Primary School Students' Intrinsic Motivation to Plugged and Unplugged Approach to Develop Computational Thinking

Abstract: This paper compared primary school students' intrinsic motivation to plugged and unplugged approach to develop computational thinking using a revised Intrinsic Motivation Inventory. A total of 400 fourth-graders who have completed a school-provided coding course participated in the study. The revised instrument examined students' motivation of the two learning approaches from four dimensions: interest, perceived competence, value, and relatedness. The main findings of the study are: (1) primary school students showed moderate to high motivation to learn computational thinking through both plugged and unplugged approach; (2) compared to unplugged approach, students gained higher perceived competence from plugged approach; (3) the revised Intrinsic Motivation Inventory has good psychometric properties in the context of computational thinking except for the close correlation among different dimensions. Finally, implications for developing computational thinking with mobile devices were proposed.

Keywords: Computational thinking; coding education; programming; K-12

1 Introduction

Since the seminal paper of Wing (2006), computational thinking has attracted increasingly attention from educators and researchers. Regardless of various interpretations, a consensus has been reached that computational thinking refers to the thought processes that formulate problem as well as its solution in a manner that can be processed by computer (Chen et al., 2017; Wing, 2011). Specifically, computational thinking in its essence entails problem solving ability and reasoning ability (Román-González, Pérez-González, & Jiménez-Fernández, 2017). In addition to problem solving skill, developing computational thinking is also conducive to enhancing student's creativity, algorithmic thinking, and the awareness of cooperation (ISTE, 2015). Aligning with critical 21st century skills, computational thinking is also considered as a fundamental skill for every student. Particularly due to the demand for future workforce with computing skills, researchers and educators have argued the integral role of computational thinking in all STEM curriculum. In a special session of SIGCSE symposium, Henderson and colleagues (2007) argued that computational reasoning is key to STEM disciplines. Moreover, NRC (2012) pointed out in a framework for science education that using mathematical and computational thinking is one important component in engineering curriculum. In empirical studies, some researchers argued for the transdisciplinary approach of STEM epistemology by connecting computational thinking with physical computing in inquiry-based learning (Psycharis, Kalovrektis, Sakellaridi, Korres, & Mastorodimos, 2018). These effort include integrating computational thinking with robotics engineering to facilitate K-12 students' thinking and computational problem solving ability (Bers, Flannery, Kazakoff, & Sullivan, 2014; Grover, 2011; Witherspoon, Higashi, Schunn, Baehr, & Shoop, 2017). Besides robotics, several studies embedding computational thinking in other STEM curriculums have showed the effectiveness of this approach in learning mathematical process (Calao et al., 2015), physics and biology (Sengupta et al., 2013). Hence, it is believed that developing children's computational thinking could lay the foundation for engineering education and support learning in other disciplines.

Generally, programming (plugged) and CS Unplugged (unplugged) are two common ways to help students develop computational thinking and learn computational concepts. The idea behind unplugged activities where bodily movements are stressed is embodied cognition. This approach highlights that children learn new knowledge through bodily actions and perceptual experiences (Anderson,

2003), while the programming activity is rooted in constructionism which believes that learning is supported by creating personal artifact (Papert, 1980), such as digital games. Drawing on the theories of embodied cognition and constructionism, the design of plugged activity and unplugged activity often focuses on different aspects of learning experience, which may lead to different levels of intrinsic motivation to participate in an activity.

However, there is still little understanding regarding the effective way to motivate computational thinking learning especially from the perspective of intrinsic motivation. Furthermore, there is lack of research investigating primary school students' perception of the existing ways to learn computational thinking from the perspective of human innate psychological needs. Without attention to intrinsic motivation, students can have difficulties in persevering in learning as well as improving learning performance (Ryan & Deci, 2000b). In response, we attempt to address these issues by drawing on self-determination theory. On the one hand, primary school students' intrinsic motivation to develop computational thinking by plugged and unplugged approaches is assessed by a customized version of an existing instrument, Intrinsic Motivation Inventory (IMI) (McAuley, Duncan, & Tammen, 1989). On the other hand, the psychometric properties of IMI are reported to determine the appropriateness of the instrument in the new context. By investigating intrinsic motivation of primary school students, the present work will provide insights into the question of how to address computational thinking with children (Voogt, Fisser, Good, Mishra, & Yadavd, 2015) by accommodating age-appropriate needs and interest.

