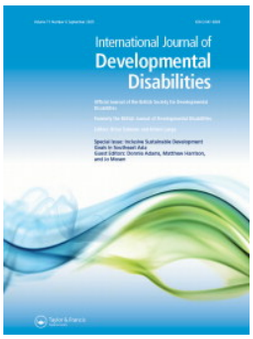


International Journal of Developmental Disabilities

ISSN: 2047-3869 (Print) 2047-3877 (Online) Journal homepage: www.tandfonline.com/journals/yjdd20

Enhancing computational thinking in students with autism spectrum disorder and intellectual disabilities: a robot programming approach

MiJeong Kim, JaMee Kim & WonGyu Lee

To cite this article: MiJeong Kim, JaMee Kim & WonGyu Lee (23 Aug 2024): Enhancing computational thinking in students with autism spectrum disorder and intellectual disabilities: a robot programming approach, International Journal of Developmental Disabilities, DOI: 10.1080/20473869.2024.2394735

To link to this article: https://doi.org/10.1080/20473869.2024.2394735

INTERNATIONAL JOURNAL OF DEVELOPMENTAL DISABILITIES
https://doi.org/10.1080/20473869.2024.2394735

Enhancing computational thinking in students with autism spectrum disorder and intellectual disabilities: a robot programming approach

MiJeong Kim[a], JaMee Kim[b] and WonGyu Lee[a]

[a]Department of Computer Science and Engineering, Graduate School, Korea University, Seoul, South Korea; [b]Major of Computer Science Education, Graduate School of Education, Seoul, South Korea

ABSTRACT

Objectives: Research examining computational thinking skills development *via* programming education using robotic teaching tools for students with Autism Spectrum Disorder and Intellectual Disabilities remains limited. This study aims to analyze the advancements in computational thinking skills among middle school students with Autism Spectrum Disorder and Intellectual Disabilities through participation in robot programming classes.

Methods: Problem-solving proficiency in computational thinking was assessed utilizing Brennan and Resnick's three dimensions framework. A tailored teaching and learning approach was implemented for robot programming education, accounting for participants' cognitive abilities and required levels of support. Four middle school students, comprising one with Autism Spectrum Disorder and three with Intellectual Disabilities, were enrolled.

Results: Significant enhancements were observed in all participants' computational thinking skills across conceptual understanding, practical application, and perspective. Notably, all participants successfully resolved the final project problem at their individual proficiency levels. The acquisition of computational concepts was found to correlate positively with improved practical computational skills. Additionally, all participants demonstrated keen interest in and positive attitudes toward both robots and programming.

Conclusions: This study underscores the efficacy of robotic teaching tools in enhancing computational thinking skills through programming education. Additionally, it highlights considerations for enhancing students' accessibility in future programming class design.

ARTICLE HISTORY
Received 23 February 2024
Accepted 30 July 2024

KEYWORDS
Robot; programming; computational thinking; Autism spectrum disorder; intellectual disability

Introduction

Computer science (CS) is defined as the study of computers and algorithmic processes, including their principles, hardware and software designs, applications, and impact on society (Tucker 2003). This means that CS is not synonymous with programming or computer literacy. Papert's work in 1980 and Bell's work in 2002 are considered the seminal works that shaped CS education (Bell, Witten, and Fellows 1998; Papert 1980). Papert's logo introduced algorithmic thinking. What is common between Papert's logo and Bell's hands-on experience is that they both emphasize students' direct problem-solving experience.

