Evaluation of the ROOT Robot System and Curriculum to Improve Computational Thinking in Chinese Children

Yanting Liu¹, Juan Rojas∗

Abstract—Computational thinking (CT) is defined as the thought process of formulating problems and expressing their solutions. In recent times, children increasingly program and CT is a fundamental skill to master. Research shows children often lack such skill. In this work, we contribute an analysis of the ROOT robot system and its associated curriculum to aid in the development of CT skills in children aged 4-10 years old (in China). In particular, we measure the concepts of abstraction, analysis, generalization, algorithmic thinking, logical thinking, sequential problem solving, self-confidence, and self-checking capability throughout a 10-week course. The results show 71% students developed “Excellent” CT performance and 29% achieved “Good” performance. The key lesson is that children’s CT skills develop positively when programming with the ROOT learning platform.

Index Terms—educational robot, computational thinking, ROOT robot, programming, evaluation, measurement

I. INTRODUCTION

Today children are increasingly exposed to programming from a young age. Equipping children with the understanding of the underlying behaviors of such systems is critical to prepare them to respond to the challenges and opportunities available through technology. Such premise is defined as Computational Thinking (CT) [1, 2]. CT is the formulation of problems and associated solutions through computational steps and algorithms. Despite the importance of CT skills; today many children still lack training in CT [2]. In China, educational robotics has received unprecedented attention this decade. The Chinese ministry of education proposed to enrich the content of artificial intelligence and programming courses to meet the challenges brought forth by the AI generation [3].

In recent years, numerous research efforts have conveyed how robotic education can promote CT skills in students of all ages [4]. In [5], researchers designed kindergarten-appropriate computer programming and robotics tools with a constructionist curriculum. Children learned CT skills, robotics, programming, and problem-solving. The research just shows children gained interest and proficiency in various aspects of robotics, programming, and CT skills, but does not refer whether students’ CT has developed. In [6], the paper assessed the impact of robotics on 15 & 18 year-old students’ CT skills. It shows the development of CT skills on older than 15 year-old students, but don’t mention younger children. Others have also identified side-or-second-order—benefits, such as improvement in learning and understanding in the fields of math and science [7]. Since Wing et al. [8], stated that CT will be a fundamental skill used by everyone in the world by the middle of the 21st Century, and we should add CT to every children’s analytic ability, there were many literature about CT skills. In [9], CT was studied in K-12, and referred that CT is a problem solving methodology that can be automated and transferred and applied across subjects, students become not merely tool users but tool builders to use a set of concepts, such as abstraction, recursion, and iteration to process and analyze data, and to create real and virtual artifacts. In [11], CT has been described as the use of abstraction, automation, and analysis in problem-solving. [12] pointed out that implementing CT during the school day is a compelling vision, but there are substantial challenges to this, including existing curriculum standards lack of opportunities for teachers to learn CT as part of their professional development, and lack of access to necessary infrastructure. Various studies focus on slightly different concepts of CT. However, many paper about CT just repeatedly discussed the importance and definition of CT concepts, without actually investigating CT development. Most literature has focused on English speaking students in developed nations. In this paper, we set out to study CT skill development in Chinese children and will question whether the same learning patterns will emerge. To this end, this work evaluated CT skill developed through an after-school program. We evaluate five key CT concepts: abstraction, analysis, generalization, logical thinking and algorithmic thinking. Table I shows the concepts of CT mentioned by literature. In this work, we

1All authors are with the School of Electro-Mechanical Engineering, Guangdong University of Technology, Guangzhou, China, 51006.

study whether the use of the novel ROOT learning program

Fig. 1: The Root robot system consists of a cohesive environment between its mobile and its custom designed iOS app, and a curriculum.
TABLE I: The CT skills in various literature

