Faculty of Cognitive Sciences and Human Development

SCAFFOLDING THE LEARNING OF PROGRAMMING USING LEGO MINDSTORMS EV3

Tengku Farah Hanum Nor Binti Tengku Mohd Noor

Bachelor of Science with Honours (Cognitive Science) 2020

UNIVERSITI MALAYSIA SARAWAK

Grade: A

Please tick one

Final Year Project Report ☒

Masters ☐

PhD ☐

DECLARATION OF ORIGINAL WORK

This declaration is made on the 9 day of AUGUST year 2020.

Student's Declaration:

I, TENGKU FARAH HANUM NOR BINTI TENGKU MOHD NOOR, 62770, FACULTY OF COGNITIVE SCIENCES AND HUMAN DEVELOPMENT, hereby declare that the work entitled, SCAFFOLDING THE LEARNING OF PROGRAMMING USING LEGO MINDSTORMS EV3 is my original work. I have not copied from any other students' work or from any other sources with the exception where due reference or acknowledgement is made explicitly in the text, nor has any part of the work been written for me by another person.

9 AUGUST 2020

Tengku Farah

TENGKU FARAH HANUM
NOR BINTI TENGKU MOHD
NOOR (62770)

Supervisor's Declaration:

I, DR. FITRI SURAYA BINTI MOHAMED HAPNI JOBLIE , hereby certify that the work entitled, SCAFFOLDING THE LEARNING OF PROGRAMMING USING LEGO MINDSTORMS EV3 was prepared by the aforementioned or above mentioned student, and was submitted to the "FACULTY" as a *partial/full fulfillment for the conferment of BACHELOR OF SCIENCE WITH HONOURS (COGNITIVE SCIENCE), and the aforementioned work, to the best of my knowledge, is the said student's work

Received for examination by: _____ Date: 9 August 2020

PROF MADYA DR
FITRI SURAYA BINTI
MOHAMAD HAPNI JOBLIE

I declare this Project/Thesis is classified as (Please tick (√)):

☐ **CONFIDENTIAL** (Contains confidential information under the Official Secret Act 1972)*****
☐ **RESTRICTED** (Contains restricted information as specified by the organisation where research was done)*****
☒ **OPEN ACCESS**

I declare this Project/Thesis is to be submitted to the Centre for Academic Information Services (CAIS) and uploaded into UNIMAS Institutional Repository (UNIMAS IR) (Please tick (√)):

☒ **YES**
☐ **NO**

Validation of Project/Thesis

I hereby duly affirmed with free consent and willingness declared that this said Project/Thesis shall be placed officially in the Centre for Academic Information Services with the abide interest and rights as follows:

- This Project/Thesis is the sole legal property of Universiti Malaysia Sarawak (UNIMAS).
- The Centre for Academic Information Services has the lawful right to make copies of the Project/Thesis for academic and research purposes only and not for other purposes.
- The Centre for Academic Information Services has the lawful right to digitize the content to be uploaded into Local Content Database.
- The Centre for Academic Information Services has the lawful right to make copies of the Project/Thesis if required for use by other parties for academic purposes or by other Higher Learning Institutes.
- No dispute or any claim shall arise from the student himself / herself neither a third party on this Project/Thesis once it becomes the sole property of UNIMAS.
- This Project/Thesis or any material, data and information related to it shall not be distributed, published or disclosed to any party by the student himself/herself without first obtaining approval from UNIMAS.

Tengku Farah

Student's signature: _____ Supervisor's signature: _____

Date: 9 August 2020 Date: 9 August 2020

Current Address:
Jalan Datuk Muhammad Musa, 94300
Sarawak

Notes: ***** If the Project/Thesis is **CONFIDENTIAL** or **RESTRICTED**, please attach together as annexure a letter from the organisation with the date of restriction indicated, and the reasons for the confidentiality and restriction.

The project entitled ‘Scaffolding Learning of Programming Using Lego Mindstorms EV3 was prepared by Tengku Farah Hanum Nor Binti Tengku Mohd Noor and submitted to the Faculty of Cognitive Sciences and Human Development in partial fulfillment of the requirements for a Bachelor of Science with Honours (Cognitive Science).

