CONTEXTUAL FACTORS INFLUENCING ENGINEERING STUDENTS' MOTIVATION IN LEARNING COMPUTER PROGRAMMING

NOR AZLINDA BINTI AZMI

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Master of Philosophy (Engineering Education)

Faculty of Engineering
Universiti Teknologi Malaysia

SEPTEMBER 2019

ACKNOWLEDGEMENT

Alhamdulillah. Praise be to Allah.

I would like to express my heartiest gratitude to my supervisors, Prof. Dr. Khairiyah Mohd. Yusof and Assoc. Prof. Dr. Fatin Aliah Phang for their guidance, advice, and help throughout the whole journey of this study. For their patience, time and support, they will always be inspirations for me to strive harder in my life. To all the lecturers of the courses, workshops, colloquiums that I attended, due to all of your knowledge sharing and advice, I am able to learn new knowledge of this new field I venture. To all my colleagues and friends, who are always there to help both in academic and moral support, I am truly grateful.

Last but not least, the most special gratitude, to my beloved parents Azmi Man and Aziah Ibrahim as well as my brothers Aziq and Alif, for trusting me in pursuing my passion regardless of the many breakdowns I faced. Thank you for all the prayers and supports.

ABSTRACT

Having computer programming skills are crucial for engineers. Thus, computer programming has been made a compulsory course for all engineering students. However, computer programming has the reputation of being demotivating and is viewed as not interesting by most engineering students. Consequently, this hampered their learning. Therefore, there is a need to find ways to motivate the engineering students to learn computer programming through appropriate pedagogical approaches that meet students' motivational needs. This study aims to explore the influence of contextual factors of the learning environment towards engineering students' motivation in learning computer programming. The studied elements were the contextual factors relating to the learning environment, the students' motivational beliefs and the students' behaviours. This naturalistic inquiry research was adopted to gather the data through non-participant observations and semi-structured interviews. The observations were conducted over a period of 14-weeks in three classrooms taught by different lecturers of the introductory computer programming course for first-year engineering students. Eighty-seven students participated in a survey using the Motivated Strategy Learning Questionnaire (MSLQ) which later was used to identify the 10 selected interview respondents. The observation and interview data were recorded and transcribed for analysis using thematic analysis technique. The analysis discovered eight contextual factors and motivational beliefs that influence them to study computer programming and they are "Students' Characteristics", "Teaching Method", "Lecturer's Characteristics", "Assessment", "Task Driven Motivation", "Peer Driven Motivation", "Involvement" and "Persistent Behaviour". The findings conclude that there are two significant pedagogical approaches that can help to increase engineering students' motivation in learning computer programming namely collaborative learning environment and formative assessment.

ABSTRAK

Mempunyai kemahiran pengaturcaraan komputer adalah penting kepada jurutera. Oleh itu, pengaturcaraan komputer telah dijadikan kursus wajib untuk semua pelajar kejuruteraan. Walaubagaimanapun, pengaturcaraan komputer mempunyai reputasi tidak memotivasikan dan dilihat sebagai tidak menarik dalam kalangan pelajar kejuruteraan. Dengan yang demikian, ini boleh menjejaskan pembelajaran mereka. Oleh itu, terdapat keperluan untuk mendapatkan cara untuk memotivasikan pelajar kejuruteraan dalam pembelajaran pengaturcaraan komputer melalui pendekatan pedagogi yang sesuai bagi memenuhi keperluan motivasi pelajar. Kajian ini bertujuan untuk mengkaji pengaruh faktor kontekstual dalam persekitaran pembelajaran terhadap motivasi pelajar semasa pembelajaran pengaturcaraan komputer. Unsur-unsur yang dikaji adalah faktor kontekstual dalam persekitaran pembelajaran, kepercayaan motivasi pelajar dan tingkah laku pelajar. Kajian inkuiri sejadi telah digunakan untuk mengumpul data melalui pencerapan tak berpandu dan temu bual separa struktur. Pencerapan dijalankan sepanjang tempoh 14 minggu pelajaran di tiga bilik darjah yang diajar oleh pensyarah yang berbeza bagi kursus pengenalan kepada pengaturcaraan komputer untuk pelajar kejuruteraan tahun pertama. Lapan puluh tujuh pelajar mengambil bahagian dalam kaji selidik yang menggunakan Kaji Selidik Strategi Pembelajaran Bermotivasi *(MSLQ)* yang kemudiannya digunakan untuk mengenal pasti 10 orang responden untuk temu bual. Data pencerapan dan temu bual telah direkodkan dan ditranskripsi untuk dianalisis menggunakan teknik analisis tematik. Analisis ini menemui lapan faktor kontekstual dan kepercayaan motivasi pelajar yang mempengaruhi pelajar dalam mempelajari pengaturcaraan komputer, iaitu "Ciri-ciri Pelajar", "Kaedah Pengajaran", "Ciri-ciri Pensyarah", "Penilaian", "Motivasi yang Didorong oleh Tugas", "Motivasi yang Didorong oleh Rakan Sebaya", "Penglibatan" dan "Perilaku yang Berterusan". Dapatan kajian menunjukkan, terdapat dua pendekatan pedagogi yang penting dalam membantu meningkatkan motivasi pelajar terhadap kursus pengaturcaraan komputer, iaitu persekitaran pembelajaran kolaboratif dan penilaian formatif.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Introduction

As we are approaching the year 2020, Malaysia is expecting a continuing change in the engineering education paradigm (Nor et al., 2008). This has also become a need due to the demand of the industries to employ highly competent engineers that can work to enhance productivity. Due to this competition in the industry, the engineers are expected to be quick in providing novel solutions (Mohd-Yusof et al., 2015). This is relatable to the growing demand for a technology-savvy workforce in Industry 4.0 which has stirred a number of debates around how to best equip engineering graduates with computational knowledge and computer programming skills.