<table>
<thead>
<tr>
<th>Article</th>
<th>Age</th>
<th>CT Concepts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Varible Barrs et al. [9]</td>
<td>k-12</td>
<td>decomposition, abstraction, negotiation, consensus building</td>
</tr>
<tr>
<td>Irene Lee et al. [15]</td>
<td>k-12</td>
<td>abstraction, automation, analysis</td>
</tr>
<tr>
<td>Soumela Atmatzidou et al. [6]</td>
<td>15~18</td>
<td>abstraction, generalization, algorithm, modularity, decomposition</td>
</tr>
<tr>
<td>Karen Brenman et al. [16]</td>
<td>8~17</td>
<td>concepts, practices, perspectives</td>
</tr>
<tr>
<td>Marina Umaschi Bers et al. [5]</td>
<td>4.9~6.5</td>
<td>debugging, action &amp; instruction correspondences, instruction sequencing</td>
</tr>
<tr>
<td>Elizabeth R. Kazakoff et al. [10]</td>
<td>4.7</td>
<td>sequence</td>
</tr>
<tr>
<td>This work</td>
<td>4~10</td>
<td>abstraction, analysis, generalization, negotiation, algorithmic thinking</td>
</tr>
</tbody>
</table>

(ROOT) can offer significant gains in CT skills for young children in a distinct Chinese background. The ROOT system (see Fig. 1) offers a uniquely cohesive environment between the hardware and the software [13]. The IDE is carefully designed to facilitate programming for people across all ages: from block programming, to parametric programming, to swift programming. The robot is a small, two-wheeled mobile robot equipped with a rich sensor suite as shown in Fig. 1. The ROOT robot is programmable through an iOS APP [8]. The app consists of 3 modes: (i) an instructional mode; (ii) a coding interface; and (iii) an open interface where users can post or use publicly available projects. The curriculum is constructionist and scaffold as well. Our contribution is an evaluation of CT skills in children aged 4-10 years old through ROOT.

We implemented a 10-week after school program using ROOT. The goal is to measure the impact the system has on the development of student’s CT skills. 15 students were divided into two classes according to their age. CT skills are measured as a function of abstraction, analysis, generalization, algorithmic thinking, logical thinking, sequential problem solving, self-confidence, and self-checking capability. The development of students CT skills was evaluated during the class. To measure these, we employed likert-scale based questionnaires, think-aloud protocols, and pre- & post-test.

Children’s feedback indicated that robotics was a motivating tool to learn CT concepts compared with just working with screen animations (i.e. Code Monkey [14]). Students without a programming background were able to learn at the same speed as those with a programming background at the beginning when simpler concepts were present. However, students with a programming background excelled in understanding more complex concepts. Learning ability and gender showed no correlation. The results show 71% students had “Excellent” CT performance and 29% had “Good” performance. Fundamentally, we see that children develop positively when they learn programming with the ROOT robot learning platform.

II. METHODOLOGY

A. The ROOT robot system

We chose the ROOT robot as the teaching and learning tool, and our evaluation is based on this robot and curriculum. The robot is a small, two-wheeled mobile robot equipped with a rich sensor suite. Children learn of ROOT as a robot that can move and turn, climb whiteboards surfaces (thanks to its magnetic surface); draw (with built-in functionality to raise and lower markers); detect colors and edges on its bottom; see light on its top, feel human touch (and thus trigger customizable events), bumps, and inertial motion; and hear sounds. The ROOT robot is programmable through an App designed for Apple’s iPad tablet [15].

Three different programming interfaces are available in the app to facilitate learning for all ages: Graphical coding (ages 4+): Graphical blocks teach essential logic skills of coding; Parameter coding (ages 7+): Hybrid drag and drop language for building computational fluency; and Full text coding (ages 12+): Learn the structure and syntax of professional coding languages in Swift. As shown in Fig. 2. At the core of the system is the instructional mode. The system is designed to provide students with only small chunks of concepts at a time. It follows a constructionist approach which allows children to learn through hands-on activities. More than 50 lessons are scaffold such that learning difficulty increases sequentially. Content is presented under multiple media representations supporting different learning preferences. Through multilevel programming, difficulty is partitioned across three levels:

1) Level 1 imparts young children coding concepts including: the understanding of what coding is, functions and parameters, events, event sequencing and handling, conditional statements, loops, and sensors and actuators. ROOT also promotes skills such as counting techniques, basic robotics, visual spatial navigation, logic, problem-solving skills, design thinking, and creativity.

2) Level 2 imparts coding concepts including: variables, sensor values, algorithms, get/set operations, operators, randomization, lists, parallelism, and booleans. This level promotes skills including: iterative design, modularity, communication and collaboration, mathematical reasoning, the generation and analysis of patterns, and data collection and analysis.

