show students exactly which icons to select in sequence to write a specific program will not engender the type of thinking discussed here. However, allowing the students to decide how to solve a problem and giving them time to pursue their solutions will.

Debugging Activity

Further, the iterative nature of debugging activity assists students in developing their understanding of control of variables. Through an iterative process, students narrow down their ideas as to why the program is not working, leading eventually to a solution that rests on changing one variable at a time. The data reported here supports this interpretation. Most of the time, students were able to hold the variables constant in testing a hypothesis. Occasionally, some students would change two variables at a time, but this activity was less common, and the students exhibiting this activity, also controlled variables at other times. So, although the environment itself does not necessarily prompt control of variables, successful solutions depend on narrowing down the problem. This activity is seen in Adrian's example. Adrian looked at only one factor at a time in solving the line-following task. First he changed the speed of the motors, and when that did not solve the problem, he changed the position of the sensor on the robot. By focusing on one variable at a time, Adrian was able to eliminate certain factors and arrive at a solution. All but one of the students who took part in this study successfully controlled variables as they narrowed down the problem.

Finally, system understanding is promoted in robotics activity as no one element of the robotics system can operate on its own. The Robolab software computer programs mean nothing if there is no RCX on which to execute them, the RCX has little purpose without a Robolab generated program to execute. The only purpose of the tower is to transmit the Robolab program to the RCX. Understanding of this system is endemic to the most basic robotics activity, and greater systems understanding is developed as activity moves from basic programs to more complex challenges involving the sensors.

The final point I would like to address is the applicability of these findings beyond the robotics environment. Because it is specific features of the environment coupled with a specific pedagogical approach that leads to science literacy outcomes, any technological design activity that features tool use, immediate environmental feedback, open-ended, extended inquiry, and an iterative activity structure centered on creating a working system, will most likely result in similar outcomes for students with similar profiles as those involved in this study.

Implications and Conclusion

This study has identified important science literacy learning outcomes of robotics study—the utilization of thinking skills and science process skills typical of scientifically literate individuals and gains in student understanding of systems concepts, a common theme in science. I argue that these outcomes are a result of both the affordances of the robotics environment itself and a pedagogical approach that emphasizes open-ended, extended inquiry. The participants in this study were not representative of the U.S. middle school student population. Therefore, future studies should focus on replicating these findings with more diverse groups.

Furthermore, these results provide important information to curriculum developers, after school science program coordinators, and middle and high school science teachers regarding the increasingly popular, out-of-school, student activity of robotics study. A major goal of all science teaching is the development of student understanding of, and facility with, the process of science inquiry. Robotics provides a hands-on method of teaching thinking skills and science process skills through the technological design and computer programming activities that inhere in any robotics project. This report provides a theoretical and empirical basis for the development of well-crafted curricular materials that emphasize the relationship of robotics learning to the thinking skills, science process skills, and systems understanding associated with science literacy. Future research studies should focus on the extent to which robotics study prepares students for further learning about systems and for employing thinking and science process skills to solving problems in other environments.

The author would like to thank Xiaodong Lin, Celia Oyler, Flavio Azevedo, and Huang-Yao Hong for valuable feedback in the preparation of this manuscript.

Notes

[1]The thinking skills of communication and critical response, as defined by the AAAS (1993), generally refer to critical analysis and discussion of scientific reports, these activities are not inherently part of robotics activity.

Appendix B

Pre–Post Test

DIRECTIONS: Answer the questions below on this paper. Use program-like language whenever possible.

1. Imagine you are a robot maker of the future. Create the robotic instructions for the tasks below. Use program-like language whenever possible. Use the back of this paper.
   a. Take out the garbage from your kitchen to the outdoor garbage can (if you live in an apartment house, to your garbage shoot).
   b. Ask you for a drink and give you the drink you requested. (The robot has a refrigerator in its belly with your three favorite drinks!)
   c. Add a routine to the above program that causes your most favorite drink to be the first choice the robot gives you. Also, make routine to change your favorite drink in the computer's memory.
2. Using arrows and words, label the diagram of a tv remote control showing how information (your tv channel selection) flows through the device. Start with the input keys.