Recently, there has been a burgeoning interest in enhancing computational thinking (CT) skills among K-12 students through CS education (Luo, Israel, and Gane 2022). Wing (2006) initially proposed the concept of CT as a novel problem-solving strategy applicable across virtual and real-world domains, advocating for the cultivation of CT akin to computer scientists. Subsequent research has popularized the integration of CT principles into CS education across various fields and educational levels, leading to the formulation of curriculum standards aimed at teaching CT to K-12 students (Pollak and Ebner 2019; Yadav, Stephenson, and Hong 2017). Notably, certain CS concepts such as algorithms, composition, patterns, abstraction, and debugging are deemed pertinent to K-12 education (Grover and Pea 2013; Yadav, Hong, and Stephenson 2016). In addition to the Computer Science Teachers Association (CSTA), organizations that have developed their own framework for introducing CT to schools include the Computing At School (CAS) of the British Computing Society and the Australian Curriculum,

CONTACT WonGyu Lee lwg.koreauniv@gmail.com Department of Computer Science and Engineering, Graduate School, Korea University, 145 Anam-ro, Seongbuk-gu, Seoul, South Korea

Supplemental data for this article can be accessed online at https://doi.org/10.1080/20473869.2024.2394735.

Assessment and Reporting Authority (ACARA) (Tedre and Denning 2016).

The K–12 Computer Science Framework, established by the CSTA, underscores the importance of introducing fundamental CS concepts to all K-12 students, with the goal of fostering computer-literate individuals capable of both consuming and creating technology proficiently (K-12 Computer Science Framework Steering Committee 2016). This CT-based educational approach emphasizes programming, constructivism, self-efficacy, motivation, and teacher training, aiming to provide students with diverse and sophisticated CS experiences throughout their academic journey (González-González et al. 2021; Saqr et al. 2021). Consequently, expanding CS education opportunities and enhancing accessibility for students with disabilities, who may require specialized support, have emerged as crucial imperatives. However, there remains a dearth of research on CT among students with disabilities, particularly those with Autism Spectrum Disorder (ASD) or Intellectual Disability (ID), who often encounter various developmental challenges affecting language acquisition, social interaction, and behavioral regulation. It has been reported that constructivist approaches are not considered best practices when teaching programming to students with ID (Taylor 2018). Previous research has successfully taught programming to students with ASD or ID using specialized educational approaches such as model-lead-test (MLT) instruction and evidence-based practice (EBP) of explicit instruction, with reported positive effects (Knight, Wright, and DeFreese 2019; Taylor 2018). However, further research is needed to explore additional educational strategies for CS education among students with developmental disabilities.

Contemporary studies explore a myriad of approaches to programming education for children and novice programmers, including video games, apps, animations, visual block-based programming languages, robots, and unplugged activities (Statter and Armoni 2020). Robots, in particular, have gained prominence as teaching aids in both educational and therapeutic settings owing to their accessibility and ease of use for individuals with ASD and ID (Conti et al. 2021; Díaz-Boladeras, Claver I Díaz, and García-Sánchez 2023; Desideri et al. 2018; Santos et al. 2023; Syriopoulou-Delli and Gkiolnta 2022). Consequently, the role of robots in enhancing the education and quality of life of individuals with ASD or ID is expanding, underscoring the significance of research on CT skills among this population using robotic tools.

Therefore, this study aims to investigate the improvement of CT skills through robot programming education designed as a specialized approach for students with ASD and ID.

Literature review

Computational thinking research

Various scholars have modified and improved the terminology and meaning of CT. However, there is currently no consensus on the components that should be included in the definition of CT (Kang et al. 2023). A general statement about the definition of CT that many scholars agree on is as follows: CT is 'the thought processes involved in formulating a problem and expressing its solution(s) in such a way that a computer-human or machine can effectively carry out' (Wing 2014).

Until now, studies with various perspectives on CT have been conducted. Representative CT studies include (1) a study that defines CT as the use of abstraction, automation and analysis in problem-solving (Lee et al. 2011), (2) CT concepts and capabilities in Data collection, Data analysis, Data representation, Problem Decomposition, Abstraction, Algorithms & procedures, Automation, Parallelization, and Simulation, a study that presents examples of methods to be integrated into activities in the fields of mathematics, science, social studies, and language arts (Barr and Stephenson 2011), (3) a study that identifying the CT framework three dimensions (computational concepts, computational practices, and computational perspectives) through the learning and development process that occurs in Scratch programming activities (Brennan and Resnick 2012).