Furthermore, to remain relevant and sustain high productivity, automation and computing technologies are highly adapted into the engineering education curriculum. In the current information society, new computational technologies such as cloud computing, artificial intelligence, development of advanced materials, advanced analytics, big data and robotics, have largely impacted the job market, industrial production, social relationships, health, security, commerce, research and development and education (dos Santos et al., 2018). Also, to decrease the cost of raw materials and energy, the need for efficient and optimized processes is required in production lines in which low production cost is also one of the demands in the current industrial revolution (Mohd-Yusof et al., 2015). Therefore, engineering students are anticipated to be well-equipped with computational and programming skills. They have to be continually adaptive and increase their computational knowledge and computer programming skills for today's market competition.

It is as well noted that, with the existence of computer programming, it enables engineers' everyday life becoming more efficient. According to the National Academic Sciences report reviewed by Urban-Lurain & Weinshank (2001), the term "computer literacy" no longer means just to make engineering students understand the concepts but also to incorporate extensive training on computer applications to face the upcoming challenges in information technology. The report is particularly addressing the importance of teaching computer programming skills to produce Fluent with Information Technology (FIT) students. This has been similarly emphasized in the Engineering Accreditation Council (EAC) requirements for accreditation. Parts of the accreditation requirements (EAC, 2015) are to equip engineering students with the ability to:

1. Apply knowledge of mathematics, science, and engineering.
2. Be able to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability.
3. Be able to identify, formulate, and solve engineering problems.
4. Be able to use the techniques, skills, and modern engineering tools necessary for engineering practice.

Knowing how to program is essential to engineers (Chandramouli, Zahraee, & Winer, 2014; Ring, Giordan, & Ransbottom, 2007). Engineers develop programs by coding, through the use of programming language that will help them to perform tasks (Forte & Guzdial, 2005). Good programming knowledge and skills will also enable them to tackle engineering problems easily and be better in the discipline (Bradley, 2011). Sharing his thoughts to address practicing 21st-century scientist and engineers, Philip Guo's (2013) value proposition to them is:

"If you're a scientist or engineer, programming can enable you to work 10 to 100 times faster and to come up with more creative solutions than your colleagues who don't know how to program."

In addition, computer programming is also considered as a powerful tool in engineering due to its ability to develop computational, critical thinking and problem-solving skills (diSessa and Abelson, 1986; Ring et al., 2007; Chandramouli et al., 2014). These prove that computer programming is an important skill in every engineering discipline to produce better learners cognitively (Gomes and Mendes, 2007b; Jerez et al., 2012; Chandramouli et al., 2014). Hence, it should be extensively improved in terms of teaching and learning.

To achieve the desired objectives, computer programming courses (or sometimes called as subjects or classes) are common and mandatory to most engineering undergraduates. It is introduced as early as the first semester to engineering undergraduates such as "Introduction to Engineering Computation" course (Attaway, 2010). By introducing computer programming to engineering undergraduates, it is anticipated that they would be able to get ideas on solving a particular type of problem by developing programs in a given situation. This would facilitate them to acquire not only the knowledge of the concepts of programming but the ability to demonstrate problem-solving skills using programming in their projects and future career. Parts of the course is also to provide the students with the ability to do the programming by themselves (hands-on programming skills) (Dabney & Ghorbel, 2005). The course also equips the students with the basic programming languages, instructions and coding methods (Dagiene et al., 2014; Sun & Sun, 2011).

However, there is a major concern with learning computer programming among engineering students which is poor understanding. Poor understanding of basic concepts will make advanced subjects more difficult. For example, courses such as numerical methods, control system, process control, and other related courses will need the students, at least, to understand how to execute algorithms or coding using concepts of programming.

Meanwhile, if the introductory programming courses are generally labeled as overly difficult and hard to pursue, this will discourage the students to perform tasks or other courses that require programming as the basic knowledge. From a pedagogical point of view, it is an undesired situation for both the students and the lecturers, to learn knowledge that is believed to be a life-long skill, but at the end of the course, is not even motivated to explore and pursue such skill in the future.

1.2 Problem Background

According to Pears et al. (2007), computer programming is challenging for every novice regardless of their age. Accordingly, in addressing the challenges faced by novice programmers, there are still many unanswered questions about how computer programming should be taught to non-computer science majors including to engineering students (Chilana et al., 2015). There are many conceptual challenges faced by novice programming learners i.e. first-year engineering students in and one example is the misconception about the fundamental of computing mechanism (Ben-Ari, 1998). The syntax of computer languages is also one of the factors affecting their learning process (Mladenović, Krpan, & Mladenović, 2016). Another example is their abstraction ability which is a complex cognitive development of a learner which is considered difficult to achieve (Bennedsen & Caspersen, 2008).

The negative perceptions of engineering students towards programming seem to persist even though the generation these days are used to using computers daily (Askar & Davenport, 2002). It leads to recognizing that many engineering students claim to "dislike computer programming" and not able to do it. According to studies by Gomes and Mendes (2007a, 2007b), Kinnunen and Malmi (2008), Gomes et al. (2012) and Chandramouli et al. (2014), engineering students also perceive computer programming as a difficult subject. It is also often perceived as an activity for a smaller part of the population (Mladenović et al., 2016), could be for the "computer geeks".

