Mutual Improvement between Teaching Materials and Assessment Tools for K-12 Programming Education

Hideo Nagumo
Dept. of Clinical Psychology
Niigata Seiryo University
Niigata, Japan
nagumo@n-seiryo.ac.jp

Yasumasa Oomori
College of Education
Joetsu University of Education
Joetsu, Japan
oomori@juen.ac.jp

Yasuhiro Takemura
Dept. of Art Science
Osaka University of Arts
Osaka, Japan
yasuhi-t@osaka-geidai.ac.jp

***Abstract*— This Innovative Practice Full Paper presents an approach to improve teaching materials and assessment tools for K-12 programming education together by creating a virtuous cycle. Improving teaching materials for computing education is in line with the conference theme "Creating a convergence in engineering education and workforce development". In this research, coding sheets are used as a part of the teaching materials, and the assessment tools are used to assess the algorithmic thinking ability of the students. The assumption here is that there is a mutual impact between the improvement of teaching materials and that of assessment tools. This kind of strategy is important, especially in the setting of elementary school education in Japan in which conducting a pilot study is difficult. In Japan, the new curriculum guidelines were fully implemented in April 2020 in elementary schools and in April 2021 in junior high schools respectively. They will be implemented in April 2022 in high schools. Since the announcement of the new curriculum guidelines, the boards of education in the local governments have been preparing for the programming education in the schools. However, there are still remaining problems, including the shortage of trained teachers and the lack of good assessment tools. In order to remedy the shortage of trained teachers, we designed some coding sheets for the teaching materials consisting of block parts, servo motors, DC-motors, LEDs, and sensors. In our assessment tools which are supposed to assess students' Computational Thinking ability, we included only the questions for assessing algorithmic thinking ability, as there is a limited amount of time for the assessment in schools. We have used our teaching materials and our assessment tools in some elementary schools. Based on the results of the teaching activities, we discuss the strategy to improve the teaching materials and the assessment tools together.**

Keywords*— *programming education, teaching material, assessment tool, Computational Thinking, elementary school

I. INTRODUCTION

Today, computer programming is viewed as a critical and necessary skill that everyone should learn [1,2], though, in the past, it was considered as a task carried out only by engineers, computer scientists, and computer geeks [3,4]. Consequently, programming education in high schools, middle schools, and elementary schools is being promoted around the world. Since programming education in elementary schools is relatively new, there are many challenges such as how to make a curriculum, what programming environment and lesson method should be used, and how to assess the learning [5]. Therefore, many research activities are underway in programming education for elementary school children [6]. In most countries, the purpose of programming education from elementary to high school is not to teach coding with a particular programming language, but to foster Computational Thinking (CT) [7]. CT involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to Computer Science (CS) [8].

In Japan as well, programming education in elementary schools, junior high schools, and high schools is increasingly seen as important. The Ministry of Education, Culture, Sport, Science and Technology (MEXT) has established the standards by which each school organizes its curriculum according to the School Education Act. This is to ensure that students receive a certain level of education, no matter where they are educated. These standards are referred to as curriculum guidelines. The new curriculum guidelines were fully implemented in April 2020 in elementary schools and in April 2021 in junior high schools respectively. They will be implemented in April 2022 in high schools. The new curriculum guidelines stipulate that the programming education is compulsory in elementary schools, and that the learning activities involving computer programming be strengthened in junior high schools and high schools. As is the case worldwide, the purpose of the programming education in those schools is to nurture CT.

Since the announcement of the new curriculum guidelines, the boards of education in the local governments have been preparing for the programming education in the schools. Especially, they distributed tablet computers to each elementary and junior high school student from 2020 to early 2021 under the GIGA (Global and Innovation Gateway for All) School concept. This provided an environment in which each student could program. Also, the MEXT has created a database of practical examples of programming education. However, there are still remaining problems, including the shortage of trained teachers and the lack of good assessment tools.

The research question of this work is how can we tell if a school child has acquired CT ability, and the purpose of this research is to improve teaching materials and assessment tools for K-12 programming education together by creating a virtuous cycle. In this research, coding sheets are proposed as the teaching materials with which children can learn programming without much interaction with teachers. The ideas of coding sheets in this study are borrowed from Scratch coding cards. Especially, in the experience class presented in this paper, the coding sheets are used to explain how to program a robot built with block parts, servo motors, LEDs, and some sensors. These coding sheets can realize small step learning. In addition, they can change the role of a teacher from an instructor to a tutor. Also, an assessment tool for evaluating the CT ability of children is proposed. This assessment tool actually measures the ability of algorithmic

thinking, which is a typical element of computational thinking. This is because there is a limited amount of time for the assessment in schools. The assumption here is that there is a mutual impact between the improvement of teaching materials and that of assessment tools. This kind of strategy is important, especially in the setting of elementary school education in Japan in which conducting a pilot study is difficult.

A programming experience class was conducted in an elementary school in Niigata prefecture Japan in July 2019 with the permission of the principal. Based on the results of the pre-test and post-test using the assessment tool, the ways of improving the teaching materials, teaching methods, and the assessment tool was proposed.