Statter and Armoni (2020) classified five major approaches to teaching CT to K-12 students. In other words, they are robot activities, kinesthetic activities (e.g. CS unplugged), games and puzzles (e.g. CodeMonkey, CodeCombat), simple tutorials (e.g. code.org), visual block-based programming environments (e.g. Scratch, Alice, LaPlaya). Moreover, diverse methods for evaluating CT exist, and CT assessment remains a challenging endeavor (Fronza, Ioini, and Corral 2017). Traditionally, evaluation has been conducted through means such as multiple-choice tests, project analysis, and specifically designed tasks for assessment, primarily focusing on gauging students' grasp of computing concepts. However, contemporary perspectives suggest that the assessment should encompass a blend of multiple data sources to yield a more comprehensive insight into CT development

(Grover et al. 2017; Kutay and Oner 2022; Luo, Israel, and Gane 2022; Merkouris and Chorianopoulos 2019; Statter and Armoni 2020). Hence, it is imperative to assess the CT abilities of students with disabilities comprehensively, utilizing a diverse array of data sources akin to their nondisabled counterparts.

In this regard, the framework proposed by Brennan and Resnick proves particularly informative. Their assessment framework delineates three dimensions encompassing computational concepts, computational practices, and computational perceptions. Employing various methodologies including project evaluation, artifact-based interviews, and scenario analysis, they offer a nuanced understanding of CT development (Statter and Armoni 2020). Additionally, CT research should delve into students' attitudes as CT can influence perceptions regarding careers in computing (Fronza, Ioini, and Corral 2017; Grover and Pea 2013). Computational perceptions, however, are not adequately captured within the conventional framework for evaluating computational concepts and performance. Notwithstanding these limitations, studies investigating the computational perspective have synthesized and analyzed qualitative data from interviews alongside questionnaires (Merkouris and Chorianopoulos 2019).

Research on educational robots for students with ASD or ID

Over the past ten years, studies using robots targeting students with ASD or ID have reported positive effects in participants' educational, cognitive, and social domains (González-González et al. 2019; Knight, Wright, and DeFreese 2019; Kolne, Bui, and Lindsay 2021; Lindsay and Hounsell 2017; Lindsay and Lam 2018; Perez et al. 2021; So et al. 2018; Taylor 2018).

Prior research can be summarized as follows. First, various effects of the robot can be confirmed. Rich interfaces for interacting with robots (e.g. touch, voice, hand, full-body gestures, etc.) are more than just educational tools, but also assist with limitations due to disabilities. Taking advantage of the physical interaction aspect, which is one of the biggest advantages of robots, can help increase interest and participation in programming among students with ASD and ID, and ultimately improve CT. Second, a majority of studies targeted children and elementary school age groups (seven studies). In other words, there is a relative lack of research targeting middle school and high school age groups. Third, the most used robot was the LEGO Mindstorms commercial kit (e.g. WeDo, Mindstorms EV3) (three studies). Dash, Ozobot, Arduino UNO-based social robot, KIBO robot, and NAO robot were one study. LEGO teaching tools are relatively inexpensive and efficient. Benefits similar to those seen by using high-cost robots instead of high-cost robotic systems costing $12,000–$30,000 can be achieved (Adams and Cook 2014). Fourth, research on programming education using robots sought to employ educational strategies suitable for students with developmental disabilities. Alongside EBP such as explicit instruction MLT approaches, various educational activities incorporating mathematics, language skills, emotional development, and social skills were integrated. Fifth, only one study discussed CT skills. An existing study discussing CT used a KIBO robot targeting students with Down syndrome to analyze participants' emotional states, engagement, and comprehension of the programming sequences. Research has shown that even people with cognitive impairments can acquire basic programming skills using robots. However, further research discussing improving CT skills through programming education is needed.