Meanwhile, many students, including engineering students have issues with motivation to learn to program (Gomes and Mendes, 2007a, 2007b; Ring et al., 2007; Gomes et al., 2012; Jerez et al., 2012; Serrano-Cámara et al., 2014). These include their self-efficacy, test anxiety, intrinsic and extrinsic motivation and task value. Since motivation can be considered the stimulus that causes learners to take action or engage in positive behaviours, the behaviours can be observed by their goals, effort, readiness, persistence, performance and high cognitive processing (Pintrich, 1994). For most cases in programming courses for engineering students, less motivated students would resort to a trial-and-error approach rather than learning the real problem-solving skills (Allan & Kolesar, 1996). They would also tend to give up and avoid choosing computer programming as part of their future projects (e.g. final year project) or careers (Jenkins, 2002). Moreover, they are not ready to learn to program in the difficult period of their life (Gomes & Mendes, 2007b), for instance, during the early stage of their undergraduate study which is the adaptation period from the school to their university life. They are also were much more motivated on courses which they claim have greater relevance to engineering curriculum (Jerez et al., 2012). Another study has also reported that most engineering students who learn to program are lack of intrinsic motivation (Ring et al., 2007). They take programming as irrelevant to their study and is not beneficial to them in any means (Jerez et al., 2012).

From the issues mentioned, based on studies, motivation is influenced by the teaching and learning environment (Bandura, 1986; Pintrich, 1994; Svinicki, 2004; Chou and Hsiao, 2011; Schunk, Meece and Pintrich, 2014). In a view of this, teaching and learning in programming courses for engineering students is no longer a process of teaching knowledge; it is much more of a process of motivating. Rather, it is also the engineering lecturers' role to create a motivating learning environment. Meanwhile, learning environment, defined by Garvin-Doxas and Baker (2004), is characterized by the communication and interaction practices that leave the perceptions of the learners who decide to stay to learn the specific domain or major. In other words, the learning environment decides whether the learners will stay motivated and pursue the learning or choose to drop out.

Most typical computer programming courses in engineering programmes have the reputation of demotivating and not interesting. Learning and teaching computer programming for engineering also often remains unchanged although students and their environments are significantly different (Mladenović et al., 2016). For that reason, it is essential for engineering lecturers to embed changes in the learning environment not only by the matter of improving the teaching methods but also sustaining the students' motivation. Recent works dealt with the problem of engineering student motivation in programming such as Gomes et al. (2012), Jerez et al. (2012), Mendes et al. (2012) Serrano-Cámara et al. (2014) and Mladenović et al. (2016) have agreed that the lecturers should create an environment which can motivate the students. To attain such an environment, the lecturers need to understand the elements that motivate the students in any learning environment.

This could, for example, be attained by considering the students' need and ability as a part to develop interest and sustain their motivation towards using programming skills in engineering. One of the described learning environment problems that are significant to the issue of learning computer programming is the instructor i.e. the lecturer (Gomes and Mendes, 2007b). Meanwhile, in instruction, the lecturers' approaches were seen as not reassuring to all students' behaviours (Gomes & Mendes, 2007b). Lecturers focus more on teaching the synthetic details of programming language, instead of teaching engineering problem solving through programming. For content issues, engineering students also cannot relate the applicability of programming constructs in engineering (e.g., control structures, variables) due to the vague explanation in most computer programming instructions (Bowen, 2004; Jenkins, 2002; Urban-Lurain & Weinshank, 2001).

1.3 Problem Statement

Having programming skills are crucial to engineers thus, programming has been made a compulsory course for all engineering students to learn; but computer programming is demotivating and not interesting, and as a consequence engineering students are not learning, i.e. they are not acquiring the programming skills that they need. So there is a need to enhance motivation to make engineering students learn to program through appropriate pedagogical approaches that meet students' motivational needs. However, before appropriate pedagogical approaches can be identified and designed, there is a need to understand the engineering students' motivation to learn and how external (contextual factors in the learning environment) and personal factors (motivational beliefs) interact to result in motivation/demotivation that is observed in learning behaviours. Identifying and understanding of the relevant contextual factors and engineering students' motivational beliefs as well as the behaviour in learning is important in informing appropriate computer programming courses for engineering students pedagogy. Previous studies have only focused on the content of programming subjects whereas limited consideration is observed on the contextual factors of the learning environment (in this manner, is the motivation to learn). This has created a gap in studies which is important to be addressed. Therefore, the aim of the study is to explore the influence of the contextual factors of the learning environment towards the engineering students' motivation in learning programming.

1.4 Research Objectives

The objective of the study is:

To explore the influence of the contextual factors of the learning environment towards engineering students' motivational beliefs and behaviours in learning computer programming.

1.5 Research Questions

Hence, the research questions are:

1. What are the contextual factors of the learning environment that can influence engineering students' motivational beliefs towards learning computer programming?
2. What are engineering students' motivational beliefs in learning computer programming?
3. What are the engineering students' motivational behaviour towards the contextual factors and their own motivational beliefs in learning computer programming?

1.6 Significance of the Study

The findings of this study will significantly benefit the engineering faculty members who teach computer programming courses in addressing issues in the teaching and learning. This study will be an example for the faculty members to initiate research in their own classrooms and teaching area. For computer programming lecturers in engineering faculty, the findings of this study will help them to get some pre-conceptions about their students' motivation factors in learning. The relationship of the three elements studied in this study (contextual factors, motivational beliefs and motivational behaviour) will help the engineering lecturers' to understand engineering students' motivation as a whole. Thus, this can help for further investigations using better instruments, in-depth questions and in depth actions. The recommendations provided by this study is also a benefit for the whole engineering faculty to design effective learning environments for engineering students to learn computer programming.