2 Constructionist and Embodied Approach to Computational Thinking

The term "computational thinking" is firstly put forward by Papert (1980), referring to procedural thinking. In Wing's (2006) article, computational thinking was defined rigorously as using concepts in computer science to solve problem, design system, and understand human behavior, after which research related to computational thinking in K-12 educational context has gained much traction (Lye & Koh, 2014). Traditionally, programming is regarded as a gateway to computational thinking and therefore has occupied a critical position in computational thinking curriculum (Kafai & Burke, 2013). Papert's (1980) Constructionism is an epistemological approach to teaching rooted in Constructivism. The two ideas share a central proposition: learning is constructing knowledge structure; nevertheless, constructionism stresses the importance of learner's engagement in creating public entity consciously (Papert & Harel, 1991). Specifically, Papert considered child as creator and builder who "learn by doing" with the aid of computer programming. Constructionism provides a new angle of view that learners are empowered to interact with the fundamental ideas from science and express themselves while actively constructing the artifact that is meaningful to them through programming. Visual-based programming is dominant programming environment in K-12 educational context due to its easy-to-use drag-and-drop features (Lye & Koh, 2014). Commonly used visual-based programming languages include Scratch, Alice, App Inventor, etc. It is reported that through doing programming, students enhanced computer science knowledge (Burke, 2012), problem solving skills (Chen et al., 2017), as well as transferrable skills (Grover, 2017). Researchers interested in examining the impact of programming on learners' perception suggested that participating in programming led to enhanced learning interest and enjoyment in various age groups (Tran, 2018) Furthermore, integrating computer

programming into content knowledge, such as mathematics, was found to be effective in increasing students' reasoning skills and self-efficacy (Psycharis & Kalia, 2017). More recently, drawing on the pervasiveness of mobile devices, innovative programming environments was created for preschool children to familiarize themselves with computational thinking, such as ScrtachJr. For example, a mixed-method study examined the effects of SctrachJr on kindergarten children's learning of computational concepts. The findings suggested that mobile assisted learning was effective and interesting even for kindergarteners (Papadakis, Kalogiannakis, & Zaranis, 2016). Acknowledge that developing computational thinking with mobile devices could take place without learning programming. For instance, a mobile application was designed by Fronza and Gallo (2016) to assist language learning along with computational thinking development. The application provided diagnostic assessments which could be useful to learners and teachers.

Apart from learning by programming, there exist another different approach to introduce students to computational thinking, which is related to embodied approach. Computer Science Unplugged is a series of activities used to help novices to learn computer science concepts in the absence of real computer (Bell, Alexander, Freeman, & Grimley, 2009), and thus keeping them from programming (Grover & Pea, 2013). The fundamental idea of unplugged activity is in line with embodied cognition that advocates cognitive process is highly related to sensorimotor experience (e.g. sight, touch, bodily movement) and outside environment (e.g. physical manipulative) (Anderson, 2003; Barsalou, 2008; Wilson, 2002). It means perceptual and motor experiences, such as using physical manipulatives in unplugged activity, impact learner's cognition and thus appropriate use of embodiment in instruction could be beneficial to learning. Actually, one of the major features of CS Unplugged activity is kinesthetic, which means physical objects (e.g. cards, weights) are usually used to replace computer (Nishida et al., 2009). Seeing the potential of embodied cognition and physical computing, researchers have made attempts to apply them to teach computational thinking. For example, a model which connected computational thinking and physical computing in inquiry-based activities was proposed to engage college students in STEM epistemology. According to the researchers, the model may have the potential to increase students' self-efficacy and internal motivation (Psycharis, Kalovrektis, Sakellaridi, Korres, & Mastorodimos, 2018). In addition, in an up-to-date research, Aggarwal, Gardner-McCune, and Touretzky (2017) used tiles and flashcards designed for Microsoft Kodu to introduce primary students to basic program called "Pursue and Consume" after school. The treatment group was given tiles and flashcards which allow manipulation by students, while controlled group was provided with a single paper with programming rules printed on it. For treatment group, learning, therefore, is grounded in sensory-motor experience due to the use of manipulatives and physical movement. Similarly, other researchers who tried to apply embodied approach in teaching computational thinking have found that pupils enjoyed the unplugged learning experience (Daily et al., 2015), and outperformed other students who learnt computational concepts by coding (Fadjo, Lu, & Black, 2009). In general, embodied approach seems to be particularly helpful for young children to get familiar with computational concepts (Bell et al., 2009; Conde et al., 2017) in the absence of digital devices.

Drawing on the theories of embodied cognition and constructionism, the design of plugged activity and unplugged activity often focuses on different aspects of children's learning experience. Plugged approach allows children to explore and create artifacts in front of a computer whereas unplugged activity often engages students in group work with bodily movements. Therefore, different learning experiences may lead to different levels of intrinsic motivation that drives children to participate in an activity autonomously. For example, some students may find writing programs interesting while other enjoy learning with tangible tools. Regarding interpersonal relatedness needs, unplugged approach may provide more opportunities for students to interact with peers whereas programming entails more teacher guidance to help students write a correct program. In self-determination theory, whether these aspects are satisfied could influence individual's subjective understanding of a particular activity and further influence intrinsic motivation. In addition, being intrinsically motivated is extremely important to learning since it reflects inherent tendency to accomplish a task, such as learning new knowledge. In fact, it is suggested that intrinsic motivation leads to higher learning performance and longer persistence (Ryan & Deci, 2000b). Hence, it is critical to understand and compare students' motivation of constructionist and embodied approach so as to better design the motivating learning activities for computational thinking.