II. Related Work

A. Development of Teaching Materials

Over the last decade, the idea of CT has become popular around the world. It just overlaps with the fact that programming education has been introduced to elementary schools all over the world. Along with this, research on CT is actively being carried out. The themes of CT research include the definition and components of CT, the relationship between CT and computer science and programming, interventions for developing CT, and assessment of CT [5]. Among them, assessment of CT ability is considered to be a major weakness in this area of research. One reason for this is there is not a clear definition of CT. An operational definition of CT is needed both to develop interventions (such as teaching methods, programming environments, the equipment to be programmed etc.) to enhance CT and to assess the CT ability of school children. Without a clear definition of CT, it would be impossible to create pedagogical material that seeks to strengthen it. In addition, it would be difficult to tell what to assess specifically when trying to assess CT ability.

For the clear definition of CT, V. Barr and C. Stephenson [9] defined nine core CT concepts: Data collection, Data analysis, Data representation, Problem decomposition, Abstraction, Algorithms and procedures, Automation, Parallelization, and Simulation. In addition, the International Society for Technology in Education and the Computer Science Teachers Association have developed an operational definition of CT for all K-12 educators [10]. This definition stated that CT was a problem-solving process that included (but is not limited to) six characters such as automating solutions through algorithmic thinking.

Through the definition of CT, it is considered that interventions to enhance CT and tools for assessing CT are interrelated, because an assessment tool is needed to evaluate the intervention, and experimental lessons with an intervention is required to test an assessment tool. For this reason, this research associated teaching materials for programming education and assessment tools for CT ability. However, there are not so many research activities relating the two themes, though there are many research activities on teaching materials for programming education for K-12, and on assessment tools for CT ability. The following are the papers aimed at developing teaching materials of programming education to promote CT.

I. Ioannou and C. Angeli [11] examined the students' development of Computational and Algorithmic Thinking by utilizing the framework of Technological and Pedagogical Content Knowledge and the instructional design model of Technology Mapping (TM). In their experiment, they divided 240 8th grade participants into an experimental group with 127 children and a control group with 113 children. In their lessons, they used Robomind 4.3 and Lego NXT Mindstorm for the experimental group, and Alice 3.1 for the control group. In order to assess participants' conceptions about the algorithms and CT, they administered a pre-test and post-test, each of which took ten minutes. The results of the two tests indicated that their theoretical framework and the instructional design model facilitated the development of Computational and Algorithmic Thinking of students.

Fernandes et al. [7] proposed developing CT and literacy skills through an approach to creating games. They stated that CT skills are worked on when students think about the problem of the game, its decomposition, the algorithms with the rules, the change of phases, and the overcoming of challenges. Their target learners were primary school children, but they experimented their teaching method with 22 students of higher education. The students manipulated printed cards containing images of characters, objects, actions, and scenarios that should be used in the creation of the idealized games. To evaluate and analyze the results, the researcher interviewed the students and the teacher to analyze qualitatively the materials produced, the strategy proposed, and its potential impact on student learning.

Fanny et al. [12] proposed to focus on the discourses made by teachers and learners during robot literacy activities. They stated that the robots could be used to introduce learners to the fundamental concepts of computer science and CT. Their target learners were children aged 8-15 years. In their experiments, they used Bee-bot, Blue-bot, and/or Ozobot. Data were collected through ethnographic observation, and analyses were carried out using the conceptual metaphor theory. They identified three roles of metaphors, (1) The metaphor that helps to understand, (2) the metaphor that makes tangible, and (3) the metaphor that serves as a catchphrase.

Storjak et al. [6] organized innovative STEM education workshops in which two learning scenarios, unplugged game-like coding activities and puzzles, and then applying acquired knowledge to the controlling of educational robots, were practiced. The target group of the workshop were preschool and elementary pupils, and the activities in the workshop were designed to promote CT. In their research, data acquisition was performed by using pre- and post-questionnaires and the unplugged programming worksheets. The questions in the questionnaire were created to measure possible changes in the perception of the pupils about programming and robotics before and after the workshop. They found that pupils liked the unplugged activity more than the robot play.

Shim et al. [3] proposed an educational programming environment for elementary school students with which students can easily learn and practice computer programming. This environment used a tangible programming tool with which students can easily create robot programs, without learning syntax, and validate their programming results. There were three purposes for creating such an environment. The first purpose is to allow young students to readily understand the programming tools and use them intuitively. The second is to arouse students' interest and motivation on programming. The third is to make students concentrate on, and immense themselves in the programming process. They used this programming environment for 50 elementary school students

to analyze the elements of usability, edutainment, programming attitude, and understanding of programming concepts.

B. Development of Assessment Tools

The works mentioned above aimed to develop various teaching materials, pedagogical strategies, or instructional interventions for promoting students' CT ability. In these cases, the teaching materials, pedagogical strategies, and instructional interventions were designed in such a way that the concepts such as algorithmic thinking, abstraction, and modeling, which are considered to be the components of CT, are acquired by the learners. In order to check the effectiveness of the tools, strategies, and interventions, it is required to apply some assessment instruments. On the other hand, the following previous works aimed to develop or propose assessment tools for CT. In these cases, the assessment tools were designed in such a way that the acquired levels of one or several concepts among algorithmic thinking, abstraction, and modeling could be measured with them. In order to check the effectivness of the assessment tools, it is required to test them in an experiment with some teaching materials, pedagogical strategies, or instructional interventions.