3) Level 3 trains students on full text coding through Swift as well as advanced syntax and flow Control.

B. Participants

For this work, we worked with two separate classes. Parents were informed of our class through social media. Classes took place as part of an after-school program. A total of 15 children participated. The class 1 was comprised of 9 primary school students (8 male and 1 female), their ages ranged from 7-10. Children were typically set to learned
practice in pairs. 5 students of his class had had prior training in basic programming (i.e. Code Monkey [14]), 4 students had not. The second class consisted of 6 primary school children with ages ranging from 4-6.5 years old. The group consisted of 4 boys and 2 girls. These children had no prior programming skills.

The class teachers were master students with majors in computer science and robotics. And they had a good understanding of CT skills. Their age range was in their 20s and had a strong command of both Chinese and English. Teachers managed the classroom logistics and hardware, provided guidance through the class, taught basic knowledge concepts, and collected feedback and data throughout the class.

C. Curriculum

Content of the curriculum was organized as part of a 10 to 12 week courses: 1 to 1.5 hour class per week. Students were generally paired up and provided with a ROOT robot, an iPad, one portable whiteboard. At times, for finishing some simple tasks, all students worked individually with their own robot system. A typical class would first introduce students to the equipment and concepts necessary for the module at hand. The teacher would demonstrate the function of each coding concept and let students practice with their teacher. Thereafter, students were given tasks that were to be completed independently under a specified amount of time. Typically such tasks could be solved through a variety of different ways. Robot competitions were used at the end of a lesson, where several coding concepts could be galvanized together, and each group recommends one student to explain the ideas for the final comprehensive task to teacher and other classmates. Projects gave children an opportunity to share their solutions with each other. The content of Root robot curriculum was shown in Table II.

D. Evaluation and Measurement

In order to measure progress in CT skills which include five key concepts: abstraction, analysis, generalization, algorithmic thinking and logical thinking. A variety of data was collected from the students through the class (modules and projects): project completion time, task completion, degree of task completion, the children’s enthusiasm during a module, and whether or not the children could understand the relevant coding concepts to reflect students’ development of CT skills.

1) Measurements: This work tried to develop a suitable measurement system to measure students’ CT skills by using ROOT robot. To this end, we reviewed a comprehensive list of metrics as shown in Table III. The detailed content of the metrics in literature is shown as following:

TABLE II: The content of Root robot curriculum are arranged in 10 weeks (1.5 hours per week), each lesson present different knowledge points, and the tasks gradually increase in difficulty

<table>
<thead>
<tr>
<th>No</th>
<th>Topic</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Robot Intro</td>
<td>robot, system, coding</td>
</tr>
<tr>
<td>2</td>
<td>Robot movement</td>
<td>simple sidewalk</td>
</tr>
<tr>
<td>3</td>
<td>Light &amp; Music</td>
<td>cross obstacle &amp; dance floor</td>
</tr>
<tr>
<td>4</td>
<td>Touch sensors</td>
<td>touch interaction</td>
</tr>
<tr>
<td>5</td>
<td>Bump sensors</td>
<td>build touch-bot game board</td>
</tr>
<tr>
<td>6</td>
<td>Robot drawing</td>
<td>draw simple shapes &amp; initials</td>
</tr>
<tr>
<td>7</td>
<td>Boomerang</td>
<td>shape wheel</td>
</tr>
<tr>
<td>8</td>
<td>Wheel speed</td>
<td>draw a face &amp; wiggly Root</td>
</tr>
<tr>
<td>9</td>
<td>Color sensing</td>
<td>line follower &amp; color guitar</td>
</tr>
<tr>
<td>10</td>
<td>Sensing &amp; Responding</td>
<td>voice &amp; light controlled vehicle</td>
</tr>
</tbody>
</table>

TABLE III: A comprehensive list of metrics: (a) Profile questionnaires, (b) Intermediate questionnaires, (c) Student opinion questionnaires, (d) Think-aloud protocols, (e) Observations, (f) Project portfolio analysis, (g) likert-scale, and (h) Pre- & post-test.

<table>
<thead>
<tr>
<th>Paper Age</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atmatzidou et al. [6]</td>
<td>15~18</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Brennan et al. [4]</td>
<td>8~17</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bers et al. [5]</td>
<td>4.9~6.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Kazakoff et al. [10]</td>
<td>4~7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>
demographic data (e.g. student gender, age), the students’ background on programming and experience with robotics.