Received for examination by:

(FITRI SURAYA BINTI MOHAMAD HAPNI JOBLIE)

Date:

9 August 2020

Grade
A

ACKNOWLEDGEMENTS

First and foremost, thank you Allah, the Almighty, for the blessings and help throughout my research.

To Ayah and Mama, thank you for the love and support throughout my entire life. Thanks for not giving up on me. I'm forever indebted to both of you.

To Dr. Fitri, greatest thanks to you, not only for the guidance you have given me throughout this whole process, but also for the knowledge you have bestowed to me throughout my entire degree life. Big or small, I will carry them to all corners of the world.

To Zulykha and Liyana, both of you are like the Innosuke and Zenitsu to my Tanjiro. I am super grateful that we crossed paths.

Lastly,

To my participants, thank you for believing in me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This study examined the effect of Lego Mindstorms EV3 on learners' problem solving and computational thinking scales, and its impact on their motivation and interest in computer programming. An intervention using Lego Mindstorm EV3 is conducted on 10 first year students of Cognitive Science. Problem Solving Survey (PSS) and Computational Thinking Test was used to measure students' pre and post-test scores. The pre and post-test scores was analyzed using paired sample T-test. To measure the impact of EV3 on the students' motivation and interest, a focus group interview was conducted. It is found that there's a significant increase between the pre and post test for both tests. For problem solving scores, there was a significant difference in the scores for pretest (M = 65.6, SD = 5.87) and posttest (M = 68.0, SD = 6.32); t(9) = -3.88, p = .004. For computational thinking scores, there was a significant difference in the scores for pretest (M = 66.2, SD = 7.63) and posttest (M = 70.0, SD = 9.15); t(9) = -4.13, p = .003. EV3 give positive impact on the learning experience. Before the intervention, majority of the participants had negative mindsets on programming, in general (Refer Table). These mindsets affected their motivation and interest in learning, thus, making it hard for them to learn. However, we can see from both observations during the sessions and the responses from post-intervention interview, that the mindset and behavior of the students had changed positively (Refer Table). The participants enjoyed the learning process. Thus, we can say that motivation factor should not be undervalued in educational process. Motivated learners often achieve more than unmotivated learners (Chetty, 2015).

Keywords: Novice programmer, robotics, problem solving skill, computational thinking skill, constructivism

CHAPTER ONE

INTRODUCTION

This research is done to study and analyse the effect of LEGO Mindstorms EV3 on learners' problem-solving & computational thinking skills and to examine its impact on their motivation & interest in computer programming. LEGO Mindstorms EV3 is a robot set use in programming education. EV3 consists of building components, programmable brick, active sensors, and motors. There is software for which both Graphical User Interface (GUI) and command-line interfaces are available. The robots, together with their interfaces provide an opportunity for educators to transform classrooms into rich laboratory or software studios, where learners can experience learner-centered learning, collaborative learning, and peer-to-peer programming experimentation (Yamazaki, Sakamoto, Honda, Washizaki, & Fukazawa, 2015). The environment provides an opportunity for learners to 'put their programming skills to the test' as what they program comes to life through the LEGO Mindstorms robot. They can visually understand 'what works', 'what does not work' and 'why'. In other words, LEGO Mindstorms is an interactive robot that produces output according to the user or learner's input. EV3 is different from Scratch (block-based visual programming language) in a sense where it uses action instead of explanation and affords an interactive, decision making context.

This study will be conducted with a focus group, which are the selected first-year students of the Cognitive Science program. It is quasi-experimental research as the students' will be assessed with two tests (Problem Solving & Computational Thinking Test) and interview on motivation and interest, before and after the intervention of EV3.