1.7 Theoretical framework

This study is underpinned by social cognitive theory by Albert Bandura (Bandura (1986). In general definition, social cognitive theory holds that a learner's knowledge acquisition is done by observation through social interactions, experiences, and external influences (Bandura, 1986; 2001). It postulates a concept of guided instruction and "modeling" where the learners' behaviour on the environment are gained from symbolic models portrayed through social verbal or visual means.

Meanwhile, in the perspective of cognitive science, there are multiple views on how students learn to program. According to Thuné and Eckerdal (2009), Pea and Kurland (1984) and Jenkins (2002), learning programming is different than any learning other types of knowledge. It is about learning knowledge, skills, language, and tools. Hence, due to the assumption that computer programming is skill knowledge, not a content driven knowledge, learning is mostly done through observation (Caspersen and Bennedsen, 2007). However, in many engineering undergraduate courses, part of them are the learning of problem solving skills such as solving a particular types of problem or using tools to solve problems. In the other words, the students are anticipated to satisfy problem solving skill and using the programming knowledge, skills and tools in a given problems for applications (Kapuno, 2010). The students are also introduced to problems which require them to solve them according to certain steps or instruction. Therefore, Svinicki (2010) has categorized them as intellectual skills. This type of information transfer process suits the perspective of social cognitive theory (Bandura, 1989; Bandura, 1986; Bandura, 2001; Svinicki, 2010) where learning takes place when the students witness and observe behaviours conducted by others, and then reproduces those actions.

The feedback received by the computer is considered as a powerful reinforcement in the learning such that students are working on practical tasks individually and collaboratively. Working with visual models and output also encourage the students' motivation in solving the given problem. A similar conclusion is reached through this learning concept which agrees to the principles of social cognitive theory saying that learning is best done through social observation and modelling (Bandura, 1986). Therefore, the suitable learning theory to illustrate the learning in computer programming courses in engineering is social cognitive theory.

Social-cognitive theory is based on several key assumptions. Part of the theory lies in the principles of:

a. Reciprocal interaction between personal, behavioural and environmental factors
b. The relation of learning and motivation

In social cognitive model of reciprocal interaction, the learning process is controlled by internal and external factors. It is derived by a model of reciprocal interaction between behaviour, personal factors (cognitive and others), and the environment (Bandura, 1986, 2001).

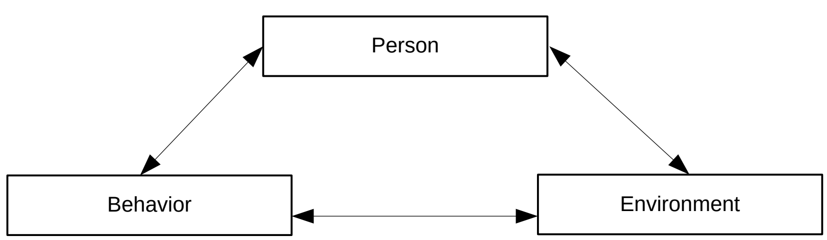

Figure 1.1: Reciprocal Interaction Model of Personal, Behaviour and Environment (Bandura, 1989; 2001)

In social cognitive theory's explanation on motivation, the concept of self-efficacy plays the central part of the learners' motivation. Self-efficacy influences students' achievement behaviours such as persistence and efforts. In return, the students 'positive behaviours affect their self-efficacy when they believe they can perform well. While the third interaction which is the person-environment interaction postulates that students with learning difficulties are creating reactions on the actions of the community. In return, the instructor which is also considered as part of the environment will influence these students' self-efficacy by encouragement. As a sum, these three reciprocal of interaction are displayed in most instructional environment.

1.8 Conceptual framework

The Reciprocal Interaction Model of Personal, Behaviour and Environment by Bandura (1989) in Figure 1.1 (in theoretical framework) is adapted to develop the conceptual framework of this study. Parts of Bandura's work highlight on the importance of addressing students' motivation as part of a reciprocal interaction between the person, environment and behaviour. In line with this, the three components of the model of academic motivation by Pintrich et al. (1994) model are generally representing the reciprocal interaction model in social cognitive theory which consists of the environment, person and behaviour, in a more explicit meaning. The contextual factor represents the environment, the motivational beliefs represents the person and the motivated behaviour represents the behaviour. This model is also aligned with the general cognitive perspective which is individuals are assumed to be active processors of information and to construct their own meaning and perceptions of the context.

It is adapted to show the relationship of a simple integrative model of student academic motivation (Figure 1.2) by Pintrich (1994). The model defines student motivation and some of the classroom contextual factors that can influence the students' motivation. Hence, this produce Figure 1.4 as the conceptual framework of the study.

There are three major variables in the Pintrich (1994) model:

1. The **contextual factors** which is defined as the various features of the classroom environment. These features are assumed to influence students' motivational beliefs, which is the second variable of the model:
   a) The nature of task
   b) The reward and goal structure of the classroom
   c) The instructional methods
   d) The instructor's behaviour

The definition agrees to the view that students need to be taught how to work in groups cooperatively, through direct instruction of modelling (Pintrich, 1994, p.38), aligned with the social cognitive theory of learning. Meanwhile, instructional methods and instructor behaviour are generally discussed can influence student motivation. It includes the quality of the instructional methods and the importance of the instructors' (lecturers') behaviours. It shows that there are strategies that the instructors (lecturers) can use to make their classrooms more motivating to students.