Two measures are mostly used when intrinsic motivation is considered as dependent variable. First, behavioral measurement refers to participant's free choice in an experiment where the individual has the autonomy to do target activity or other distractive activity. The second measurement can be conducted through self-report instrument designed specifically to test individual's intrinsic motivation. For example, IMI is a report firstly validated by McAuley and colleagues (1989) based on self-determination theory. Besides students' enjoyment, the instrument also includes several dimensions regarding human psychological needs (e.g. relatedness, competence, autonomy) to gain a comprehensive picture of how students perceive a learning activity. As a result, this study revised and adopted the instrument to probe into the following research questions: (1) What is the psychometric properties of Intrinsic Motivation Inventory in the context of computational thinking learning? (2) Are students more intrinsically motivated to learn computational thinking by plugged and unplugged approach?

3 Method

3.1 Sample

The present study targeted five government funded primary schools in Hong Kong. 400 fourth-graders whose age range from 9 to 11 took part in the research. These students have completed a same six-week coding course in which programming and unplugged activities are incorporated to expose the students to basic computational concepts. Prior to the present study, students had no experience in coding in general. Therefore, all participants have the similar experience with coding and unplugged activity. However, to ensure the quality of data, 180 students' responses are excluded from analysis due to two reasons: (1) obvious patterns were identified in the response; (2) random filled answers were found (e.g. a majority of items in the instrument were filled in with the same choices). As a result, a total of 220 responses were analyzed.

3.2 Procedure

All participants were required to complete a coding course in which plugged and unplugged activities were integrated. In the coding course, students learnt computational concepts through unplugged activities and then created computational artifacts by engaging in programming tasks. Figure 1 shows the activities that were used to teach and apply the idea of "sequencing". In unplugged activity on the left side, students used pen and paper to draw a route between the starting point and the destination at the beginning. Then teachers guided students to simulate the scenario by moving around on the floor in order to understand "sequencing". The corresponding plugged activity on the right side required students to demonstrate "sequencing" by constructing a program in Scratch to manipulate the green sprite to reach the cat. In this regard, applying embodied learning before constructionist approach made it possible for students to learn the computational concepts first and internalize knowledge after. After completion of the coding course, a revised IMI was distributed to students either in the form of paper-based questionnaire or online survey. To ensure the quality of data, a check on all students' responses was conducted.

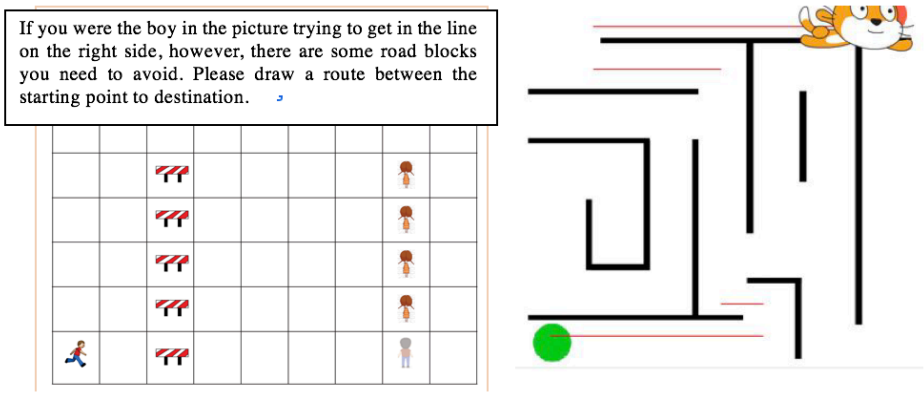

Figure 1 Example of unplugged (left) and plugged (right) activity

3.3 Instrument

All participants were given a survey which was adapted from IMI to measure their intrinsic motivation in programming activity and unplugged activity. The survey was a five Likert scale where strongly disagree=1, disagree=2, neither disagree or agree=3, agree=4, strongly agree=5. IMI is a generic instrument, meaning that it can be modified slightly to fit specific activities. Hence, to make a comparison between plugged and unplugged approaches to learn computational thinking, the survey was grouped into two parts (part A: programming; part B: unplugged) with the same statements in each part. Originally, there are seven dimensions in IMI covering respondents' self-efficacy, affective tendency (e.g. enjoyment, pressure), perception of the activity (e.g. usefulness, choice, interaction). However, it is worth noting that only four dimensions applicable to students' coding experiences were used in the present study, namely interest/enjoyment (Int) (e.g. part A "I enjoy doing programming activities very much", part B "I enjoy doing unplugged activities very much"), perceived competence (PC) (e.g. part A "I think I am good at programming", part B"I think I am good at unplugged activities"), value/usefulness (Use) (e.g. part A "I believe doing programming could be

beneficial to me", part B "I believe doing unplugged activities could be beneficial to me"), and relatedness (Rel) (e.g. part A "I'd like a chance to interact with my teacher more often in programming activities", part B "I'd like a chance to interact with my teacher more often in unplugged activities"), resulting in 14 items in each part. A total of 28 items form the revised IMI. While there is no specific item directly addressing students' perception of computational thinking *per se*, the instrument examines how they perceive the learning experiences in developing computational thinking through the two approaches.

Having taking into account of the participants' English skill and reading ability, the researchers translated the instrument from English into traditional Chinese. Firstly, it was interpreted to Chinese by one main researcher with a careful consideration of participants' reading ability. Four experts in computational thinking, including one local researcher, were involved in the subsequent stage to confirm the wordings of the instrument were clear and understandable. In the last stage, one teacher who taught the six-hour coding course was asked to comment on the instrument. The main researcher made modification to the instrument based on the comments and suggestions.