This study aimed to analyze improvements in CT through LEGO robot programming education, designed specifically for middle school students with ASD or ID. To this end, we sought an in-depth understanding of the development of CT competencies through individualized teaching and learning strategies tailored to the cognitive abilities and support needs of middle school students with ASD and ID. The evaluation of problem-solving using CT utilized Brennan and Resnick's three dimensions of CT, commonly used in current research. The research questions are as follows:

1. What is the impact of LEGO robot programming classes using individualized teaching and learning strategies on computational concepts among three dimensions of computational thinking in middle school students with ASD and ID?
2. What is the impact of LEGO robot programming classes using individualized teaching and learning strategies on computational practices among three dimensions of computational thinking in middle school students with ASD and ID?
3. What is the impact of LEGO robot programming classes using individualized teaching and learning strategies on computational perspectives among three dimensions of computational thinking in middle school students with ASD and ID?

Methods

Participants

To fulfill the research objectives, this study employed individualized teaching and learning strategies tailored to accommodate students with special learning needs and support (Zigmond and Kloo 2017). To enhance research efficiency, students were selected from a middle school in South Korea where the first author holds responsibility, utilizing convenience sampling. The specific selection criteria were as follows: (1) Students currently enrolled in middle school diagnosed with ASD or ID by specialized educational institutions. (2) To mitigate age-related confounders, students within the same grade level were chosen, ensuring minimal divergence in chronological age, with a maximum difference of 12 months. (3) Students with no prior exposure to robot programming education. (4) Students possessing the cognitive capacity to grasp computational concepts with assistance from the teacher, either verbally or physically. (5) Students capable of consistently participating in the educational sessions mandated by the program. (6) Students devoid of motor impairments hindering their ability to operate robots and tablets effectively. (7) Students capable of expressing their thoughts or emotions either verbally or in writing were selected. Following the procedure outlined above, written informed consent was obtained from those who met the criteria after fully explaining the study's purpose to students and their parents. Ultimately, the study included a total of four students, comprising three with ID and one with ASD. Table 1 illustrates the characteristics of the study participants.

Materials

In addition to WeDo and Mindstorms, LEGO Mindstorms commercial kits also encompass Boost, all serving as children's STEM toys commonly utilized within educational settings. These kits comprise various building blocks, including motors, hubs, light bulbs, and sensors, and offer an intuitive drag-and-drop canvas-coding environment compatible with tablets. Within this user-friendly coding environment, students can develop projects that exhibit movement, responsiveness, and interaction based on their created code.

In contrast to Mindstorms, WeDo and Boost feature elementary block levels and functions. Consequently, Mindstorms was excluded from this study owing to considerations regarding the participants' programming proficiency. Furthermore, the LEGO robot teaching tool selected for this study needed to accommodate all seven computational concepts (Sequences, Loops, Events, Parallelism, Conditionals, Operators, and Data). Following evaluation, the WeDo application was found capable of teaching all seven computational concepts but could only handle a maximum of five concepts within a single project. Conversely, the Boost application demonstrated proficiency in handling all computational concepts through activities involving a humanoid robot project named Vernie. The Vernie robot exhibits humanoid characteristics, capable of verbal communication and expressing a range of emotions through body movements. Its selection aimed to enhance student-robot interactions and sustain student engagement (Islam, Hasan, and Deowan 2023; Warren et al. 2015). Ultimately, the Vernie robot was chosen for the study, although it has since been discontinued as of 2024.

Procedure

The detailed procedures for the robot programming class are outlined in Table 2. Conducted by the first author, the class spanned a duration of 5 weeks for each participant, with each session extending for 45 min.

A pre-intervention survey was conducted before the beginning of the classes to establish a baseline. The purpose of measuring baselines is to track participants' development of computational concepts and practices and to set their behavioral goals (Figure 1(a)). The participants were divided into three groups according to the level and degree of support they required from the teacher based on their baseline (Table 3).

Following the baseline measurement, participants underwent the robot programming class (Intervention), the independent variable of this study, which consisted of nine sessions in total (comprising seven class sessions and two project sessions). The robot programming curriculum was developed based on previous studies (Fronza, Ioini, and Corral 2017; Grover et al. 2017; Kutay and Oner 2022; Merkouris and Chorianopoulos 2019; Statter and Armoni 2020; Witherspoon et al. 2017) (Table 4). The curriculum consisted of seven classes focused on individual computational concepts and two project classes where participants could use all the computational concepts they had learned.