González et al. [13] developed CT Test (CTt) following the practical guide to validating computer science knowledge assessment and other CT tests developed by other researchers. CTt aims to measure the development level of CT in the subject. The target population was 5th to 10th grade school children. It contained 28 multiple choice items with four answer options each. The question items were presented in either 'The Maze' or 'The Canvas'. The time needed for the test was 45 minutes. The CTt was administered on a total sample of 1251 children to confirm the reliability and criterion validity. As the criterion validity, it was confirmed that CTt correlated with Primary Mental Ability battery and also with RP30 problem-solving test. Their CTt was not developed for use in pre- and post-test.

Grover et al. [14] used the multiple forms of assessments in a 6-week middle school curriculum to capture a holistic view of student learning. The curriculum was designed to promote the idea of foundational computational concepts such as algorithmic flow of control comprising sequence, looping constructs, and conditional logic. Their goal was to study multiple and novel mechanisms for assessing learning of these core computational concepts, and helping to refine the curriculum. Formative assessment was integrated throughout the course as multiple-choice quizzes designed to give learners encouraging feedback and explanation. These quizzes included small snippets of Scratch code on which questions were based. They conducted two studies, Study 1 and Study 2 in a middle school, and they altered some strategies in Study 2 based on the assessment results of Study 1.

Oliveira et al. [15] proposed to use Drawing Machine Model (DMM), which is a model of computation based on the Turing Machine model, to assess student's ability to compute. They designed a test with seven objective questions using a DMM, each containing five alternatives. The test is divided into two parts. Part 1, which is composed of question 1 through 7, evaluates the student's ability to abstract and calculate. Part 2, which is composed of question 5 and 7, assesses the student's ability to read, understand, design and write. The target population was 6th to 9th grade school children. As the result of their experiment, in which 81 students participated, they observed strong and moderate correlation between the ability to compute and student performance in the school.

Werner et al. [16] created an assessment tool for measuring CT in middle school in which students were supposed to answer three tasks of Alice. They designed the tasks to measure aspects of algorithmic thinking, abstraction, and modeling. The data in their research were collected over two years in their study of how game creation and pair programming can promote CT. A total of 325 middle school students participated in the study. In this study, they found that computer game courses for middle school students were a promising strategy to introduce CT to a broad population, and that pair programming was effective in acquiring CT.

Koh et al. [17] developed a system that allowed teachers to see which high-level CT concepts students have mastered and which ones they are struggling with as students code in real time. This system was named REACT (Real Time Evaluation and Assessment of Computational Thinking). The REACT system was embedded into the online publicly available Scalable Game Design Arcade, and its testing took place over four weeks with 134 participants of 6th grade. In their results, they received an overwhelmingly positive reaction from teachers who participated in the study.

Araujo et al. [18] investigated how appropriate the Bebras challenge was as an instrument to assess and measure CT abilities. The participants in their study were 160 CS undergraduates who attended an introductory programming course in which the Python programming language was used. Those students took two, simulated multiple choice, Bebras tests before and after the term. Their result was that the two tests did not show a statistically significant difference.

In this way, a number of research activities have been carried out to develop teaching materials for fostering CT and to develop evaluation tools for measuring CT. However, no attempt has been made to link teaching materials and assessment tools for programming education and to improve both together.

III. Background

A. Programming Education in Japanese Elementary Schools

Though programming is compulsory in Japanese elementary schools, there are no subjects that teach computer science. Instead, programming is to be taught in traditional subjects such as English, math, and science. That means that English, math, and science teachers teach programming. Therefore, there is a concern that there will be a shortage of teachers who are fully trained in programming education. In urban elementary schools, university students and IT company staff may be able to come to support programming education in elementary schools. In rural elementary schools, however, this is not possible. Therefore, there is a need for a method that allows students to learn programming in small steps while helping each other.

B. Teaching Materials

The coding sheets proposed in this research can realize this kind of programming education. Examples of coding sheets are shown in Fig. 1. These coding sheets have similarity with the digital worksheets presented by Serth et al. [19]. Their digital worksheets, however, are very complex in that they

consist of texts, video quizzes, and practical programming exercises. The coding sheets presented in this research are the sheets with simple instruction of small steps in programming.

In this research, robot programming kits called ArtecRobo2.0 [20] were used in the programming education for elementary school children. ArtechRobo2.0 consists of Artec blocks, robot parts, and dedicated software. Artec blocks are plastic blocks with which children can build robots of any shape. Robot parts are electronic part such as servo motors, DC-motors, LEDs, and sensors, which are blocks too. The dedicated software is a visual programming environment based on Scratch 3.0. Robots built with the kit are shown in Fig. 2.