3.4 Data analysis

To validate the instrument in the new context, internal reliability and convergent validity were tested. Previous studies investigating the psychometric properties of IMI have reported that the adequacy of models varied from different contexts. For example, multi-factor model was adequate in sports setting (McAuley et al., 1989), while bi-factor model was good description of IMI in mathematics and language learning (Monteiro, Mata, & Peixoto, 2015). Hence, confirmatory factor analysis (CFA) were conducted using R package lavaan to test two models: single factor, multifactor. One thing worth mentioning is that students' responses to the two learning approaches were analyzed separately to have a more precise understanding of the psychometric properties of the instrument.

In addition to validation process, data were analyzed in SPSS 24.0 using standard techniques of descriptive statistics and inferential statistics. For inferential statistics, paired samples t-test was conducted to compare whether there is significant difference between students' motivation of plugged approach and unplugged approach.

4 Results

4.1 Psychometric properties of revised IMI

Average variance extracted (AVE), composite reliability, and Cronbach's α of both models were calculated to examine the instrument's convergent validity and internal reliability (Table 1). Cronbach's α for both sets of data exceeded the common threshold of 0.70, indicating that the modified IMI had a desirable internal consistency. The composite reliability was 0.948 and 0.932 for two sets of data respectively, which was much higher than the recommended value. AVE for programming activity was 0.497 which was slightly lower than common threshold of 0.5. To improve convergent validity, measurement error variance of each item was examined. Item with largest measurement error was item 11 ("While I was programming, I was thinking about how much I enjoyed it") from interest/enjoyment construct. Hence, item 11 was excluded to improve AVE. Then, the AVE of remaining items was tested again. The result was improved to 0.509 which satisfied the recommend threshold.

Table 1 Convergent validity and internal reliability

	Programming		**CS Unplugged**	
	Single factor	**Multifactor**	**Single factor**	**Multifactor**
AVE	0.497	0.537	0.567	0.595
Composite reliability	0.948	0.932	0.932	0.948
Cronbach's α	0.948	0.938	0.932	0.952

Discriminant validity is a measure testing whether constructs are distinct from each other. Table 2 presents the correlation matrix of the instrument, which showed unsatisfying discriminant validity. All off-diagonal coefficients were greater than 0.8, suggesting high correlation among different constructs. In other words, there was no clear distinction among the four constructs.

Table 2 Correlation matrix

	PC	***Int***	***Use***	***Rel***
PC	1.000			
Int	0.888	1.000		
Use	0.854	0.898	1.000	
Rel	0.947	0.954	0.973	1.000

The goodness-of-fit indices for single factor model in addition to multifactor model are tested. All the goodness-of-fit indices satisfied their correspondingly recommended value, indicating that both models had a good fit (Table 3). To be more specific, the multifactor model showed better model fit as all goodness-of-fit indices satisfied their criteria to a greater extent.

Table 3 Fit indices for two models

	Single Factor	**Multifactor**	**Recommended Value**
Chi-square/degree of freedom($\chi 2$/df)	2.32	1.96	< 3
SRMR	0.045	0.039	< 0.08
RMSEA	0.080	0.070	< 0.08
CFI	0.931	0.954	> 0.9
TLI	0.919	0.941	> 0.9
GFI	0.890	0.912	> 0.8
AGFI	0.849	0.869	> 0.8

4.2 Comparison between plugged and unplugged activities

The average score of all items in the revised IMI for programming and unplugged approach was 3.88 and 3.74, which indicated moderate to high level of intrinsic motivation of students. When looking into the four dimensions respectively, the average scores of both learning approaches were above the mid-point, meaning that students showed positive perception with respect to their enjoyment, self-efficacy, as well as the affordances of the two learning approaches, including usefulness and opportunity for interaction.

Notwithstanding the slightly lower mean scores of unplugged activities, the only statistically significant difference was found in PC ($p<.001$). With regard to other dimension: P value was .454 for Int dimension, .119 for Use dimension, .114 for Rel dimension. This indicated that students gained higher perceived competence from programming activity. Table 4 presents the comparison of plugged and unplugged approach.

Table 4 Comparison between plugged and unplugged approach

	Programming		**CS Unplugged**		**P**
	Mean	***SD***	***Mean***	***SD***	
PC	3.99	0.88	3.70	1.01	<.001
Int	3.81	0.96	3.76	1.04	.454
Use	3.89	0.91	3.77	0.99	.119
Rel	3.83	0.88	3.70	0.97	.114

5 Discussion

In this paper, we reported the findings of a pilot study that focusing on evaluating psychometric properties of revised IMI in the context of computational thinking and making comparison between primary school students' intrinsic motivation to plugged and unplugged approach.