2) Intermediate questionnaires are carried out in the middle of the lesson, students will be asked to solve programming problems and practice CT skills during their solution process.

3) Student opinion questionnaire needs to be filled individually by students after the training to explore students’ development of CT skills, understanding of basic programming concepts, students’ subjective feelings of learning harvest and so on.

4) Think-aloud protocol refers that students will be given a certain robot programming task and will be asked to describe aloud the process they would follow to solve it. [6]

5) Observation refers that systematic monitoring of the students’ work is applied by taking notes on a structured form (observation sheets). [6] used likert-scale to record the completion of tasks to reflect the development of students’ CT skills

6) Likert-scale is a rating scale, for example, it ranges from 0 (did not attempt the task) or 1 (did not complete the goal, task, or understanding) to 5 (completely achieved the goal, task, or understanding). In [5], it evaluated the development of CT by executing tests.

7) Pre- & Post-test, one is executed on students before the course, another is executed after the course. [10]

In [6], the research used three different questionnaires, Think-aloud protocol, and observation to evaluated the advancement of students’ CT skills. Combined with the measurement experience of other researchers and the characteristics of this course, we finally adopt the following measurement methods:

(a) Questionnaires: used to collect the information of students and learning feedback from students.

(b) Think-aloud protocol: used to improve the expression and confidence of students, and to show students’ mastery of programming knowledge and CT skills.

(c) Observation: used in the process of students completing tasks to collect the quality and completion time of the task.

(d) Pre- & Post-test: check the development of students programming skills and CT.

(e) Likert-scale: used to record the scores of various measurement methods by teachers.

The results of measurement reflect the development of students CT skills focus on abstraction, generalization, algorithmic thinking, analysis, logical thinking. The concepts of these CT skills are shown in Table IV. The intuitive metrics measured by us were:

(a) Course: the difficulty of the tasks, number of lessons, the time of each lesson;

(b) Participants: the number of students, the number of the teachers, the background of students (e.g. age, grade, programming or robotics basic learning), the sex ratio, the grouping of students;

(c) Learning outcomes: the time of completing each task, the quality of completing the task, the level of mastery of programming knowledge.

2) Procedure: Every semester included 10 to 12 lessons. Each lesson was divided into three sessions. Session 1 was review, which lasted 10 to 15 minutes. At session 1, teacher would help students review learning content of last class, and give students a small task; Session 2 was learning

Fig. 3: Course Structure: each lesson has three sessions, the three measurement methods are arranged at different times. Likert-scale records students’ CT skill performance measures.
concept and completing small tasks, which lasted 30 minutes. In session 2, teacher imparted new knowledge and new small tasks to the students; Session 3 was that group of students completing a comprehensive task to compete with each other, which lasted 15 to 20 minutes. During session 3, teacher would give students a comprehensive task and ask students show their tasks to other students, then teacher guided students summarize what they have learned in this lesson.

Questionnaire 1 was conducted at the beginning of a new semester to collect students’ information (e.g. students’ number, ages, grade, sex ratio, background of programming or robotics basic learning ). We collected these information of students through oral questionnaire.

Questionnaire 2 was conducted at the end of session 1 to collect students’ learning effect of last lesson.

Questionnaire 3 was conducted at the end of session 3 to collect students’ learning effect of this new lesson. At the same time, teacher would record students’ completion time and observe results of session 1 task and session 2 task. (we used a likert-scale to record the scores of each students’ tasks) In each lesson, we conducted pre- & post-test, one at the end of the session 1, another at the end of the session 3. In this way, the result of pre- & post-test could be compared with each other to evaluate the development of the CT skills. The course structure is shown in Fig. 3.

1) Questionnaire scoring criteria: the options in our questionnaire are composed of patterns representing different degrees. For example, with three smiley faces of varying degrees, when the child chooses different answers, we give the child a score between minus 1, 0, and 1 for CT skills.

2) Time recording scoring criteria: all evaluation metrics of time recording are assigned the following labels for given scores: 10’ Less than 50% of the allotted time(t); 6’: 50%≤t≤70%, 2’: 70%≤t≤100% , and -2’: t≥100%.