2. The **internal factors (students' motivational beliefs)**. According to Pintrich (1994), this variable can be divided into three types of beliefs, and defined as:
   a) **Expectancy** component can be defined as the students' control beliefs, attributions, learned helplessness and self-efficacy.
   b) **Value** component which consists of the students' intrinsic motivation, extrinsic motivation, task value and personal interests.

c) **Affective** component which consists of the students' test anxiety, self-worth and other emotions such as pride and shame.

Expectancy refers to the students' self-efficacy while learning. **Value** is defined as the student's aims and beliefs about learning and its significance. While **affective** is the students' emotional state during the learning process.

3. The **motivated behaviour**. The third variable in the model is the students' behaviours which are observable in the classrooms.

The value element comprises the student's goals and beliefs about a task and its importance. The expectancy element is often referred to as students' academic self-efficacy. This is the student's beliefs about their ability to perform a task. The affective element concerns to students' feelings or emotional reactions to either the task or the environment in general. To decrease concerns or anxieties, students need extra processing adjustment before they can turn back to the task.

As shown in Figure 1.2, the three major components are linked in reciprocal ways. Therefore, the major assumptions of this model presentation are:

1. Students' actual behaviour will provide feedback that influences their motivational beliefs.
2. Students' behaviour in the class will influence instructors' behaviour and actions.
3. Students are occupied with prior knowledge and beliefs before attending the course which can influence their perceptions on the course.
4. The three variables are interacting dynamically.

Contextual factors	Internal factors	Motivated behaviour
Classroom factors that can influence students' motivation	Students' motivational beliefs and emotional that are assumed to mediate between the context and behaviour	Actual observable behaviours that can be used as indicators of motivation
1) Nature of task – Content – Product	1) Expectancy components – Controls beliefs – Attributions – Learned helplessness – Self-efficacy	1) Choice behaviour – Working on course material instead of leisure activity – Electing to take another course in area of discipline – Choosing a discipline for a major – Choosing to go on to graduate school or a career in area
2) Reward and goal structure – Individualistic – Cooperative – Competitive	2) Value components – Intrinsic motivation – Extrinsic motivation – Task value – Personal interest	2) Level of activity and involvement – Trying very hard, putting forth a great deal of effort – Studying effectively, use of learning strategies – Thinking deeply, critically about material – Asking questions, taking risks in expressing ideas – High levels of performance/achievement
3) Instructional Methods 4) Instructor behaviour	3) Affective components – Test anxiety – Self-worth – Other emotions (pride, shame)	3) Persistent behaviour/Regulation of effort – Maintaining effort in face of difficulty – Maintaining effort on "boring" tasks – Maintaining effort when fatigue

Figure 1.2: An Integrative Model of Student Academic Motivation based on Pintrich (1994)

However, recent practices and technology seem to be moving from general classrooms contextual factors to more explicit characteristics. A learning framework like the How People Learn framework by the Bransford et al. (2000) explicitly defines four overlapping lenses on how to develop an effective learning environment for students. Accordingly, a consideration of the different components of motivational beliefs and their interactions to different characteristics of the learning environment may unfold multiple ways to facilitate students' motivation while learning. This framework (shown in Figure 1.3) combines four lenses of learning environment design. It synthesizes works in educational research and recognises four areas that lecturers should include to design effective learning environments.

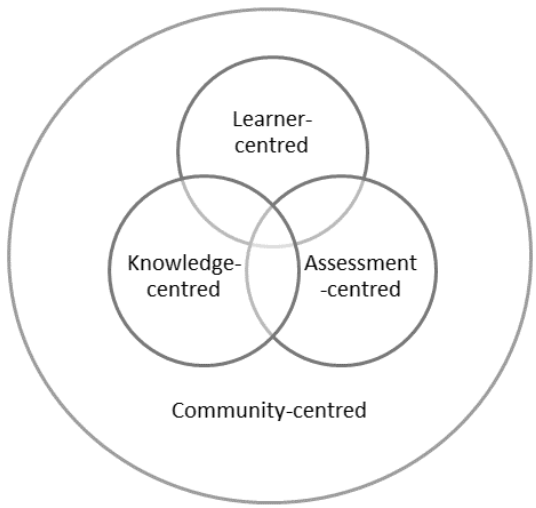

Figure 1.3: How People Learn (HPL) Framework (Bransford et al., 2000)

In the other words, it includes the theories about how students learn, the students' characteristics and what the students take along to the learning environment, the learning settings and their influence on learning process. The four lenses are:

1. **Knowledge-centered:** to craft the objectives of the course, to provide the students with fundamentals knowledge, skills and attitudes for a successful transfer. For this context of study, it is assumed that the students will be motivated due to the relevance of learning, applications of the knowledge and in-depth understanding.

2. **Learner-centered:** To connect the students' characteristics. For example, the students' strengths, interests, beliefs and prior knowledge. The characteristics will be the base for the lecturers to design the teaching and learning activities and become the foundation of the students' beliefs and behaviours during learning.
3. **Assessment-centered:** To provide opportunities for students to receive feedback and chances for revision during learning.
4. **Community-centered**: To provide a comfortable and safe learning environment for students to ask questions, adapt technology for learning and work in groups. It is also to help developing life-long learning environment. In this context, social opportunities are assumed to facilitate students' motivational beliefs.

Hence, for this study, the definition of the HPL lenses are also adapted as the comprehensive version of the contextual factors in the previous model by Pintrich et al. (1994). Figure 1.4 shows the conceptual framework used for this study.