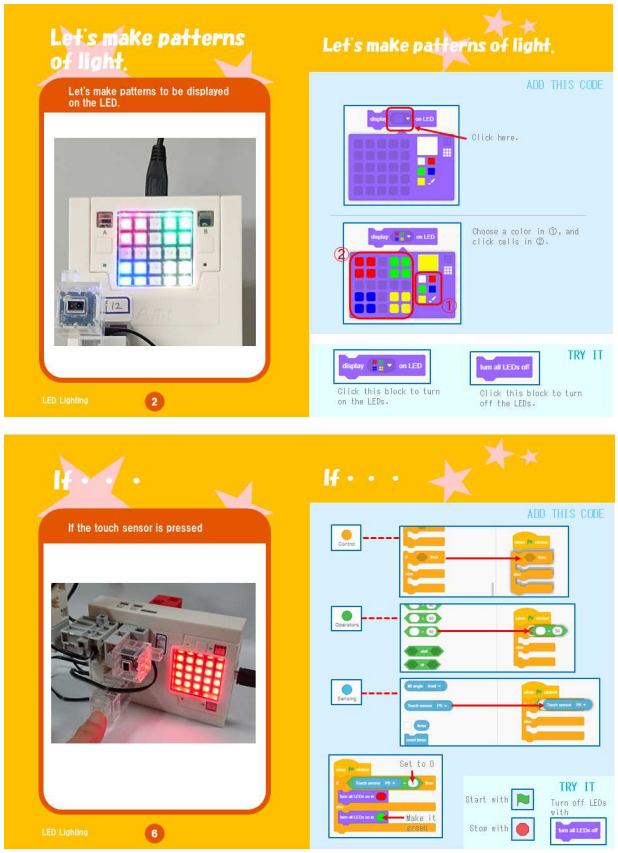

Fig. 1. Examples of coding sheets

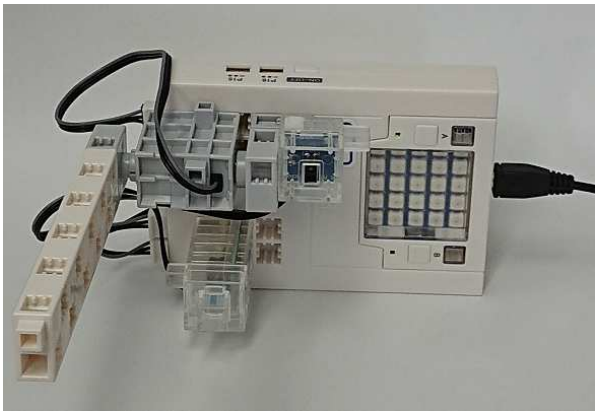

Fig. 2. An example of a robot with ArtecRobo2.0 kit

In the programming lessons for elementary school children with these robots and coding sheets, children are supposed to be divided into groups of two. The instructor does not explain how to use the programming environment nor did he/she explain how to program. Instead, he/she gives out one coding sheet to each pair of children in each step, and encourages children to teach each other.

Factors to be improved regarding these teaching materials and teaching methods are (1) the level of acquisition of CT, (2) difficulty level appropriate for the child's age, and (3) degree of interest of children. Of course, these factors are also influenced by the duration of programming lessons, as well as the frequency with which programming lessons are conducted. However, it is difficult for researchers to freely decide the duration and interval of experimental programming classes in Japan. They are usually determined by the convenience of the school.

C. Assessment Tools

Among the components of CT, the assessment tool developed in this research evaluates the level of algorithmic thinking ability. Therefore, each question in the tool is related to sequential processing, conditional branch processing, iterative processing, or combinations thereof. When solving these questions, school children have to interpret the representation of the algorithms. There are several different expressions of algorithms such as ordinary Japanese, structured Japanese, representation in blocks, flow charts, activity diagrams, Nassi-Shneiderman charts. The expression of the algorithm should be independent of any programming language because the purpose of programming education is not to learn a particular programming language, but to acquire CT ability. Therefore, the expression in blocks with which children can comprehend the constructs of algorithms easily, is used in this research. When designing questions regarding CT, the environment-interfaces: "Maze" and "Canvas" are common in popular sites for learning programming such as Code.org [13,21]. The environment-interfaces used in the assessment tool are basically "Canvas" in which children draw some diagrams.

Examples of questions in the assessment tool are shown in Fig. 3. In this figure, the left-hand side is a question with an expression of the algorithm. The right-hand side is the worksheet on which children draw their answers. The correct answer is shown on the worksheet. In the assessment tool, there are eight similar questions.

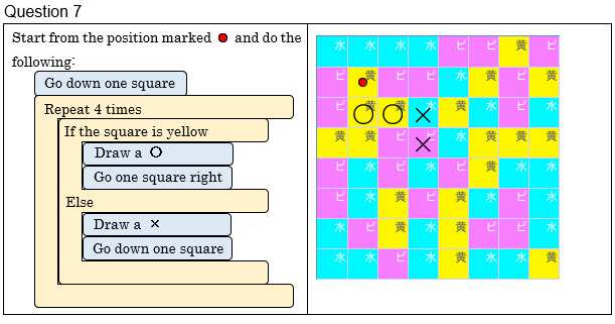

Fig. 3. Example of a question in the assessment tool

Table I shows the algorithm configuration for each question. When school children answer this assessment tool for the first time, examples like the ones in Fig. 4 are shown to the children to help them understand how to answer. Also, the procedure to answer the questions is explained to the children with a five-minute video. In this way, differences in

explanations by teachers are avoided. After the explanation, school children spend ten minutes to answer the questions in the assessment tool.