3) Tasks’ quality scoring criteria: all evaluation metrics of quality are assigned the following labels for given scores: 10’ within the allotted time and no mistakes; 5’ within the allotted time and small mistakes which can be debugged by themselves; -2’ students can not finish the task by themselves.

4) Evaluation criteria of results: after pre- & post-test, we will calculate the total score of students’ CT skills in each class and give an evaluation. We’re going to represent the 10 weeks lessons as 10 tasks. All of our evaluation metrics are assigned the following labels for given scores: 14’ to 21’ “Excellent”, 7’ to 14’ “Good”, 0’ to 7’ “Average”, -5’ to 0’ “Poor”.

III. RESULTS

After a period of data collection, the data results are shown as follows:

1) The questionnaire 1 showed the background information of students. It represented that 5 students in Class 1 had basic programming background(Code Monkey [14]), 4 students did not, they were 7 to 10 year-old student; the students in Class 2 had no basic programming background, they were 4 to 6.5 years old.

2) The questionnaire 2 was conducted after session 1, it consisted of three questions. The questions were related to learning feedback of students (e.g. the level of knowledge understanding, the difficulty of tasks and knowledge concepts). The options were three different degrees of smiley faces. Through the results collected from these questionnaires, teachers could timely adjust the difficulty and speed of classes to adapt to the learning efficiency of students.

3) Pre-test was carried out at the end of session 1, and post-test was conducted during session 3. Students were asked to complete two equally difficult tasks in a given time which were related to program the Root robot. At the end of session 3, group of students would finish a comprehensive task, which was more difficult than the small tasks.

4) Observation and likert-scale were conducted when students were completing the tasks, comprehensive task and think-aloud protocol to record the task completion time, task completion quality, student’s knowledge presentation and collaboration with classmates.

After 10-week programming lesson by using ROOT robot, the development of child’s CT skills are shown in Fig. 4. In Class 1, 5 students had basic programming background(Code Monkey [14]), 4 students had no. 7 students achieved "Excellent", and 2 students achieved "Good". In Class 2, the students were younger, 3 students gained "Excellent" and 3 students gained "Good". Before the course, child’s CT skills were hovering at "Average". After the class, 71% students showed "Excellent" performance of CT skills, and 29% students showed "Good" performance. All girls showed "Excellent" performance in CT skills. Older students performed higher "Excellent" performance rate in CT skills.

IV. CONCLUSION

Children feedback indicated that Chinese child’s CT skills were improved 1 or 2 levels through ROOT robot. Children between 7-10 years old, had better CT skills for more difficult problems. Younger children (4-6 years old), showed good performance under guidance for difficult tasks. A nice characteristic of ROOT robot learning system is that students can directly control a physical robot through programming, which attracted most students’ attention. According to students with Code Monkey [14] background, educational robotics is a motivating tool to learn computational concepts compared with screen animations. Physical robots are intrinsically motivating creating an emotional connection between them and the children. Children showed strong motivation to complete tasks better than compared to screen animations. Students without a programming background were able to learn at the same speed as those with a programming background during the initial more simple concepts, but students with a programming background excelled in
Fig. 4: CT Skill Performance Measurements: A set of 10 comprehensive tasks were measured according to metrics in Sec.II-D by likert-scale. On the left, results for Class 1 and on the right results for Class 2 in sec.III. Students performed "Average" CT skills (see scoring criteria in sec.II-D) before starting curriculum, but their CT skills developed into "Excellent" or "Good" after curriculum. 78% Students in Class 1 (7 to 10 years old) showed "Excellent" CT performance, 22% students performed "Good". 50% students in Class 2 (4 to 6 years old) showed "Excellent" CT performance, 50% students performed "Good". Students’ CT skills of both classes were developing in a positive direction.

understanding more complex concepts, but their learning ability and gender showed no correlation on the improvement of programming skills and development of CT. However, long term differences in learning seem to be caused by the interests of the individual student. On the whole, the average development of young students’(4 to 10 years old) computational thinking (CT) skills are "Good" or "Excellent", their CT skills(abstraction, analysis, generalization, algorithmic thinking, logical thinking, sequential problem solving, self-confidence, and self-checking capability) tend to developing in a positive direction when they learn programming with the ROOT robot.

REFERENCES