ROBIN: Using a Programmable Robot to Provide Feedback and Encouragement on Programming Tasks

Ishrat Ahmed, Nichola Lubold, and Erin Walker

Computing, Informatics, and Decision Systems Engineering,
Arizona State University, Tempe, AZ
{iahmed7,nlubold,eawalke1}@asu.edu

Abstract. LEGO Mindstorms robots are a popular educational tool for teaching programming concepts to young learners. However, learners working with these robots often lack sufficient feedback on their programs, which makes it difficult for them to reflect on domain concepts and may decrease their motivation. We see an opportunity to introduce feedback into LEGO Mindstorms programming environments by having the robot itself deliver feedback, leveraging research on learning companions to transform the programmable robot into a social actor. Our robot, ROBIN, provides learners with automated reflection prompts based on a domain model and the student's current program, along with social encouragement based on a theory of instructional immediacy. We hypothesize that by having the robot itself provide cognitive and social feedback, students will both reflect more on their misconceptions and persist more with the activity. This paper describes the design and implementation of ROBIN and discusses how this approach can benefit students.

Keywords: LEGO Mindstorms, Feedback, Immediacy

1 Introduction and Background

Learning through building and programming robots can lead to improvements in a learners' computer science (CS) skills and motivation [1, 2]. LEGO Mindstorms is a programmable robotics kit that is widely used as an educational tool. Programs written in this kit provide visible results, giving the learner hands-on experience to understand the fundamentals of abstract CS concepts like loops and conditional statements. Watching the direct effect of their coding on the robot provides a motivating learning environment for participating students [2].

While there is some evidence that LEGO Mindstorms can be used to improve domain learning and motivation [2, 3], the lack of feedback in the rapid compile-run-debug cycle is considered to have a negative impact on students. When a learner tries to load a program into the robot and the program does not behave as expected, several issues may lead to frustration and inhibit learning. For example, the learner may be unaware of an error they made or may not be motivated to explore the cause of the error, and repeated failed attempts may

further discourage the learner. In such scenarios, feedback which can draw the learner to a correct problem-solving path could greatly influence students' CS skill and motivation.

A great deal of educational research has focused on designing effective prompting and feedback for various learning environments [4, 5]; however, feedback delivered by programmable robots is still under-explored. In this work, we introduce a feedback system into the LEGO Mindstorms programming environment using a learning companion paradigm. Learning companions are based on the framework of peer learning, where a learner and an agent work together to solve problems [6]. We design a feedback model for ROBIN based on a learning-by-doing approach [7], where reflective feedback is provided by the programmable robot itself, transforming it into a learning companion. Learning companions have direct interaction with the learner, so behaviors that enhance interpersonal communication, such as immediacy [8], can affect the nature and quality of the learning environment. By leveraging social behavior like immediacy into the reflective feedback provided by the programmable robot and encouraging a dialogue, we expect the learner to develop rapport with the robot. Due to this rapport, when students receive feedback from ROBIN, they may be more motivated to reflect on their misconceptions and solve encountered errors.

In this paper, we discuss the design and implementation of ROBIN, proposing a novel type of learning companion where the agent is (i) a programmable robot that is (ii) responsible for providing feedback, and (iii) socially engages with students. Future work will evaluate the use of ROBIN in order to investigate the potential of a feedback system delivered by a programmable robot.

2 System Description

ROBIN consists of an iPhone mounted on a Lego Mindstorms EV3 robot, and a desktop application written in Java. To program ROBIN, the student uses the EV3 development environment that comes with the LEGO Mindstorms robotics kit, a graphical interface that models programming as a process where the user drags and drops different sets of blocks (representing programming steps) on a screen to complete a program. For example, if the user adds a *Move Steering* block, ROBIN can go forward, backward, turn, or stop depending on the input parameters (i.e. speed, power). Once done, the finished program is downloaded to the robot via a connection cable. ROBIN executes the program independently and initiates a spoken-language interaction based on the correctness of the program. Primarily, simple programs are used, for example, moving forward, backward or picking an object; but more complicated programs can be used as well. Figure 1 shows an image of ROBIN and Figure 2 shows a sample program.