Table 1. Participant characteristics.

Participants	Chronological age	Grade	Gender	Disability types	Ability to perform learning related to this study	IQ (Korean-Wechsler Intelligence Scale for Children-Fourth Edition, K-WISC-IV)
S1	15.0	2	Male	Autism Spectrum Disorder	(a) Answer questions in spoken language, one or two sentences long. (b) The speed of learning and understanding of new knowledge is relatively fast.	70
S2	15.11	2	Male	Intellectual Disability	(a) Answer questions in spoken language one or two words long. (b) Requires verbal instruction (partial assistance)	49
S3	15.2	2	Male	Intellectual Disability	(a) Difficult to express colloquially, answer with shaking or turning heads, facial expressions, gestures, and handwriting. (b) Requires verbal instruction (partial assistance)	51
S4	14.3	2	Male	Intellectual Disability	(a) Difficult to express colloquially, answer with shaking or turning heads, facial expressions, gestures, and handwriting. (b) Requires physical assistance (overall assistance).	38

Table 2. Robot programming class procedure.

Pre-Step	Program Progression Step			Post-Step
Pre-intervention survey (Likert 5-Point Scale)	Baseline (3 sessions)	Robot programming class (Intervention)		Post-intervention survey (Likert 5-Point Scale)
		class (7 sessions)	Project (2 sessions)	

Figure 1. (a) baseline phase (manipulation exercise: waiting for a handshake), (b) program design (7th session).

Table 3. Participants' behavioral goals.

-Educational purpose: Students (S1, S2, S3, and S4) design a program that demonstrates understanding of computational concepts.
-Behavioral evaluation: Record the programming block related to the computational concept in the data form only when used in accordance with the participant's behavioral goals

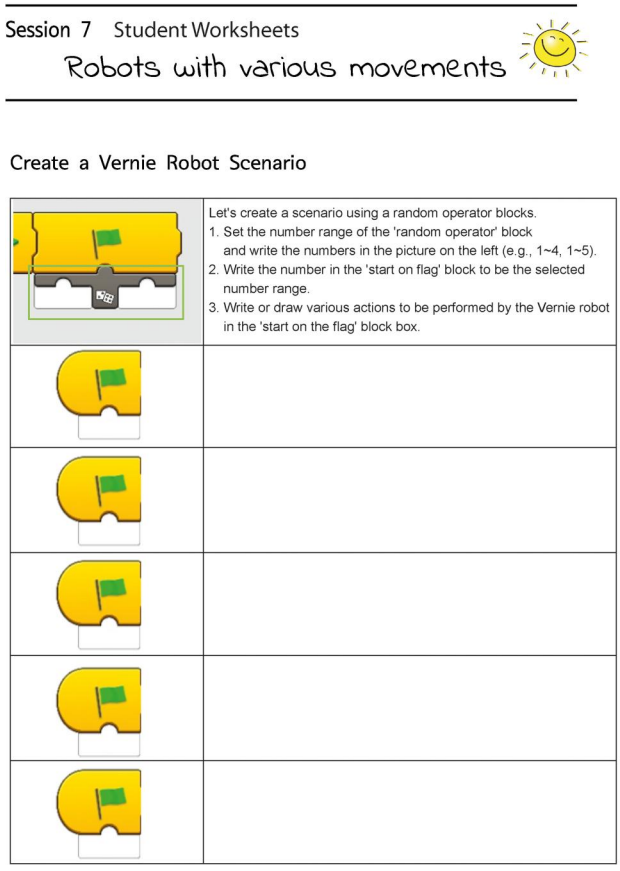
Educational objectives	S1	Give one programming problem related to the computational concept learned in each session, and give the instruction 'Create a scenario containing programming blocks related to the computational concept learned in this session, design the programming according to the scenario to move the Vernie robot', S1 successfully designs programs by independently using the block programming language of the LEGO Boost app installed on the tablet.
	S2 S3	Give one programming problem related to the computational concept learned in each session, and give the instruction 'Create a scenario containing programming blocks related to the computational concept learned in this session, design the programming according to the scenario to move the Vernie robot', S2 and S3 successfully design programs using the block programming language of the LEGO Boost app installed on the tablet with linguistic assistance.
	S4	Give one programming problem related to the computational concept learned in each session, and give the instruction 'Create a scenario containing programming blocks related to the computational concept learned in this session, design the programming according to the scenario to move the Vernie robot', S4 successfully designs programs using the block programming language of the LEGO Boost app installed on the tablet and by looking at linguistic assistance and programming block illustration hints associated with the concept.