When scoring the answers of a child, one question is evaluated as correct or incorrect, with no intermediate points. By doing so, unambiguous scoring is possible. There are eight problems in the test, all of which have the same weight of 1. Therefore, one person's score is an integer between 0 and 8. The assessment tool is conducted as a pre- and post-test, and the questions for them are the same. Factors to be improved regarding the assessment tool and evaluation method are (1) difficulty level appropriate for the child's age, (2) answer time, and (3) reliability / validity. The ideal difficulty level of a test is that the mean value is half of the full-score, and the distribution is normal, because in this situation the analysis would be most accurate.

TABLE I. THE ALGORITHM CONFIGURATION FOR EACH QUESTION

Question	Algorithm configuration
1	Sequential processing of 4 tasks.
2	One task is followed by a conditional branch with "else" syntax. One task is contained in "if" syntax of the conditional branch, and one task is contained in "else" syntax.
3	Sequential processing of 4 tasks is contained in the repetition.
4	Sequential processing of 11 tasks
5	There are 2 conditional branches with "else" syntax in between 2 tasks. For each of the 2 conditional branches, 2 tasks are contained in both "if" and "else" syntaxes.
6	There are 3 consecutive repeated syntaxes. Each repeat syntax contains 2 tasks.
7	One task is followed by a repeat syntax. Conditional branch with "else" syntax is nested in the repeat syntax. There are 2 tasks in the "if" syntax of the conditional branch, and 2 tasks in the "else" syntax.
8	One task is followed by a 2 conditional branches, and a repetition is nested in each conditional branch. Two tasks are contained in each repeat construct.

IV. METHODOLOGY

A. Pre-Test

Using the teaching material, a programming experience class was held for 22 sixth graders (11 boys and 11 girls) from an elementary school in Niigata prefecture, in July 2019. None of the school children had prior programming experience. The class consisted of two consecutive lesson of 45 minutes. the author was the instructor of the class. The first lesson started at 9:35 and the second one at 10:40. Before starting the first lesson, pre-test with the assessment tool was conducted. At first, a number was given to each child so that each child can write the same number both on pre- and post-test. Apart from this number, children wrote down their gender information on the test sheets. Then, the students viewed the five-minute video to understand how to answer the questions in the tool. After that, they answered eight questions in the assessment tool in ten minutes.

B. Robot Setting

In the experience class, the 22 school children were grouped into 11 pairs, and each pair was provided with a robot shown in Fig. 2. Since there was not enough time to let the children build the robot using the blocks, already completed robots were handed out to the children. Also each pair was provided with a laptop computer in which the dedicated software for ArtecRobo2.0 is installed. In Fig. 2, There is a 5x5 full color LED matrix on the right side. Middle top is an IR photo-reflector that can detect an object in front of it. Left top is a servomotor to which an arm is attached. Left bottom is a touch sensor. the scenario for creating a program in this educational practice is to simulate the opening and closing gate bar of a parking lot.

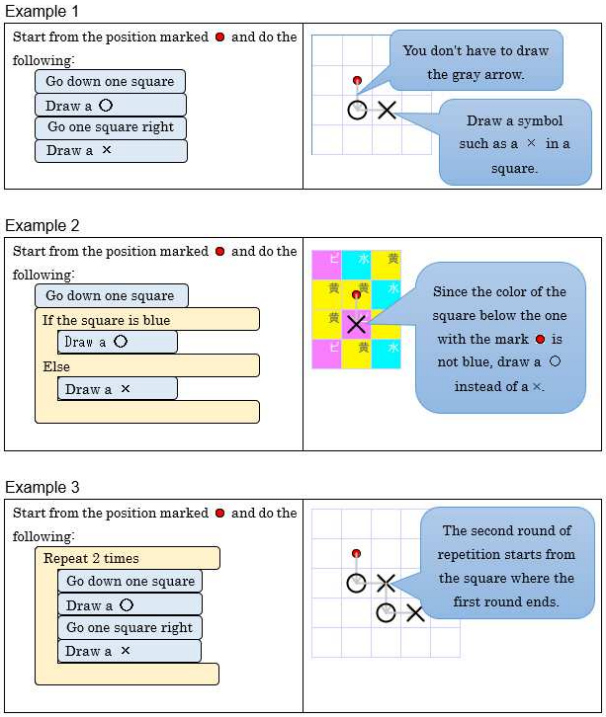

Fig. 4. Exercise questions

C. Distribution of Coding Sheets

The theme of the set of coding sheets in the first lesson was to turn on and off the LEDs in the LED matrix. This set of coding sheets was called "LED lighting Sheets", and contained the following eight sheets: title and contents, "Let's select a block", "Let's make patterns of light", "Let's delete a block", "Turn on and off", "Let's repeat", "If", "Let's repeat forever". An example of a coding sheet in this set is shown in Fig. 2. As can be seen from the titles of the sheets, this set of coding sheets contained the sheets to practice how to place the blocks in the script area, and how to remove them. Also, this set contained the sheets to practice repetition and conditional branch. The instructor gave out one sheet to each pair, and waited for all the pairs to finish the task of the sheet. Since the time to complete a task varies from pair to pair, the instructor instructed members of the pairs that finished early to go and help other pairs. After all the pairs completed a task, the instructor gave out the next sheet.