5.1 Psychometric properties of revised IMI

The first aim of this research is to analyze the features of a modified version of Intrinsic Motivation Inventory, including validity, reliability, and internal structure. Therefore, this research made attempt to test the characteristics of single factor model and multifactor model. The results gained in the factor analysis revealed that both models (single factor and multifactor) fitted in with the study. In general, the instrument has desirable reliability and validity, except for the divergent validity which measures whether a dimension is distinct from others. This suggested the existence of intrinsic motivation as one general factor, which is consistent with previous study (Monteiro et al., 2015). Therefore, it can be concluded that the revised IMI was measuring respondents' intrinsic motivation in computational thinking learning. From the other hand, the high correlations among different dimensions were not commonly found in other researches where the instrument was used, the only exception was a study conducted by Fonseca and Paula Brito (2012) to assess students' attitude toward information technology. Three constructs in the questionnaire aligned with IMI, which were confidence, interest, and usefulness. Similarly, the three constructs were found highly correlated in this study. One possible interpretation for the analogous results is that children and even adolescents have difficulties in responding questionnaire that involves value, attitude, and belief, especially when the contents are not explicitly developed for their age groups. In conclusion, the revised IMI is a reliable instrument to measure children's intrinsic motivation of different learning approaches to learn computational thinking. Nevertheless, it still calls for further study and revision to address the high correlation among different dimensions.

5.2 Intrinsic motivation to plugged and unplugged approach

This section discusses on students' motivation from four aspects being measured in IMI. First, based on the overall results, students showed moderate to high intrinsic

motivation to develop computational thinking through both programming activity and unplugged activity. The theoretical ground of IMI is self-determination theory whose fundamental argument is that individual's enjoyment gained from participating in an activity generates intrinsic motivation (Deci & Ryan, 1985). Therefore, interest/enjoyment dimension is the only direct measure of intrinsic motivation in IMI. Since there is no significant difference between students' interest level, it indicated that plugged and unplugged activities have the equal potential to arouse pupils' learning interest. Two reasons might account for the reasons of students' learning interest in programming activities. First, programming empowered students to construct artifacts that reflect their inner feelings and ideas. It means students were able to express themselves through making digital game or scenario they care about (Brennan & Resnick, 2012). Moreover, Scratch is a "low threshold, high ceiling" tool characterized by abundant materials and easy-to-use interface (Maloney et al., 2010), which could naturally increase students interest. With respect to unplugged activities, it provided opportunities for students to get familiar with abstract computational concepts through interacting with concrete objects. This method freed students from constructing programs on computer. In light of the characteristics, young children were likely to feel more engaged in unplugged games and increase their interest through manipulating simple tools and implementing physical actions (Nishida et al., 2009). Hence, despite the different focuses, both constructionist and embodied approach succeeded in ensuring attractiveness to children.

Further, of equal importance to intrinsic motivation are human psychological needs, including self-efficacy, and social needs (Ryan & Deci, 2000a), mapping on the dimension of perceived competence and relatedness respectively. Particular attention should be paid to students' perceived competence as they showed statistically significant higher sense of mastery after programming activities, compared to unplugged activities. In fact, coding experience and students' self-efficacy has long been noticed in computer science education. Positive correlation between programming experience and students' confidence was reported in previous literature (Tsai, Wang, & Hsu, 2018), suggesting that increasing programming experience could lead to enhanced self-efficacy. Similar results in the present research revealed the importance of plugged approach in improving students' perceived competence as well. This could be explained by two reasons: the design of learning procedure of the coding course; the different characteristics of plugged approach and unplugged approach.

In the coding course, students progressed through learning computational concepts to creating computational artifacts by engaging in unplugged activities and programming tasks. This means that the major goal of unplugged activities was to introduce students to basic computing concepts, e.g. sequencing, loop, etc. In contrast, programming tasks were designed to provide opportunities for applying and internalizing the concepts in the process of creating personal digital products. Due to the course design, when recalling the programming experience, students were likely to refer to the whole learning procedure which started from learning concepts to constructing artifacts. As a result, it could contribute to higher perceived competence, compared to unplugged learning approach only. Furthermore, programming as a way to learn knowledge empowers learners to construct personal meaningful artifacts (Papert, 1980). It highlights the centrality of actively creating digital products in

constructing new knowledge, such as animation and games in the coding course. Plugged approach, therefore, makes it possible for students to see the actual outcomes of learning. Hence, learners consider themselves as creator rather than passive receiver of knowledge. However, unplugged activities only help students learn fundamental principles and ideas from computer science through manipulating non-digital tools. It is possible to understand that students are unable to gain a sense of competence through visible outcomes because they play the role of consumers of these existing tools and tasks instead of creators. In these regards, students in the study gained higher sense of mastery from plugged approach, probably due to the learning design of the coding course and different focuses of learning approaches.

Regarding the social aspect of psychological needs, plugged and unplugged approach were considered equally in fulfilling primary school students' needs for student-teacher interaction. Specifically, the average score to item "I feel close to my teacher" was low compared with other statements, which was a direct indicator of distant teacher-student relationship. From student' perspective, it can be inferred that teachers did not engage with students sufficiently regardless of plugged or unplugged activity, although some opportunities were provided. In addition, the item "I'd like to interact with teachers more often" received a high score, meaning that students actually required higher level teacher involvement. In fact, in a study involving primary school students, Skinner and Belmont (1993) reported that teacher involvement would directly impact how students perceive teacher-student relationship. It means that inadequate teacher involvement could have a negative influence on relationship between teacher and student. Moreover, it is believed that intrinsic motivation will decrease because the interpersonal relatedness need is not satisfied. As a result, to satisfy students' psychological needs of close relationship, it is imperative for teachers to involve in students' learning process and provide more guidance regardless of plugged or unplugged activity.