Background of Study

The mean worldwide for failure rates in introductory programming courses have been suggested to be as high as one-third of students failing the course (Bennedsen & Caspersen, 2007). A recent study showed that despite advances in pedagogy, the worldwide failure rates have not improved over time (Watson & Li, 2014). Despite decades of research, internal factors based upon traditional learning theories have also failed to explain the CS1 (First Introductory Programming Course) failure rate phenomenon, and no factor to date has been shown to influence programming performance across a range of different teaching contexts (Watson, Li, & Godwin, 2014). More recently, researchers have explored the connections between desirable aspects of programming behavior and performance (Vihavainen, Luukkainen, & Kurhila, 2013). But despite yielded promising results, there is still no overall understanding as to why many students can program, whilst others endlessly struggle.

Many theories have been put forward as to why learning to program is a difficult task. Some are attributed to the nature of programming itself. Programming is not a single skill, but rather a complex cognitive activity, where a student must simultaneously build and apply several higher-order cognitive skills to solve a particular problem (Robins, Rountree, & Rountree, 2003). Other reasons are attributed to the aspects of the students. Students may lack motivation, and they may be unable to create a mental model of how programs relate to the underlying system (Tew, Fowler, & Guzdial, 2005) or create a clear model of program flow (Milne & Rowe, 2002).

Further reasons are associated with the teaching methodology used. Whilst many students fail programming courses, instructors face the additional challenge of adjusting their expectations to the students' level of ability (Utting et al., 2013). These challenges have been acknowledged, and decades of research effort has been put into creating and applying teaching interventions that facilitate students' learning (Robins et al., 2003; Pears et al., 200). These interventions can

include moving from a traditional lecture and lab-based approach to using pair programming, game-based learning, or extreme apprenticeship. However, to date, only little study has attempted to quantitatively compare the impact that these approaches can have on improving the pass rates of failing programming courses. Without any quantitative evidence on the relative strengths of different approaches, the research community will continue to lack a clear consensus of precisely which methodologies provide the most effective means of teaching programming and saving failing programming students.

Problem Statement

As a novice programmer who never been exposed to programming subjects before, it is very tough for them to learn introductory programming courses. Coming from a non-technical background, it is expected that the novice students didn't develop enough programming-related skills such as problem solving and computational skills. Given that such students are new to the discipline, their cognitive load increases exponentially, often exceeding their critical threshold level of cognitive capacity (Mason, Cooper, Simon, & Wilks, 2015). This situation would eventually make them demotivated and decreased their interest in learning. The aftermath of this whole thing is the student might fail their course or drop out of the program. Thus, the problem here is on how to help the students to reinforce their problem solving and computational thinking skills and increase their motivation towards programming course.

Although there is a lot of methods and tools used in teaching programming, the problem in learning introductory programming for novice students still exist. Researchers still yet to find the best method to curb the problem. This is where robotics comes into the equation. Educational robots help students in a variety of ways, including providing them with a feedback-oriented learning environment (Sullivan & Bers, 2016), helping the understanding of abstract concepts (Eguchi, 2013), giving them opportunities to work and explore solutions to real-world problems (Miller, Nourbakhsh, & Siegwart, 2008), and giving them a collaborative working environment (Eguchi & Uribe, 2017). LEGO Mindstorms EV3 is a good example of educational robotics. Unfortunately, although there is an abundance of studies on learning to program using robotics, especially using LEGO EV3, most of the studies are carried out on K12 students and only minimally on students in higher education. Thus, the impact of robotics or EV3 is not very clear.

Objectives of the Study

Main Objective. The main objective of this study is to determine the effect of LEGO Mindstorms EV3 on learners' problem-solving & computational thinking skills and to explore its impact on their motivation & interest in computer programming.

Specific Objectives. The specific objectives of this study are:

1. To determine the learning difficulties of novice programmers in an introductory programming course.
2. To investigate learners' perception towards learning introductory programming.
3. To determine the effect of LEGO Mindstorms EV3 on learners' problem solving and computational thinking skills.
4. To explore the impact of LEGO Mindstorms EV3 on the learners' motivation and interest in computer programming.