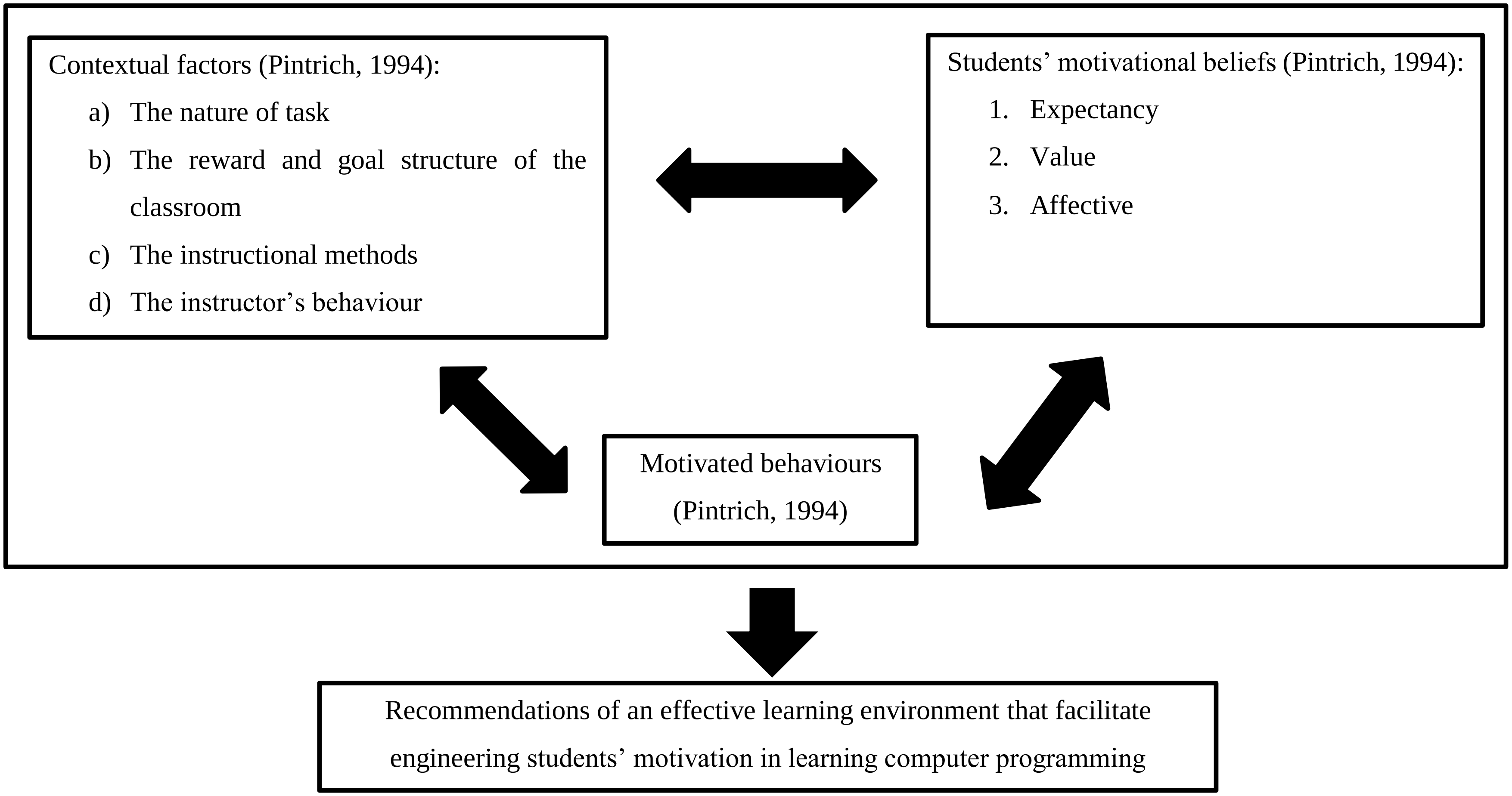

Figure 1.4: Conceptual Framework

1.9 Operational Definitions

This section provides definitions of terminologies that are used in this thesis. The definitions are derived from the literature, theories and the conceptual framework.

Computer Programming (or programming)

Computer programming is a technique to instruct electronic machines to execute tasks, solve problems and deliver human interactivity by breaking down a problem to small, manageable pieces that can be understood by a computer (Bebbington, 2014). Computer supported problem solving, incorporates development of problem solving strategies (not necessarily in computer science field) (Mladenović et al., 2016). In this study, computer programming refers to the course of Introduction to Computer Programming for engineering students, mainly for first-year chemical engineering undergraduates.

Contextual Factors

The various features of the classroom environment which are assumed may influence students' motivational beliefs (Pintrich, 1994). In this study, it mainly refers to the surrounding and any situations happen in the classroom.

Motivational Beliefs

The internal emotions that students have about themselves which can influence their perceptions on the learning environment (Pintrich, 1994). In this study, it refers to the self-thoughts of the students and their inner perception on their ability, and the learning environment.

Behaviour

The third variable in the model is the students' behaviours which are observable in the classrooms (Pintrich, 1994). This study holds that behaviour is any behaviours portrait by the students in the classroom.

1.10 Organisation of the Thesis

Chapter 1 explains the introduction of the problem background, problem statement, research objective, research question, significance of the study and theoretical and conceptual frameworks used to frame the study. Chapter 2 will present the literature review based on the previous studies and theories in education. This chapter will help the next coming chapters to contribute to the discussion and conclusion of the study. Chapter 3 explains the methodology employed in this study to conduct the data collection and the underlying principles of the method. It also explains the flow of the data collection and the data analysis process. Next, Chapter 4 presents the data and outcomes of the analysis in the form of themes and codings and how they are group into specific research questions. Finally, in Chapter 5, the results are discussed with reference with the literature and significant results. Implications for the area programming education in engineering education is also addressed in this chapter to conclude the main contribution of the work.