With respect to usefulness dimension, there is no difference between two approaches as well. Applying computational thinking in diversified problem contexts has been attracted attention for years. Some researchers argued that the uniqueness of computational thinking lies in its potential to be applied in other types of reasoning (Barr & Stephenson, 2011; Wing, 2006). In this study, some effort was put in emphasizing the link between solving computational problems and tackling authentic problems, leading to students' enhanced awareness of the value and usefulness computational thinking. To be more specific, a daily life scenario was usually depicted in each lesson to illustrate the usefulness of computational thinking. For example, a sprite watched a flashlight show in a park and would like to imitate the effect using loops. By engaging in unplugged activity and plugged activity, students learnt the computational concept and constructed a program to solve the computational problem. In this way, they learnt the importance of using different approaches in understanding and internalizing a computational concept. As a result, students showed their willingness to continue to learn computational thinking in their future studies not only because they improved their coding skills, but because learning to code was conducive to solving daily life problems. Considering that people tend to become self-regulated if they perceive some activities are valuable (Deci, Eghrari, Patrick, & Leone, 1994), increasing students' motivation requires more emphasis attached on the practical use of computational thinking. Especially in mandatory course offered by schools where some pupils could be less interested in programming, strengthening the idea of applying computational thinking in other authentic contexts could be a solution to increase

motivation, for example, the application of computational thinking in robotics, mathematics, etc.

5.3 Implication for developing computational thinking with mobile devices

Despite the promising overall results, some issues emerged in the study. On the one hand, regarding the results of quantitative data, students' responses to some items revealed issues related to interpersonal relatedness needs. It is found that students were not satisfied with the frequency and depth of teacher-student interaction despite different learning approaches. On the other hand, there are different characteristics of plugged and unplugged approach, which, we believe, makes it necessary to combine two approaches to support learning. Commonly speaking, implementing the combined approach requires physical movements carried out in an open space along with programming activities on computers. This means that students need to move around in classroom, which could cause inconvenience.

In light of the challenges, mobile learning could be a solution offering unique affordance to support the development of computational thinking. Actually, mobile assisted learning has been used in higher education and has been reported as effective to motivating college students to learn programming (Shrestha, Moore, & Nocera, 2011). In the context of K-12, some efforts have also been put in developing programming tools specifically for portable devices. As mentioned before, ScratchJr is a graphical programming language available on the iPad focusing on the needs of young learners ranging from kindergarten to second grade (Flannery et al., 2013). It addresses the lack of technologies for children to learn programming on mobile devices. For example, in the study conducted by Papadakis, Kalogiannakis, & Zaranis (2016), after a 13-hour programming experience, the preschoolsers found ScratchJr especially attractive. They participated in the programming activities with intrinsic interest due to the game-based problem-solving feature of the tool. Mobile learning itself, therefore, could be an efficient way to increase learning interest for computational problem solving.

Moreover, mobile learning is often concerned with offering convenience (Kynäslahti, 2003) and empowering social interactivity (Melhuish & Falloon, 2010). Integrating mobile learning in computational thinking learning will free students from moving between plugged and unplugged activities as portable device can easily be carried by learners. This could address the challenge of moving around classroom when combining unplugged approach and plugged approach that was identified in the research. Also, given the insufficient teacher involvement in the present work, mobile devices could be used to enhance teacher-student interaction though communication tools. Another advantage that we can take of mobile devices is the possibility to do diagnostic assessment. Besides students learning interest, how to assess the progress of computational thinking is of importance (Shute, Sun, & Asbell-Clarke, 2017). Drawing on the real-time communication characteristics of mobile devices, formative assessment can be implemented multiple times to gain a precise picture of students' thinking level.

Although there is still lack of studies investigating the effect of mobile learning on learning computational thinking, the finding from this study could provide insights on the potential of embedding mobile computing in learning and teaching.

6 Conclusion

The research made attempt to validate a revised IMI by testing two models with regard to internal validity, reliability, and goodness-of-fit indices and sought to understand to what extent primary school students are motivated to learn computational thinking through programming and unplugged approach. The paper indicates desirable reliability and validity of a revised Intrinsic Motivation Inventory for computational thinking learning in the primary school setting to some extent except for the high correlation between different dimensions. Besides, by comparing students' motivation to different approaches in learning computational concepts from four aspects, we found that students are intrinsically motivated by activities specifically designed for them. First, pupils find plugged and unplugged activities equally interesting and enjoyable. Second, programming task helps students gain a higher sense of self-perceived competence, which may be due to the learning procedure of the coding course in addition to the different focuses of the two learning approaches. Third, regardless of programming or unplugged activity, teacher involvement is a critical factor contributing to satisfaction of students' social needs. It is particularly important to enhance teacher-student interaction as students feel distant to teachers in the present study. Fourth, emphasis should be placed on helping students understand the usefulness of learning computational thinking to solving other problems. Especially when the learning content is demanding for novices, integrating authentic problems helps students recognize the possible application of computational thinking and hence become more self-regulating. In conclusion, the findings provide important information on students' perceptions of different learning activities, which sheds light on the important factors to notice in developing intrinsically motivating activities for computational thinking. This research also contribute to the body of knowledge regarding the potential of combining unplugged with plugged approach in motivating primary school students to learn computational concepts as well as solving computational problems.