3. Below is a list of actions. Check off whether each action is an input of information, the output of information or the processing of information.

Input Output Processing

a. A beep from your computer
b. Pressing a button on your phone
c. A printout from your printer
d. Thinking about which soda you want from a machine.
e. A picture on your computer monitor
f. Talking into a cell phone
g. A calculator adding a sum
h. The movement of a remote controlled car.
i. The ringing of your alarm clock
j. Your digestion of breakfast.

4. Below is a list of systems. Each system is either a centrally controlled system (one piece controls info to all of the other pieces) or a decentralized system (no one piece controls the flow of information to all of the others).

	Central Control	Decentralized
a. The Internet		
b. A personal computer system		
c. E-mail		
d. A soda machine		
e. A group of ants collecting food.		
f. The U.S. Post Office		
g. A remote controlled car		
h. Geese flying in formation		
i. A marching band		
j. Synchronous fireflies		

5. Describe one system with which you are familiar. It could be a computer system, a mechanical system, biological system or any other of your choice. Explain what behaviors make it a system and how it functions as a system. Continue on the back of this sheet if necessary.

Appendix C

Rubric for Open-Ended Questions

Question No.	Master Level Score = 3	Competent Level Score = 2	Novice Level Score = 1
1a	Uses programmatic terms to explain situation in procedural language	Uses some programmatic terms to explain situation in procedural language	Partially explains situation in procedural language
1b	Uses programmatic terms to explain situation in procedural language using conditional statements	Uses some programmatic terms to explain situation in procedural language using conditional statements	Partially explains situation in procedural language using conditional statements
1c	Uses programmatic terms to explain situation in procedural language using conditional statements and variables	Uses some programmatic terms to explain situation in procedural language using conditional statements and variables	Partially explain situation in procedural language using conditional statements and variables
2	Indicated flow of input, output in processing correctly	Indicated either flow of input, flow of output or some sort of processing partially correctly	Labeled the remote
5	Demonstrates how the system operates and identifies how system terms apply to the system. Correctly describes entities, their relationships and the overall functioning that results.	Demonstrates how the system operates and identifies how some of the entities are related and functions.	Demonstrates somewhat how the system operates and either mentions entities or discusses functioning.

References

American Association for the Advancement of Science. (1993). Benchmarks for science literacy. New York: Oxford Press.

Barnes, D.J. (2002). Teaching introductory Java through LEGO MINDSTORMS models. ACM SIGCSE Bulletin, 34, 147–151. Retrieved May 5, 2006 from http://portal.acm.org.silk.library.umass.edu/.

Beer, R.D., Chiel, H.J., & Drushel, R.F. (1999). Using autonomous robotics to teach science and engineering. Communications of the ACM, 42, 85–92.

Chambers, J.M., & Carbonaro, M. (2003). Designing, developing, and implementing a course on lego robotics for technology teacher education. Journal of Technology and Teacher Education, 11, 209–241.

Chi, M.T.H. (1997). Quantifying qualitative analyses of verbal data: A practical guide. Journal of the Learning Sciences, 6, 271–315.

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Hillsdale, NJ: Lawrence Erlebaum.

Crismond, D. (2001). Learning and using science ideas when doing investigate-and-redesign tasks: A study of naïve, novice and expert designers doing constrained and scaffolded design work. Journal of Research in Science Teaching, 38, 791–820.

Flake, J. (1990). An exploratory study of lego logo. Journal of Computing in Childhood Education, 1, 15–22.

Flowers, T.R., & Gossett, K.A. (2002). Teaching problem solving, computing, and information technology with robots. Journal of Computing Sciences in Colleges, 17, 45–55.

Garcia, M.A., & Mc-Neill, H.P. (2002). Learning how to develop software using the toy LEGO mindstorms. Proceedings of the 7th annual conference on Innovation and technology in computer science education, Aarhus, Denmark, p. 239. Retrieved May 5, 2006 from http://portal.acm.org.silk.library.umass.edu/.

Gibson, J.J. (1986). The ecological approach to visual perception. Hillsdale, NJ: Lawrence Erlbaum.