In the desktop application, the user selects the file location of the EV3 program that they are modifying. A *filewatcher* program continuously monitors whether the selected file is being modified and saved. Once saved, the EV3 file is unzipped to create an XML file containing program instructions. For example, if the rotation parameter in the block is set as 3, the XML file contains

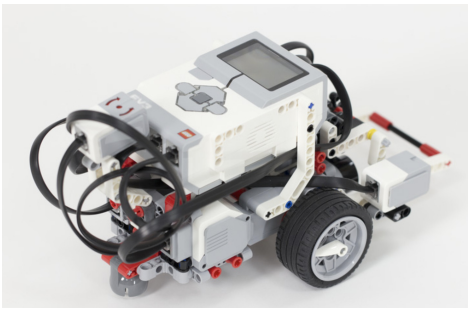

Fig. 1: ROBIN

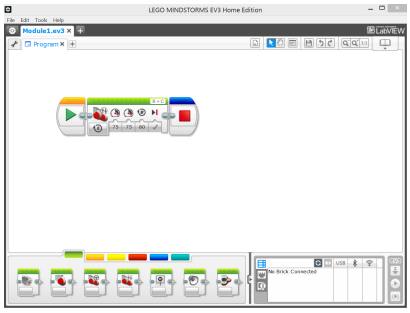

Fig. 2: Sample Block Program

<ConfigurableMethodTerminal ConfiguredValue="3">. This XML is parsed using a *parser* and information pertinent to the correctness of the program, such as which blocks were used, number of blocks used, and input values, is extracted.

Next, ROBIN uses a domain model to compare these extracted values and assess the correctness of the program. The domain model consists of several constraints, specific to each problem, that outline the parameters of the correct solution. For example, the first problem asks students to program ROBIN to go 80 cm forward. The program requires a *Move Steering* block to go forward, and a rotation parameter within the range 6.2 and 6.5. A *problem analyzer* checks for several different kinds of errors based on the domain model, including choice of an incorrect block, and incorrect input parameters for a block. Based on the number of errors, the student program is then labeled as being in one of three states: Correct (C; 0-10% incorrectness), Partially Incorrect (PI; 10-50% incorrectness) and Incorrect (I; more than 50% incorrectness). For example, to verify if ROBIN was correctly programmed to move 80 cm forward, we use the following conditions:

IF ((6.2 ≤ rotation ≤ 6.5) & block=Move Steer) THEN program=C
IF ((5 ≤ rotation ≤ 6.2) & block=Move Steer) THEN program=PI
IF (rotation < 5 || rotation > 6.5) THEN program=I

The program state and identified errors are used by a *Dialogue Manager* which runs as an iOS application on the iPhone mounted on ROBIN. The iOS application utilizes the accelerometer on the iPhone to detect when ROBIN has finished executing the learner's program. Once executed, the dialogue manager retrieves the identified errors for the program and determines an appropriate response using AIML or Artificial Intelligence Markup Language (AIML) [9]. The basic building block of AIML is a *category* containing a *pattern* and a *template*. The *pattern* is matched with a system/user input and ROBIN responds with the *template* as its answer. AIML has a collection of *pattern-template* pairs that help to generate the feedback for ROBIN. Our system utilizes a web-based service to create the AIML called Pandorabots [10].