Research Questions

Based on the main objectives, the following questions were addressed:

1. Does LEGO Mindstorms EV3 have an effect on the learners' problem solving and computational thinking skills?
2. How does the LEGO Mindstorms EV3 impact the learners' motivation and interest in computer programming?

Definition of Terms

Computational Thinking. The ability of a learner to develop problem-solving techniques and strategies that assist in the design and use of algorithms and models. The three stages are problem formulation, solution expression, and solution execution & evaluation (Repenning, Basawapatna, & Escherle, 2016)

Problem Solving. Problem-solving is defined as understanding the context of a problem, identifying key information, and making a plan to solve it (Gardiner & Zeitz, 2000)

Abstraction Skills. Wing (2010) defines abstraction as "the most important and high-level thought process in computational thinking (...) used in defining patterns, generalising from specific instances, and parameterisation (...) to capture essential properties common to a set of objects"

Pedagogical Approach. Pedagogical approaches relate to the manner in which teaching and learning take place in order to facilitate desired learning outcomes (Pears et al., 2007). There are several pedagogical approaches to teaching and learning (Boyer, Langevin, & Gaspar, 2008; Pears et al., 2007). Fosnot & Perry (1996) defined Constructivism is a learning theory found in psychology that explains how people might acquire knowledge and learn.

CHAPTER 2

LITERATURE REVIEW

Problems Faced by Novice Students

Generally, learning to program is considered hard. Thus, programming courses often have high dropout rates. Experts said that it takes about 10 years for a novice programmer to become a pro. A student who never took a programming course before is considered as a novice and as a novice, there are a lot of difficulties faced by them in learning introductory programming courses. A great deal of research has been carried out to recognise the challenges faced by novice students in higher education and among the reasons are difficulties in problem formulation, the students' skills and background knowledge, and behavior.

The first category is related to one of the stages of computational thinking. Falkner, Vivian, & Falkner (2015) defined computational thinking as the ability of a learner to develop problem-solving techniques and strategies that assist in the design and use of algorithms and models. The three stages are problem formulation, solution expression, and solution execution & evaluation (Repenning, Basawapatna, & Escherle, 2016), but most of the literature focused on the first stage, problem formulation as the main problem for novice students.

Problem Formulation. It has been reported that problem-solving is a challenge faced by students (Pears et al., 2007; Eltegani & Butgereit, 2016; Seppäl, Ihantola, Isohanni, Sorva, & Vihavainen, 2015). Mirolo (2010) claimed that the learners face more difficulties to establish relations between different problem instances. Thus, many of these students end up dropping the course due to not being able to solve problems and therefore feeling inadequate (Vihavainen, Paksula, & Luukkainen, 2011).

The abstract nature of programming makes it more difficult for novice students to solve a

problem. As stated by Giraffa, Moraes, & Uden (2014):

> Students in learning programming need to imagine and comprehend many abstract terms that do not have equivalents in real life: how does a variable, a data type, or a memory address relate to a real-life object? These concepts are difficult to grasp. Consequently, many students struggle to comprehend even the most basic programming concepts.

Other than their inability to formulate a problem, student's skill and background knowledge are also one of the challenges for them.

Skills and Background Knowledge. There are two skills that are very important for a novice student to learn programming, which is programming related skills and general education skills. These two skills play an important role in determining their success in learning programming. Programming related skills are problem-solving, mathematical ability, abstraction, and previous knowledge in programming, while general education educational skills are basic knowledge in English, critical thinking and discussion skills, creativity and time management (Medeiros, Ramalho, & Falcao, 2019; Piteira & Costa, 2013; Robins et al., 2003).

One of the most cited programming-related skills is problem-solving skills. Problem-solving is defined as understanding the context of a problem, identifying critical information, and making a plan to solve it (Gardiner & Zeitz, 2000). Previous research has established that problem solving is considered an essential and a prerequisite for learning how to program. According to (De Lira Tavares, De Menezes, & De Nevado, 2012), computer programming depend upon the use of problem-solving techniques and the lack of ability of these techniques makes it difficult to learn to program. Besides, even the primary goal of introductory programming courses is to familiarise the students with the art of problem-solving (Ateeq, Habib, Umer, & Ul Rehman, 2014).