The theme of the set of coding sheets in the second lesson was to control the gate using the servomotor. This set of coding sheets was called "Gate opening and closing sheets", and contained the following seven sheets: title and contents, "Let's move the bar", "Let's move it automatically", "Let's use the push button", "Sensing things", "Let's use both", "With blinking light". At first, the school children controlled the servomotor directly from the programming environment by changing the number of angles of the servomotor. This allowed the children to see the bar turn. After that, the children made a program to move the bar by arranging blocks that

specify the angle of the servo motor. The children then gradually transformed the program into something more complex and practical. That is, when the button is pressed, the bar becomes vertical, and when an object is detected, the bar becomes horizontal. This mimics the movement of a gate bar of a parking lot which opens when a switch is pressed, and closes automatically when a car gets inside the parking lot. As an option, the students could turn on the LED when the bar was vertical. Table II shows the themes and learning contents of each coding sheet.

D. Post-Test

The ten-minute post-test was conducted after the second lesson was over. The post-test was the same as pre-test, except that the five-minute video was not viewed by the children.

TABLE II. THEMES AND LEARNING CONTENTS OF EACH CODING SHEET

Sheet	Theme	Learning contents
LED lighting sheets		
1	Let's select a block	How to move the selected block from the block palette to the script area
2	Let's make patterns of light	How to manipulate blocks to display light patterns on the LED grid
3	Let's delete a block	How to delete a block from the script area
4	Turn on and off	Sequential processing program for turning on and off the LED
5	Let's repeat	Repetitive program that repeats turning the LED on and off
6	If	A program with if-else syntax that determines the LED color according to the touch sensor value
7	Let's repeat forever	Nesting if-else syntax in repeat-forever syntax so that the LED color changes when the touch sensor is pressed
Gate opening and closing sheets		
1	Let's move the bar	How to manipulate blocks to turn the servo motor
2	Let's move it automatically	Turning servo motor to and fro using sequential program
3	Let's use the push button	How to use touch sensor value in if syntax
4	Sensing things	How to use photo reflector value in if syntax
5	Let's use both	Nesting two if syntaxes in repeat forever syntax
6	With blinking light	Repeat until syntax

V. RESULTS AND INTERPRETATION

Due to the circumstances of Japanese elementary schools, it was not possible to divide the children into experimental groups and control groups for comparison in this experience class. Table III summarizes the results for the difference between the pre-test and post-test means. As can be seen from the table, the increase in the mean value is small, although there is a statistically significant difference according to the paired t-test. Since there is no control group, it is not possible to draw a definitive conclusion.

In order to evaluate the teaching materials and teaching methods as well as the assessment tool itself, each question in the assessment tool was analyzed. The number of correct answers to each question in the pre- and post-test and the results of the Fishers test are shown in Table IV. As can be seen from the table, no questions showed a significant difference between the pre-test and the post-test.

TABLE III. SUMMARY OF THE RESULTS OF PRE-TEST AND POST-TEST

Item	Value
sample size	22
mean of pre-test	5.818
variance of pre-test	3.240
mean of post-test	6.227
variance of post-test	2.812
mean of the differences	0.409
p-value (paired t-test)	0.03554
effect size (Cohen's d)	0.2297715
effect size (Hedges' g)	0.2256439
Cronbach's alpha of pre-test	0.76
Cronbach's alpha of post-test	0.76

Since the Fishers test does not consider whether the same person's answer results changed between the pre-test and the post-test, the McNemar test was also conducted. Table V shows the McNemar test results. In this table, there is a 4 x 4 matrix under each question. Top-left is the number of children who answered the question correctly in both pre-test and post-test. Top-right is the number of children who answered the question correctly in pre-test and incorrectly in post-test. Bottom-left is the number of children who answered the question incorrectly in pre-test and correctly in post-test. Bottom-right is the number of children who answered the question incorrectly in both pre-test and post-test. The McNemar test results in Table V also indicate that no questions showed a significant difference between pre-test and post-test.