Based on the program state, the dialogue manager generates the first feedback message of a two-turn feedback exchange (i.e., the system speaks, the

learner speaks, and then the system speaks one more time). Because this feedback is interactive, learners may feel a greater sense of connection with ROBIN while receiving the feedback. For example, in the *incorrect* state feedback is designed to interactively inspire thinking. If the learner entered the "wrong rotation parameters", ROBIN might say "*We are almost there but I think one of the parameters might be wrong! Which parameter do you think?*" Feedback for the *partial incorrect* state might provide encouragement along with a prompt to try again; "*Great job! I went half-way, why don't you try again?*" If the learner tries again and the program remains in the *partial incorrect* state, similar feedback is provided again by ROBIN; "*Any thoughts on why I am only going half-way?*" This ensures variability in the feedback. Finally, a *correct* program might generate feedback with praise; "*Keep up the good work!*"

ROBIN outputs the response through the built-in microphone on the iPhone. The initial feedback response is designed to initiate a dialogue with the learner. After ROBIN speaks, the dialogue manager captures the learner's response using automatic speech recognition (ASR) and generates a feedback using AIML. All the dialogues reflect the following verbal immediacy behaviors to foster rapport: praising the students, addressing them by name, and using "we"/ "our" instead of "i"/"your" during conversation. These behaviors should further foster social engagement and rapport with the system [11,12]. A complete dialogue-based interaction between ROBIN and a fictional student, Melissa, in the context where ROBIN did not go 80 cm forward due to a rotation parameter error follows:

ROBIN: We are almost there but I think one of the parameters might be wrong! Which parameter do you think Melissa?

Melissa: Ha, I see! Do I change the rotation parameter?

ROBIN: Wow, you got that right. Let's do it!

We designed the templates with a randomized list of relevant responses to avoid repetition and support variation in the feedback. We also designed the dialogue to handle ASR errors or failed interpretations of the user dialogues.

3 Conclusion

We introduce a novel approach for a social feedback system in programmable robots based on the concept of the learning companion. In this paper, we discuss our initial step towards the investigation of the design, implementation, and evaluation of ROBIN. In the future, we plan to run experiments exploring the effects of our design of social interaction and feedback on student motivation and CS learning gains.

Acknowledgments. The authors gratefully acknowledge Nicholas Martinez for his assistance in developing the initial version of the system. Support for this work was provided by NSF CISE-IIS-1637809.

References

1. Lawhead, Pamela B., et al. "A road map for teaching introductory programming using LEGO mindstorms robots." ACM SIGCSE Bulletin. Vol. 35. No. 2. ACM, 2002.
2. Lykke, Marianne, et al. "Motivating programming students by problem based learning and LEGO robots." Global Engineering Education Conference (EDUCON), 2014 IEEE. IEEE, 2014.
3. Alvarez, Ainhoa, and Mikel Larranaga. "Using LEGO mindstorms to engage students on algorithm design." Frontiers in Education Conference, 2013 IEEE. IEEE, 2013.
4. Lazar, Timotej, Martin Moina, and Ivan Bratko. "Automatic Extraction of AST Patterns for Debugging Student Programs." International Conference on Artificial Intelligence in Education. Springer, Cham, 2017.
5. Johnson, Amy M., et al. "iSTART-ALL: Confronting Adult Low Literacy with Intelligent Tutoring for Reading Comprehension." International Conference on Artificial Intelligence in Education. Springer, Cham, 2017.
6. Chou, Chih-Yueh, Tak-Wai Chan, and Chi-Jen Lin. "Redefining the learning companion: the past, present, and future of educational agents." Computers & Education 40.3 (2003): 255-269.
7. McKendree, Jean. "Effective feedback content for tutoring complex skills." Human-computer interaction 5.4 (1990): 381-413.
8. Mehrabian, Albert. "Silent messages". Vol. 8. Belmont, CA: Wadsworth, 1971.
9. Marietto, Maria das Graas Bruno, et al. "Artificial intelligence markup language: A brief tutorial." arXiv preprint arXiv:1307.3091 (2013).
10. Pandorabots, `https://www.pandorabots.com`
11. Frisby, Brandi N., and Matthew M. Martin. "Instructor-student and student-student rapport in the classroom." Communication Education 59.2 (2010): 146-164.
12. Gulz, Agneta, Magnus Haake, and Annika Silvervarg. "Extending a teachable agent with a social conversation module-effects on student experiences and learning." International Conference on Artificial Intelligence in Education. Springer, Berlin, Heidelberg, 2011.