Mathematical ability is equally essential as problem-solving skills, and it is closely related to each other. To date, several studies in teaching and learning introductory programming have argued that insufficient mathematical ability is one of the reasons for the difficulties encountered by novice students. According to Gomes & Mendes (2010):

> When talking with programming teachers, most of them claim that students don't know how to program because they don't know how to solve problems and don't have enough mathematical background.

However, the connection between programming and mathematics is not necessarily a cause-effect one. Ambrósio, Costa, Almeida, Franco, & Macedo (2011) stated that:

> Some teachers believe that mathematical ability is important for programming. However, they believe that the positive correlation between them is due to more basic cognitive functions that would be common to both domains. Thus, when developing a "mathematical logic", the student acquires structures or cognitive abilities that facilitate or promote learning to program.

Other than the above skills, previous knowledge in programming also gives an impact on the success of novice students. As stated by Porter & Zingaro (2014), prior programming has a positive impact on Computer Science studies, and students who studied programming in the past tend to do well compared to those with no programming experience. Furthermore, the impact of the experience will affect students' expectations, work habits, attitude and confidence, and perceptions of self and peers.

It has been conclusively shown that a student needs to have commendable abstraction skills when learning to program. Wing (2010) defines abstraction as:

> The most important and high-level thought process in computational thinking (...) used in defining patterns, generalising from specific instances, and parameterisation (...) to capture essential properties common to a set of objects.

As reported by Ambrósio et al., (2011), it is expected that students will use abstraction to pass from the concrete world to a more semantic and symbolic reality, formulating or representing a problem in a more abstract manner, less attached to the details and singular concrete elements. When analysing computational problems, a programmer needs to apply abstraction. Hence, when a student faces difficulty while attempting to do abstraction, they are incapable of moving from reality to a more generic language such as the notation used in programming.

Besides programming-related skills, general educational skills such as critical thinking, discussion skills, basic English knowledge can also be predictors to the students' success in the programming course. Both critical thinking and discussion skills affect the analysis of problems and limited knowledge in English would add as a barrier as it is used in the syntax of programming languages. Horton & Craig (2015), conducted a study to find out how characteristics such as basic knowledge in English influence drop, fail, or passes and found out that fluent English-speaking students were more likely to be successful than non-fluent ones.

Behavior. Social and emotional aspects can have an impact on students' learning, such as engagement, motivation, and confidence. Confidence stated here includes the students' perception of programming being difficult. The most discussed issue here is on how the relationship between students' engagement and motivation affect the learning outcomes.

The problem with the novice students is they perceived programming as a complex disciplined and directly contribute to students' demotivation (Gomes & Mendes, 2015). In the light of Brito & De Sá-Soares (2014):

An unmotivated student will hardly succeed. This is aggravated by the fact that programming courses usually gain a reputation that passes from student to student of being difficult that so many of them start already feeling defeated.

Coupled with the previously discussed learning challenges, it would add more difficulties for them to learn to program.

From all the problems discussed above, there are two main problems identified that need to be tackled, which is the students' computational and problem-solving skills and their low motivation and interest in programming.

Pedagogical and Instructional Approaches

Pedagogical approaches relate to how teaching and learning take place to facilitate desired learning outcomes (Pears et al., 2007). There are several pedagogical approaches to teaching and learning (Boyer, Langevin, & Gaspar, 2008; Pears et al., 2007).

Teacher Centric Approach. A teacher-centric approach consists of activities, such as lecturing, questioning, and demonstration. The lecturer is the expert who transfers their knowledge across to learners (Xiaohui & Zhou, 2006). This approach is used extensively to teach not only computer programming courses but also other disciplines of study. Although the teacher-centric approach is the most popular, these traditional pedagogical approaches do not lend themselves towards a positive experience (Lister, 2011). As mentioned by Eltegani & Butgereit (2016), some of the factors that result in un-engagement among students are: the present traditional style of education does not appeal to everyone. Besides, some educators are not sensitive to students' responses to educational methods in the class.