REFERENCES

Askar, P., & Davenport, D. (2009). An investigation of factors related to self-efficacy for Java Programming among engineering students. *Online Submission*, *8*(1).

Attaway, S. (2010). Using MATLAB to teach programming to first-year engineering students at Boston University. *Mathworks, available at http://www. mathworks. com/tagteam/65082_91847v00_NN10_FA_BU. pdf*.

Bandura, A. (1986). Social foundations of thought and action. *Englewood Cliffs, NJ*, *1986*.Bandura, A. (1997). *Self-efficacy in changing societies . Cambridge University Press*. https://doi.org/10.1177/135910539900400207

Bandura, A. (2001). Social cognitive theory: An agentic perspective. *Annual review of psychology*, *52*(1), 1-26.

Bandura, A. (1989). Human agency in social cognitive theory. *American psychologist*, *44*(9), 1175.

Ben-Ari, M. (1998, March). Constructivism in computer science education. In *Acm sigcse bulletin* (Vol. 30, No. 1, pp. 257-261). ACM.

Bennedsen, J., & Caspersen, M. E. (2006). Abstraction ability as an indicator of success for learning object-oriented programming?. *ACM Sigcse Bulletin*, *38*(2), 39-43.

Bennedssen, J., & Caspersen, M. E. (2008, September). Abstraction ability as an indicator of success for learning computing science?. In *Proceedings of the Fourth international Workshop on Computing Education Research* (pp. 15-26). ACM.

Borrego, M., Douglas, E. P., & Amelink, C. T. (2009). Quantitative, qualitative, and mixed research methods in engineering education. *Journal of Engineering education*, *98*(1), 53-66.

Bowen, J. D. (2004). Motivating civil engineering students to learn computer programming with a structural design project. *age*, *9*, 1.

Bradley, A. R. (2011). *Programming for engineers: a foundational approach to learning C and Matlab*. Springer Science & Business Media.

Bransford et al. (2000). *How people learn: Brain, mind, experience, and school: Expanded edition*. National Academies Press.

Caspersen, M. E., & Bennedsen, J. (2007, September). Instructional design of a programming course: a learning theoretic approach. In *Proceedings of the third international workshop on Computing education research* (pp. 111-122). ACM.

Chandramouli, M., Zahraee, M., & Winer, C. (2014, June). A fun-learning approach to programming: An adaptive Virtual Reality (VR) platform to teach programming to engineering students. In *IEEE International Conference on Electro/Information Technology* (pp. 581-586). IEEE.

Chou, P., & Hsiao, H. (2011). An alternative learning strategy to support engineering students' programming skills: a case study. *Global J. Eng. Educ*, *13*(1), 6-11.

Creswell, J. W. (2013). Educational research: Planning, conducting, and evaluating quantitative and qualitative research (4th ed.). Boston, MA: Pearson.

Crosby, R. W., Valentine, S., & Williams, T. L. (2014). Leveraging programming difficulty to improve understanding and perceptions of non-majors. *Journal of Computing Sciences in Colleges*, *29*(4), 27-35.

Crotty, M. (1998). *The foundations of social research: Meaning and perspective in the research process*. Sage.

Dabney, J. B., & Ghorbel, F. H. (2005). Enhancing an advanced engineering mechanics course using MATLAB and Simulink. *International Journal of Engineering Education*, *21*(5), 885.

Dagienė, V., Skūpas, B., & Kurilov, J. (2014). Programming assignments in virtual learning environments: developments and opportunities for engineering education. *International journal of engineering education. Dublin: Tempus Publications. ISSN*, 644-653.

dos Santos, M. T., Vianna Jr, A. S., & Le Roux, G. A. (2018). Programming skills in the industry 4.0: are chemical engineering students able to face new problems?. *Education for Chemical Engineers*, *22*, 69-76.

diSessa, A., & Abelson, H. (1989). Boxer: A reconstructible computational medium. *Studying the Novice Programmer*, 467-481.

Eckerdal, A., Thuné, M., & Berglund, A. (2005, October). What does it take to learn'programming thinking'?. In *Proceedings of the first international workshop on Computing education research* (pp. 135-142). ACM.

Erlandson, D. A., Harris, E. L., Skipper, B. L., & Allen, S. D. (1993). *Doing* *naturalistic inquiry: A guide to methods*. Sage.

Forte, A., & Guzdial, M. (2005). Motivation and nonmajors in computer science: identifying discrete audiences for introductory courses. *IEEE Transactions on Education*, *48*(2), 248-253.

Garcia, T., & Pintrich, P. R. (1996). Assessing students' motivation and learning strategies in the classroom context: The Motivated Strategies for Learning Questionnaire. In *Alternatives in assessment of achievements, learning processes and prior knowledge* (pp. 319-339). Springer, Dordrecht.

Gomes, A., & Mendes, A. J. (2007, June). An environment to improve programming education. In *Proceedings of the 2007 international conference on Computer systems and technologies* (p. 88). ACM.

Gomes, A., & Mendes, A. J. (2007, September). Learning to program-difficulties and solutions. In *International Conference on Engineering Education–ICEE* (Vol. 2007).

Gomes, A. J., Santos, A. N., & Mendes, A. J. (2012, July). A study on students' behaviours and attitudes towards learning to program. In *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education* (pp. 132-137). ACM.

Jenkins, T. (2002, August). On the difficulty of learning to program. In *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences* (Vol. 4, No. 2002, pp. 53-58).