7 Limitation and Future Research

Limitation of this study includes that there is only quantitative data at this stage. The discussions on the four aspects of students' motivation are drawn from the implementation of the coding course along with existing literatures. In response to this, qualitative data will be collected in future study investigating: (1) students' interpretation of the items in the instrument; (2) students' in-depth perception of their learning experience. Second, due to the school schedule, the instrument was administered at the end of the coding course, instead of each learning activity. Although we put instructions suggesting students completing the instrument while referring to learning materials, short memory span of primary school students put an obstacle to gain accurate information on students' perception. Therefore, future research will try to administer the instrument after each activity to have a better understanding. Third, in order to compare the difference between programming activity and unplugged activity, students needed to complete a questionnaire with two parts. Therefore, they needed to recall two kinds of learning experiences at the same time. As a result, it could

lead to confusion on differentiating experiences with the two approaches. In future research, the two parts could be completed in different time period to avoid interference. Also, the items used in the instrument could be reduced and simplified to lighten students' burden in the processing of reading.

References:

Aggarwal, A., Gardner-McCune, C., & Touretzky, D. S. (2017). *Evaluating the Effect of Using Physical Manipulatives to Foster Computational Thinking in Elementary School*. Paper presented at the Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education, Seattle, WA.

Anderson, M. L. (2003). Embodied cognition: A field guide. *Artificial intelligence, 149*(1), 91-130.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? *Acm Inroads, 2*(1), 48-54.

Barsalou, L. W. (2008). Grounded cognition. *Annu Rev Psychol, 59*, 617-645. doi:10.1146/annurev.psych.59.103006.093639

Bell, T., Alexander, J., Freeman, I., & Grimley, M. (2009). Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology, 13*(1), 20-29.

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education, 72*, 145-157. doi:10.1016/j.compedu.2013.10.020

Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking*. Paper presented at the Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada.

Burke, Q. (2012). The markings of a new pencil: Introducing programming-as-writing in the middle school classroom. *Journal of Media Literacy Education, 4*(2), 121-135.

Calao, L., Moreno-León, J., Correa, H., & Robles, G. (2015). Developing mathematical thinking with scratch: An experiment with sixth-grade students. In G. Conole, T. Klobučar, C. Rensing, J. Konert, & E. Lavoué (Eds.), Design for teaching and learning in a networked world (pp. 17–27). Cham: Springer.

Chen, G., Shen, J., Barth-Cohen, L., Jiang, S., Huang, X., & Eltoukhy, M. (2017). Assessing elementary students' computational thinking in everyday reasoning and robotics programming. *Computers & Education, 109*, 162-175.

Conde, M. Á., Fernández-Llamas, C., Rodríguez-Sedano, F. J., Guerrero-Higueras, Á. M., Matellán-Olivera, V., & García-Peñalvo, F. J. (2017). *Promoting Computational Thinking in K-12 students by applying unplugged methods and robotics*. Paper presented at the Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality.

Daily, S. B., Leonard, A. E., Jörg, S., Babu, S., Gundersen, K., & Parmar, D. (2015). Embodying computational thinking: Initial design of an emerging technological learning tool. *Technology, Knowledge and Learning, 20*(1), 79-84.

Deci, E. L., Eghrari, H., Patrick, B. C., & Leone, D. (1994). Facilitating internalization: The selfdetermination theory perspective. Journal of Personality, 62, 119-142.

Deci, E. L., & Ryan, R. M. (1985). The general causality orientations scale: Self-determination in personality. *Journal of research in personality, 19*(2), 109-134.

Fadjo, C., Lu, M., & Black, J. B. (2009). *Instructional embodiment and video game programming in an after school program*. Paper presented at the World Conference on Educational Multimedia, Hypermedia and Telecommunications, Chesapeake, VA.

Fonseca, A. M., & Paula Brito, A. D. (2012). Propriedades psicométricas da versão portuguesa do Intrinsic Motivation Inventory (IMIp) em contextos de actividade física e desportiva. *Análise Psicológica, 19*(1), 76. doi:10.14417/ap.344

Fronza, I., & Gallo, D. (2016). Towards mobile assisted language learning based on computational thinking. *Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10013, 141-150.

Grover, S. (2011). *Robotics and engineering for middle and high school students to develop computational thinking*. Paper presented at the annual meeting of the American educational research association, New Orleans, LA.

Grover, S. (2017). Assessing Algorithmic and Computational Thinking in K-12: Lessons from a Middle School Classroom. In P. J. Rich & C. B. Hodges (Eds.), *Emerging Research, Practice, and Policy on Computational Thinking* (pp. 269-288). Cham: Springer International Publishing.