The 7th session worksheet is shown in Figure 2; the scenario task was 'Creating a scenario using a random operator block'. The worksheet consisted of sequencing the components of small steps, such as (1) set the number range of the 'random operator' block, (2) write the number in the 'start on flag' block to be the selected number range, and (3) write or draw various actions to be performed by the Vernie robot in the 'start on the flag' block box. The worksheets of all sessions were presented in the form of a task

Table 4. Overview of robot programming curriculum.

Session	Activity title	Focused learning computational concepts	Computational concepts							Student's main activity
			S	L	E	P	C	O	D	
1	Hockey robot	Sequences	○		○				○	-View Demonstration (Modeling)
2	Maze moving robot	Data	○		○				○	-Look at the example code and try it
3	Handshake Robot	Conditionals	○		○		○		○	-Create scenarios and design programs using learned computational concepts (Figure 1(b))
4	Dancing robot	Loops	○	○	○	○			○	
5	Boxing robot	Parallelism	○	○	○	○			○	-Completing self-recording checklist of computational practices
6	Talking robot	Events	○	○	○	○	○		○	
7	Robots with various movements	Operators	○	○	○	○		○	○	
8	ProjectI	whole concepts	○	○	○	○		○	○	-Creating scenarios and designing programs that include full computational concepts
9	ProjectII	whole concepts	○	○	○	○	○		○	-Integrate what has been learned in previous lessons and recycle them appropriately

S: Sequences; L: Loops; E: Events; P: Parallelism; C: Conditionals; O: Operators; D: Data.



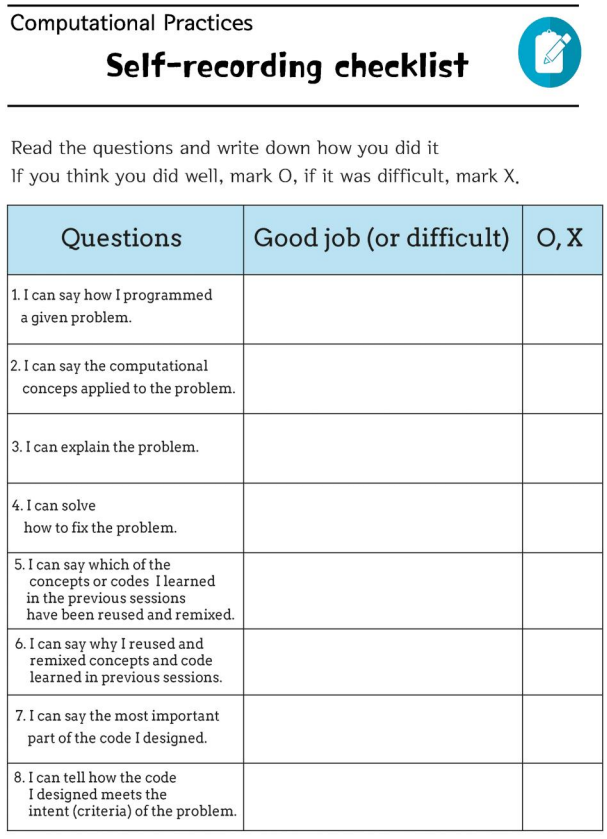


Figure 2. Student Scenario Worksheet (7th session) and Student Self-Recording Form.

analysis in which complex problems were divided into easy-to-handle components to help participants write scenarios (Alberto and Troutman 2017).