Jerez, J. M., Bueno, D., Molina, I., Urda, D., & Franco, L. (2012). Improving Motivation in Learning Programming Skills for Engineering Students. *International Journal of Engineering Education*, *28*(1), 202.

Johri, A., & Olds, B. M. (Eds.). (2014). *Cambridge handbook of engineering education research*. Cambridge University Press.

Kak, A. (2009). Teaching programming.

Kinnunen, P., & Malmi, L. (2005). Problems in problem-based learning-experiences, analysis and lessons learned on an introductory programming course. *Informatics in Education*, *4*(2), 193.

Kinnunen, P., & Malmi, L. (2006, September). Why students drop out CS1 course?. In *Proceedings of the second international workshop on Computing education research* (pp. 97-108). ACM.

Kinnunen, P., & Malmi, L. (2008, September). CS minors in a CS1 course.

In *Proceedings of the Fourth international Workshop on Computing Education Research* (pp. 79-90). ACM.

Kinnunen, P., & Simon, B. (2011, August). CS majors' self-efficacy perceptions in CS1: results in light of social cognitive theory. In *Proceedings of the seventh international workshop on Computing education research* (pp. 19-26). ACM.

Krpan, D., Mladenović, S., & Rosić, M. (2015). Undergraduate programming courses, students' perception and success. *Procedia-Social and Behavioral Sciences*, *174*, 3868-3872.

Kurkovsky, S. (2006). Improving Student Motivation in a Computing Course for Non-Majors. In *FECS* (pp. 82-88).

Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic Inquiry*. *Sage.*

Lister, R., & Leaney, J. (2003, January). First year programming: let all the flowers bloom. In *Proceedings of the fifth Australasian conference on Computing education-Volume 20* (pp. 221-230). Australian Computer Society, Inc..

McKenna, A. F., Nocedal, J., Freeman, R., & Carr, S. H. (2005, October). Introducing a constructivist approach to applying programming skills in engineering analysis. In *Proceedings Frontiers in Education 35th Annual Conference*(pp. T3H-20). IEEE.

Mendes, A. J., Paquete, L., Cardoso, A., & Gomes, A. (2012, October). Increasing student commitment in introductory programming learning. In *2012 Frontiers in Education Conference Proceedings* (pp. 1-6). IEEE.

Mladenović, S., Krpan, D., & Mladenović, M. (2016). Using games to help novices embrace programming: From elementary to higher education. *The International journal of engineering education*, *32*(1), 521-531.

Moons, J., & De Backer, C. (2013). The design and pilot evaluation of an interactive learning environment for introductory programming influenced by cognitive load theory and constructivism. *Computers & Education*, *60*(1), 368-384.

Nor, N. M., Rajab, N., & Ismail, K. M. (2008). Educating the engineer of 2020: Malaysian scenario. In *International Conference on Engineering Education (ICEE2008), Budapest, Hungary* (pp. 1-4).

Or-Bach, R., & Lavy, I. (2004). Cognitive activities of abstraction in object orientation: an empirical study. *ACM SIGCSE Bulletin*, *36*(2), 82-86.

Pea, R. D., & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New ideas in psychology*, *2*(2), 137-168.

Pears, A. N. (2010, October). Enhancing student engagement in an introductory programming course. In *2010 IEEE Frontiers in Education Conference (FIE)* (pp. F1E-1). IEEE.

Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., ... & Paterson, J. (2007, December). A survey of literature on the teaching of introductory programming. In *ACM sigcse bulletin* (Vol. 39, No. 4, pp. 204-223). ACM.

Pintrich, P. R. (2004). A conceptual framework for assessing motivation and self-regulated learning in college students. *Educational psychology review*, *16*(4), 385-407.

Pintrich, P. R. (1991). A manual for the use of the Motivated Strategies for Learning Questionnaire (MSLQ).

Pintrich, P. R., Smith, D. A., Garcia, T., & McKeachie, W. J. (1993). Reliability and predictive validity of the Motivated Strategies for Learning Questionnaire (MSLQ). *Educational and psychological measurement*, *53*(3), 801-813.

Pintrich, P. R. (1994). Student motivation in the college classroom. *Handbook of college teaching: Theory and applications*, 23-43.

Ring, B. A., Giordan, J., & Ransbottom, J. S. (2007). Problem solving through programming. *Consortium for Computing Sciences in Colleges*, 61-67.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer science education*, *13*(2), 137-172.

Schunk, D. H., Meece, J. R., & Pintrich, P. R. (2014). *Motivation in education: Theory, research, and applications*. Pearson Higher Ed.

Serrano-Cámara, L. M., Paredes-Velasco, M., Alcover, C. M., & Velazquez-Iturbide, J. Á. (2014). An evaluation of students' motivation in computer-supported collaborative learning of programming concepts. *Computers in Human Behavior*, *31*, 499-508.

Sorva, J. (2012). *Visual program simulation in introductory programming education*. Aalto University.

Sun, W., & Sun, X. (2011). Teaching computer programming skills to engineering and technology students with a modular programming strategy. In *American Society for Engineering Education*. American Society for Engineering Education.

Svinicki, M. D. (2004). *Learning and motivation in the postsecondary classroom*. Anker Publishing Company.

Svinicki, M. D. (2010). A guidebook on conceptual frameworks for research in engineering education. *Rigorous Research in Engineering Education*, 1-53.

Urban-Lurain, M., & Weinshank, D. J. (2001). Do Non-Computer Science Students Need to Program?. *Journal of Engineering Education*, *90*(4), 535-541.