TABLE IV. THE NUMBER OF CORRECT ANSWERS AND FISHERS TEST RESULTS

	Question 1		**Question 2**		**Question 3**		**Question 4**	
	Cor	***Inc***	***Cor***	***Inc***	***Cor***	***Inc***	***Cor***	***Inc***
pre	22	0	21	1	18	4	20	2
post	22	0	21	1	18	4	20	2
p-val	1		1		1		1	
	Question 5		**Question 6**		**Question 7**		**Question 8**	
	Cor	***Inc***	***Cor***	***Inc***	***Cor***	***Inc***	***Cor***	***Inc***
pre	17	5	11	11	13	9	6	16
post	19	3	13	9	18	4	6	16
p-val	0.6981		0.7626		0.1854		1	

("Cor": number of children who answered correctly, "Inc": number of children who answered incorrectly)

TABLE V. MCNEMAR TEST RESULTS

	Question 1		**Question 2**		**Question 3**		**Question 4**	
	post cor	***post inc***	***post cor***	***post inc***	***post cor***	***post inc***	***post cor***	***post inc***
pre cor	22	0	21	0	17	1	19	1
pre inc	0	0	0	1	1	3	1	1
p-val	NA		NA		1		1	
	Question 5		**Question 6**		**Question 7**		**Question 8**	
	post cor	***post inc***	***post cor***	***post inc***	***post cor***	***post inc***	***post cor***	***post inc***
pre cor	17	0	10	1	12	1	5	1
pre inc	2	3	3	8	6	3	1	15
p-val	0.4795		0.6171		0.1306		1	

VI. Discussion

A. Teaching Materials

The first factor to be improved regarding the teaching materials is the level of acquisition of CT. Although there is a statistically significant increase in the mean of the test score according to the paired t-test, the difference is slight. Ideally, the improvement in CT ability should be observed more clearly. One of the possible reasons for this is that the teaching materials did not contain sufficiently complex algorithmic syntaxes. This observation comes from the fact that many children answered Question 6 and Question 8 in the assessment tool incorrectly in both pre-test and post-test according to Table V. Question 6 contains three consecutive repetition syntaxes each of which includes two tasks. Question 8 contains two consecutive conditional branches each of which includes repetition syntax. It is possible that the teaching materials did not provide sufficient training to answer these questions to the children.

The second and third factors to be improved regarding the teaching materials are the difficulty level appropriate for the child's age, and the degree of interest of children. The difficulty level and the amount of teaching material was considered to be appropriate since the children just finished the tasks at the end of the second lesson. Also, the robot used by the children and the visual programming environment looked appropriate for the 6th grade children, because the children were enthusiastic on the movement of the robots and kept their motivation throughout the two forty-five minute lessons. The coding sheets seemed to help the children learn programming in small steps. If the materials showing the contents of the whole lesson had been distributed first, the fast-moving pairs would have been advancing on their own. It was observed that children who completed the task of a coding sheet early went to teach children who had not completed the task. In this way, it was suggested that the coding sheet is effective for programming education of elementary school children.

B. Assessment Tools

The first factor to be improved regarding the assessment tool is the difficulty level appropriate for the child's age. According to Table IV, Question 1 (100%), Question 2 (95%), and Question 4 (91%) had more than 90% of correct answers in the pre-test. These questions in the assessment tool were too easy for 6th grade school children. Of course, it is necessary to have a range from easy problems to difficult problems. Otherwise the distribution of the scores will be biased. If questions are too easy, however, this question will not contribute at all for increasing the average score of the assessment from pre-test to post-test. Those questions should be replaced with questions that would give less than 90% percentage of correct answers.

The second factor to be improved regarding the assessment tool is the answer time. As for the assessment time, all the children could answer the eight questions in ten minutes, and not much time was left. Therefore, the answer time of ten minutes was appropriate for the assessment tool with eight questions. Though researchers have little control on the answer time.

The third factor to be improved regarding the assessment tool is reliability / validity. With regard to validity of the assessment, whether the assessment tool presented in this paper really measure CT ability, the criterion validity of the assessment tool has not been studied. This is partially because there is not a standard CT assessment tool that is widely accepted worldwide. Without such a standard CT assessment tool, it is not possible to calculate validity coefficient of the assessment tool presented in this research. Though signs of validity in assessing CT in the context of programming have begun appearing [22][23], it is not common to measure CT ability with the performance of programming lessons. Therefore, it is not easy to evaluate validity of assessment tools of CT ability.

VII. Conclusion

In this research, the authors sought to answer the question of how we can tell if a school child has acquired CT ability. For this, a method for mutually improving teaching materials for programming education in elementary schools and assessment tools for evaluating CT ability was proposed. This method is based on the idea that there is a close relationship between teaching materials for enhancing CT and assessment tools for evaluating computational thinking ability. The target of programming education was 22 sixth graders. The content of the programming education was to program a robot with a servo motor, LED lights, a touch sensor, and a distance sensor using a visual programming environment. As for teaching materials for the programming education, coding sheets were used so that children can learn in small steps. A programming experience class was conducted as two consecutive 45-minute lessons. To evaluate CT ability of the children, pre-test and post-test were conducted using the assessment tool with eight questions. The result was analyzed and found that there was statistically significant improvement in the average score according to the paired t-test, but the increase was small.

To tell if the school children have acquired CT ability, some questions related to components of computational thinking were asked to the children. However, unless the children learn the component of CT, their answers to the questions would not be improved. For the children to learn the components of CT well, they require good teaching materials. For this reason, the authors believe that mutual improvement between teaching materials and assessment tools is important.