Learner-centered Approach. Other approaches are unique, for instance, the learner-centered approach. Contrary to teacher-centered traditional learning methods where students are

passive listeners, this approach argues that students should be in a position to be active in learning. It defends that students do not take the knowledge reaching him as it is and that pre-knowledge of individual, his features, and also learning environment are of great importance (Nakiboglu, 2001). Besides, it involves philosophies that have been around for many decades. To name a few, these include social constructivism, problem-based learning, and collaborative learning. As explained by Fosnot & Perry (1996):

> Constructivism is a learning theory found in psychology that explains how people might acquire knowledge and learn.

The theory suggests that humans construct knowledge and meaning from their experiences and human beings learn better when they are involved in planning and building artifacts (Wulf, 2005). There is a past study on the effects of 'Social Constructivist Learning Environment' (SCLE) on the 5th-grade primary school learners and the analyses showed that SCLE design is effective on learning of learners and on the provision of learners with new information through group work and multimedia. At the end of the study, SCLE design was concluded to have a positive effect on learning outcomes (Akyol & Fer, 2010).

From the literature above, it is known that the student-centered learning approach that uses the constructivist style is way more effective than the traditional teacher-centric approach.

Introduction of Graphical User Interface. Programming is also taught through the introduction of graphical user interface (Guillory) tools, such as Scratch, Greenfoot or Alice provide a simulated computer programming environment that is user-friendly. It provides ease-of-use when trying to develop computer programs (Maloney, Resnick, Rusk, Silverman, & Eastmond, 2010).

Although there is a lot of methods and tools used in teaching programming, the problem

in learning introductory programming for novice students still exist. Researchers still yet to find the best method to curb the problem.

The Use of Robotics in Learning Programming.

In related literature, several studies are demonstrating that programming with robots is a more entertaining and effective method when compared to traditional programming education (Kurebayashi, Kamada, & Kanemune, 2006; Liu, Newsom, Schunn, & Shoop, 2013; Major, Kyriacou, & Brereton, 2012). As claimed by Patterson (2011), in a total of 19 studies, 14 of them had a positive influence on programming education. It is reported that education on programming with robots is engaging and motivating (Liang, Fleming, Man, & Tillo, 2013). The problem is there is an abundance of studies carried out on K12 students, but little on the students in higher education.

LEGO Mindstorms EV3

In a related market, there are several tools for robot use in programming education. Among these tools, the most popular one is Mindstorms developed by the LEGO company. In the present study, the LEGO Mindstorms EV3 basic education set was used. In LEGO Mindstorms EV3 robots, there is one programmable brick. On this brick, four ports are found to connect the engines and sensors. Besides, in the basic education set, there are two big and one small engine, a color sensor, a touch sensor, a gyro sensor, an infrared sensor, and various plastic parts to produce simple robots.

LEGO Mindstorms robots have become a popular tool to teach and learn introductory computer programming concepts (Lawhead et al., 2002). The emphasis is on the word 'tool', where robots create a rich environment that provides a platform for novices as well as experienced teachers to implement a laboratory experience for students to learn programming skills in an

interesting, unique, and challenging manner. LEGO Mindstorms change the premise that computation is a calculation to the notion of computation is interaction. It provides a natural way to explore such a concept.

The robots, together with their interfaces provide an opportunity for educators to transform classrooms into rich laboratory or software studios, where learners can experience learner-centered learning, collaborative learning and peer-to-peer programming experimentation (Yamazaki et al., 2015). The environment provides an opportunity for learners to 'put their programming skills to the test' as what they program comes to life through the LEGO Mindstorms robot. They can visually understand 'what works', 'what does not work' and 'why'. In other words, LEGO Mindstorms is an interactive robot that produces output according to the user or learner's input.

Programming concepts are abstract and traditional pedagogical approaches to teaching programming do not help to solve the problem. However, LEGO Mindstorms can be used to help learners to learn in a more interesting and effective environment.