Grover, S., & Pea, R. (2013). Computational Thinking in K–12 A Review of the State of the Field. *Educational Researcher, 42*(1), 38-43.

Henderson, P. B., Cortina, T. J., & Wing, J. M. (2007). *Computational thinking*. Paper presented at the ACM SIGCSE Bulletin.

ISTE. (2015). CT leadership toolkit. Retrieved from https://c.ymcdn.com/sites/www.csteachers.org/resource/resmgr/471.11CTLeadershiptToolkit-S.pdf

Kafai, Y. B., & Burke, Q. (2013). Computer programming goes back to school. *Phi Delta Kappan, 95*(1), 61-65.

Kynäslahti, H. (2003). In search of elements of mobility in the context of education. Mobile learning, 41-48.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior, 41*, 51-61.

Maloney, J., Resnick, M., Rusk, N., Silverman, B. and Eastmond, E. (2010) 'The scratch programming language and environment', A*CM Transactions on Computing Education*, Vol. 10, No. 4, pp.1–15.

McAuley, E., Duncan, T., & Tammen, V. V. (1989). Psychometric properties of the Intrinsic Motivation Inventory in a competitive sport setting: A confirmatory factor analysis. *Research quarterly for exercise and sport, 60*(1), 48-58.

Melhuish, K., & Falloon, G. (2010). Looking to the future: M-learning with the iPad. Computers in New Zealand Schools: Learning, Leading, Technology, 22(3). Retrieved from http://education2x.otago.ac.nz/cinzs/mod/resource/view.php?id=114

Monteiro, V., Mata, L., & Peixoto, F. (2015). Intrinsic Motivation Inventory: psychometric properties in the context of first language and mathematics learning. *Psicologia: Reflexão e Crítica, 28*(3), 434-443. doi:10.1590/1678-7153.201528302

National Research Council (NRC). (2012) *A framework for k-12 science education: practices, crosscutting concepts, and core ideas, Committee on a Conceptual Framework for New k-12 Science Education Standards*, The National Academies Press, Washington, DC.

Nishida, T., Kanemune, S., Idosaka, Y., Namiki, M., Bell, T., & Kuno, Y. (2009). *A CS unplugged design pattern*. Paper presented at the ACM SIGCSE Bulletin.

Papadakis, S., Kalogiannakis, M., & Zaranis, N. (2016). Developing fundamental programming concepts and computational thinking with ScratchJr in preschool education: a case study. *International Journal of Mobile Learning and Organisation, 10*(3), 187-202.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.

Papert, S., & Harel, I. (1991). Situating constructionism. *Constructionism, 36*(2), 1-11.

Psycharis, S. and Kallia, M. (2017) 'The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving', *Instructional Science*, Vol. 45, No. 5, pp.583–602.

Psycharis, S., Kalovrektis, K., Sakellaridi, E., Korres, K., & Mastorodimos, D. (2018). Unfolding the Curriculum: Physical Computing, Computational Thinking and Computational Experiment in STEM's Transdisciplinary Approach. *European Journal of Engineering Research and Science*, (CIE), 19-24.

Román-González, M., Pérez-González, J.-C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior, 72*, 678-691. doi:10.1016/j.chb.2016.08.047

Ryan, R. M., & Deci, E. L. (2000a). Intrinsic and extrinsic motivations: classic definitions and new directions. *Contemporary Educational Psychology, 25*(1), 54-67. doi:10.1006/ceps.1999.1020

Ryan, R. M., & Deci, E. L. (2000b). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American Psychologist, 55*(1), 68-78. doi:10.1037/0003-066X.55.1.68

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies, 18*(2), 351-380.

Shrestha, S., Moore, J., & Nocera, J. A. (2011). Evaluation of a hands-on approach to learning mobile and embedded programming. *International Journal of Mobile Learning and Organisation*, *5*(3-4), 327-344.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142-158

Skinner, E. A., & Belmont, M. J. (1993). Motivation in the classroom: reciprocal effects of teacher behavior and student engagement across the school year. *Journal of Educational Psychology, 85*(4), 571-581. doi:10.1037/0022-0663.85.4.571

Tran, Y. (2018). Computational Thinking Equity in Elementary Classrooms: What Third-Grade Students Know and Can Do. *Journal of Educational Computing Research*, 073563311774391. doi:10.1177/0735633117743918

Tsai, M.-J., Wang, C.-Y., & Hsu, P.-F. (2018). Developing the Computer Programming Self-Efficacy Scale for Computer Literacy Education. *Journal of Educational Computing Research*, 0735633117746747.

Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies, 20*(4), 715-728.

Wilson, M. (2002). Six views of embodied cognition. *Psychonomic bulletin & review, 9*(4), 625-636.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.

Wing, J. M. (2011). *Computational thinking*. Paper presented at the VL/HCC.

Witherspoon, E., Higashi, R., Schunn, C., Baehr, E., & Shoop, R. (2017). Developing Computational Thinking through a Virtual Robotics Programming Curriculum. *ACM Transactions on Computing Education (TOCE), 18*(1), 1-20. doi:10.1145/3104982