Visualization of Program Behaviors: Physical Robots Versus Robot Simulators

Cheng-Chih Wu¹, I-Chih Tseng¹, and Shih-Lung Huang²

¹ National Taiwan Normal University, Information and Computer Education,
  162 Heping East Road, Sec. 1, Taipei, Taiwan
  {chihwu,samuel}@ice.ntnu.edu.tw

² Taipei Municipal Dali High School,
  2, Changshun St., Taipei, Taiwan
  braingrant179@yahoo.com.tw

Abstract. This study compared the effects of using physical robots (LEGO Mindstorms) and robot simulators (LEGO Mindstorms Simulator, LMS) in teaching novice programming concepts. A quasi-experiment design was implemented in this study. Four classes of high school students, totaling 151 students, participated in the study. Two classes of 76 students used the physical robots to learn programming, whereas the other two classes of 75 students used LMS. The students’ post-experiment achievement tests, replies on questionnaires, and focus group interview data were collected and analyzed. The findings of the study were: (1) no significant difference was found on students’ performance between the physical robot group and the simulator group, (2) the physical robot group demonstrated more positive attitudes toward the learning activities, and (3) the physical robot group indicated that they could better imagine the program behaviors.

Keywords: Programming, Robot, Simulator, LEGO Mindstorms.

1 Introduction

In recent years, computer science educators have advocated the use of robots in CS1 and CS2 courses, especially in teaching novices programming concepts [1, 2, 3, 4]. Some [5] further proposed that robotics-oriented projects could be used to motivate and enhance the learning of certain knowledge areas recommended in ACM Computing Curriculum 2001 [6]. The use of robots addresses the problems or difficulties that novice programmers face such as weak logical reasoning and coding before designing; and students find working with robots interesting and fun [3].

Early use of the concept of robots in teaching programming originated from the development of the “mini-language” approach. The idea of the mini-language approach is to design a small and simple language to support the beginning of learning programming. Mini-languages provide insight into programming for the general population and to teach algorithmic thinking [7]. The programming environment of mini-languages is often accompanied with a micro-world, where students learn to
program by studying how to control a robot, which can be a turtle, a ladybug, or other agents, acting in the micro-world. Students can actually “see” how their programs were executed by observing the behaviors of the programmed robot in the micro-world. For example, in Papert’s LOGO [8], students command a turtle robot to draw figures on the screen. In Karel the Robot (http://www.mtsu.edu/~untch/karel/), the robot lives in a micro-world consisting of streets, walls, and beepers. The robot can move along the streets, detect walls, and pick up, move, and put down beepers. Concepts such as procedures and basic control structures can be quickly introduced to students by using the Karel micro-world. The visual representation of the robot’s world offers many benefits, among them are: the animation provides visual feedback on the correctness of an algorithm and students can often see where their program goes wrong simply by watching the animation [2]. Since students may eventually learn to program with general-purpose programming languages, it was proposed to use a subset of a general-purpose language to replace the mini-language approach [7]. In fact, various versions of Karel have been developed based on different programming languages such as C, C++, and Java.

A good way to make the virtual robot more attractive for young students is to use a real world robot [7]. The more practical and concrete the learning situations are, the more learning takes place [9]. The game-like feature of the real robots motivates students to invest time and mental efforts on learning [10]. The benefits of using real robots have been supported by the availability of standardized, low-cost robot construction kits during the last decade. For example, the newly released LEGO Mindstorms NXT kit (http://www.lego.com), with a 32-bit microprocessor (NXT brick), 3 interactive servo motors, 4 sensors (sound, ultrasonic, light, and touch sensors), wheels, USB and Bluetooth support, and 519 construction pieces – everything needed for constructing a robot, costs approximately USD 250. It allows the user to build fully autonomous robots with all computing power located within the machine. Many CS educators have used LEGO Mindstorms to teach students programming concepts and the reported effects are mostly positive [9, 11, 12].

Although there are many claimed benefits for using the physical robots to teach novice programming, little empirical evidence has shown that its effects are indeed better than the robot simulators on the screen. If the effects of both teaching tools are the same, it may not be worth spending money on purchasing the physical robots. This study reports our preliminary attempt to compare the effects of using the LEGO Mindstorms (LM) robot and its simulator – LM Simulator in teaching novice programming concepts.

2 Methodology

A quasi-experiment design was implemented in this study. Four classes with a total of 151 10th grade high school students who enrolled in an Introduction to Computers course participated in the experiment. Two classes of 76 students used the LM robots (LMR group) to learn programming concepts, whereas the other two classes of 75 students used the LM simulator (LMS group). The experiment was conducted over a seven-week period in which students met two hours per week. All the learning activities were directed by the same teacher who was experienced in using the LM tools.