Hmelo, C., Holton, D.J., & Kolodner, J.L. (2000). Designing to learn about complex systems. Journal of the Learning Sciences, 9, 247–298.

Jordan, B., & Henderson, A. (1995). Interaction analysis: Foundations and practice. Journal of the Learning Sciences, 4, 39–103.

Klassner, F. (2002). A case study of LEGO Mindstorms'™ suitability for artificial intelligence and robotics courses at the college level. ACM SIGCSE Bulletin, 34, 8–12. Retrieved May 5, 2006 from http://portal.acm.org.silk.library.umass.edu/.

Kumar, A.N. (2004). Three years of using robots in an artificial intelligence course: lessons learned. Journal on Educational Resources in Computing, 4, 1–15.

Lewis, T. (2006). Design and inquiry: Basis for an accommodation between science and technology education in the curriculum? Journal of Research in Science Teaching, 43, 255–281.

McGinnis, J.R., & Stefanich, G.P. (2007). Special needs and talents in science learning. In S.K. Abel & N.G. Lederman (Eds.), Handbook of Research in Science Education (pp. 287–318). Mahwah, NJ: Erlbaum.

Miglino, O., Lund, H.H., & Cardaci, M. (1999). Robotics as an educational tool. Journal of Interactive Learning Research, 10, 25–48.

National Research Council. (1996). The National Science Standards. Retrieved April 5, 2005 from http://www.nap.edu/readingroom/books/nses.

Nourbakhsh, I.R. (2000). Robots and education in the classroom and in the museum: On the study of robots, and robots for study. Proceedings of workshop for personal robotics for education, institute of electrical and electronic engineers, international conference on robotics and automation, Salt Lake, Utah. Retrieved May 5, 2006 from http://portal.acm.org.silk.library.umass.edu/.

Penner, D. (2000). Explaining systems: Investigating middle school students' understanding of emergent phenomena. Journal of Research in Science Teaching, 37, 784–806.

Penner, D. (2001). Cognition, computers, and synthetic science: Building knowledge and meaning through modeling. Review of Research in Education, 25, 1–35.

Penner, D.E., Giles, N.D., Lehrer, R., & Schauble, L. (1997). Building functional models: Designing an elbow. Journal of Research in Science Teaching, 34, 125–143.

Resnick, M., & Ocko, S. (1991). LEGO/logo: Learning through and about design. In I. Harel & S. Papert (Eds.), Constructionism (pp. 141–150). Norwood, NJ: Ablex.

Ringwood, J.V., Monaghan, K., & Maloco, J. (2005). Teaching engineering through lego mindstorms. European Journal of Engineering Education, 30, 91–104.

Roth, W.M. (2001). Learning science through technological design. Journal of Research in Science Teaching, 38, 768–790.

Sargent, R., Resnick, M., Martin, F., & Silverman, B. (1996). Building and learning with programmable bricks. In Y. Kafai & M. Resnick (Eds.), Constructionism in practice: Designing, thinking, and learning in a digital world (pp. 161–173). Mahwah, NJ: Lawrence Erlbaum.

Schauble, L. (1996). The development of scientific reasoning in knowledge-rich contexts. Developmental Psychology, 32, 102–119.

Shimbakuro, G. (1989). A class act: Junior high student lego and logo. The Computing Teacher, 16, 37–39.

Suomala, J., & Alajaaski, J. (2002). Pupil's problem-solving processes in a complex computerized learning environment. Journal of Educational Computing Research, 26, 155–176.

Wagner, S.P. (1998). Robotics and children: Science achievement and problem solving. Journal of Computing in Childhood Education, 9, 149–165.

Weinberg, J.B., White, W.W., Karacal, C., Engel, G., & Hu, A.P. (2005). Multidisciplinary teamwork in a robotics course. ACM SIGCSE Bulletin, 37, 446–450. Retrieved May 5, 2006 from http://portal.acm.org.silk.library.umass.edu/.

Weinburgh, M. (2003). Confronting and changing middle school teachers' perceptions of scientific methodology. School Science and Mathematics, 103, 222–232.