As future works, firstly, we would like to add questions about problem decomposition and abstraction to the assessment tool. Secondly, we would like to create assessment tools that can be used in secondary schools and high schools. Thirdly, we would like to make these assessment tools and programming materials using coding sheets available on the web for widespread use.

Acknowledgment

This work was supported by Japan Society for the Promotion of Science KAKENHI Grant Number JP19K02985.

References

[1] D. Barr, J. Harrison, and L. Conery, "Computational thinking: A digital age skill for everyone," in Proc. International Society for Technology in Education (ISTE), 2011, pp. 20-23.

[2] J. M. Wing, "Computational thinking and thinking about computing," Phil. Trans. R. Soc. A, 2008, pp. 3717-3725.

[3] J. Shim, D. Kwon, and W. Lee "The Effects of a Robot Game Environment on Computer Programming Education for Elementary School Students," IEEE Transactions on Education, vol. 60, no. 2, pp. 164-172, 2017.

[4] C. Stephenson, R. Dovi, “More than Gender: Taking a Systemic Approach to Improving K-12 Computer Science Education,” Computer, March 2013, pp. 42-46.

[5] V. J. Shute, C. Sun, J. Asbell-Clarke, “Demystifying computational thinking,” Educational Research Review 22, 2017, pp. 142-158.

[6] I. Storjak, L. Pushkar, T. Jagust, A. S. Krzic, “First steps into STEM for young pupils through informal workshops,” in Proc. IEEE FIE 2020, 2020.

[7] K. T. Fernandes, E. H. S. Aranha, M. J. N. R. Lucena, G. L. S. Fernandes, “Developing Computational Thinking and Reading and Writing Skills through an Approach for Creating Games,” in Proc. IEEE FIE 2020, 2020.

[8] J. M. Wing, “Computational thinking,” Communications of the ACM, vol. 49, no. 3, pp. 33-35, 2006.

[9] V. Barr and C. Stephenson, “Bringing Computational Thinking to k-12: What is involved and what is the role of the computer science education community?,” ACM Inroads, vol. 2, no. 1, pp. 48-54, 2011.

[10] The International Society for Technology in Education and the Computer Science Teachers Association, “Operational Definition of Computational Thinking for K-12 Education,” 2011. https://cdn.iste.org/www-root/ct-documents/computational-thinking-operational-definition-flyer.pdf

[11] I. Ioannou and C. Angeli, “A framework and an instructional design model for the development of students’ Computational and Algorithmic Thinking,” in Proc. MCIS 2016, 2016.

[12] B. Fanny, H. Julie, C. Anne-Sophie, “Developing a Critical Robot Literacy for Young People from Conceptual Metaphors Analysis,” in Proc. IEEE FIE 2020, 2020.

[13] M. R. González, J. P. González, C. J. Fernández, “Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test,” Computers in Human Behavior 72, 2017, pp. 678-691.

[14] S. Grover, S. Cooper, R. Pea, “Assessing computational learning in K-12,” Proceedings of ACM ITICSE’14, 2014, pp. 57-62.

[15] O. Oliveira M. Nicoletti, L. Cura, “Quantitative correlation between ability to compute and student performance in a primary school,” in Proc. ACM SIGCSE’14, 2014, pp. 505-510.

[16] L. Werner, J. Denner, and S. Campe, “The Fairy Performance Assessment: Measuring Computational Thinking in Middle School,” in Proc ACM SIGCSE’12, 2012, pp. 215-220.

[17] K. H. Koh, A. Basawapatna, H. Nickerson, A. Repenning, “Real Time Assessment of Computational Thinking,”in Proc. IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), 2014, pp. 49-52.

[18] A. L. S. O. Araujo, J. S. Santos, W. L. Andrade, D. D. S. Guerrero, V. Dagiene, “Exploring Computational Thinking Assessment in Introductory Programming Courses,” in Proc. IEEE FIE 2017, 2017, pp. 1-9.

[19] S. Serth, R. Teusner, J. Renz, M. Uflacker, “Evaluating Digital Worksheets with Interactive Programming Exercises for K-12 Education,” in Proc. IEEE FIE 2019, 2019.

[20] ArtecRobo2.0 | Artec Co., Ltd. Accessed: Apr. 29, 2021. [Online]. Available: https://www.artec-kk.co.jp/artecrobo2/en/

[21] F. Kalelioğlu, Y. Gülbahar, and O. Madran, “A snapshot of the first implementation of Bebras international informatics contest in Turkey,” in A. Brodnik, & J. Vahrenhold (Eds.), Informatics in schools. Curricula, competences, and competitions, 2015, pp.131-140.

[22] J. Fagerlund, P. Hakkinen, M. Vesisenaho, J. Viiri, “Computational thinking in programming with Scratch in primary schools: A systematic review,” Computer Applications in Engineering Education, vol. 29, no. 1, pp. 12-28, 2021.

[23] M. R. González, J. M. León and G. Robles, “Combining assessment tools for a comprehensive evaluation of computational thinking interventions,” Comput. Thinking Edu. (S. C. Kong and H. Abelson, eds.), Springer, Singapore, 2019, pp. 79-98.