Computational practices were implemented during the robot programming class (Intervention). Computational practices were taught using a changing-criterion design among single-subject design to evaluate individual practices, and the effectiveness of each participant's practices was measured. The changing-criterion design has the advantage of a single-subject design emphasizing the clinical significance of the individual compared with oneself while examining the functional relationship through a gradual change in the target score (Alberto and Troutman 2017). The changing-criterion design was utilized because participants did not have to withdraw from the intervention after successful learning as the intervention explores participants' learning. All the participants systematically learned computational concepts using robots rather than simply playing games. Additionally, self-recording methods were taught to allow participants to self-manage their behavior to generalize and

maintain computational practices during the program (Figure 2).

After the completion of the robot programming class (Intervention), a post-intervention survey was conducted to explore changes in computational perspective.

Data collection and analysis

The data collection and analysis methods are detailed in Table 5. The data collection and analysis method of the computational concept is as follows. During the 9th session, we observed a video of designing an actual program based on a scenario created through the creation of activity sheets based on task analysis, and used a data form to record for each session whether actions using computational concepts occurred (see Table 6). In other words, it is a method of displaying the occurrence of a behavior in each character in a form in which character codes for various behaviors (behavior codes) are written in advance in each column of the data form (Alberto and Troutman 2017). The display method in this study was applied only when the programming block associated with the relevant computational concept was utilized to align with the participant's behavioral objective, irrespective of the number of actions. This determination was made after three authors collectively observed the video in its entirety. The operational definition of behavior using computational concepts includes seven elements. This behavior was encoded with the characters S, L, E, P, C, O, and D. Computational concepts related to each session are indicated by codes, referring to Table 4.

Table 5. Data collection and analysis.

CT three dimension	Independent variable	Dependent variable	Main evaluation methods	Supplementary assessment methods	Calculated data
Computational concepts	Robot programming class (Intervention) 9 sessions (7 sessions for class and 2 sessions for project)	Whether actions using computational concepts occur during program implementation during each session	Task analysis-based scenario worksheets	Program design video, field notes	Data format (9th session)
Computational practices		Effect of computational performance on program implementation during each session	Self-recording checklist		Changing criterion design graph (12 sessions including baseline)
Computational perspectives		Changes in computational perspective after completing all training	Pre-post intervention survey	Emotion card	Pre-post intervention survey results, 1 emotion card per participant

Table 6. Computational concepts observation results.

Participants: S1, S2, S3, S4
Behavior: Use LEGO blocks associated with the computational concepts according to the participant's behavioral goals
Observation time: Replay time of programming design video conducted in robot programming class (9 times in total)

Session	S1				S2				S3				S4			
1	S		E D		S		E D		S		E D		S		E D	
2	S		E D		S		E D		S		E D		S		E D	
3	S C		E D		S C		E D		S C		E D		S C		E D	
4	S	L	E D	P	S	L	E D	P	S	L	E D	P	S	L	E D	P
5	S	L	E D	P	S	L	E D	P	S	L	E D	P	S	L	E D	P
6	S C	L	E D	P	S C	L	E D	P	S C	L	E D	P	S C	L	E D	P
7	S	L O	E D	P	S	L O	E D	P	S	L O	E D	P	S	L O	E D	P
8	S	L O	E D	P	S	L O	E D	P	S	L O	E D	P	S	L O	E D	P
9	S C	L	E D	P	S C	L	E D	P	S C	L	E D	P	S C	L	E D	P
Behavior code	S = Sequences C = Conditionals				L = Loops O = Operators				E = Events D = Data				P = Parallelism			

The data collection and analysis method of the computational practices is as follows. Based on participants' self-recording checklists, actual program design videos were observed. The synthesis and scoring of these videos were conducted according to a rubric, with scores for each of the four scales subsequently converted. To tailor the rubric to participants' levels and abilities, item content and step-by-step levels were modified based on prior research (Fronza, Ioini, and Corral 2017). The rubric's validity was confirmed by three computer education experts and comprised eight questions, with two questions allocated to each of the four scales: (a) experimenting and iterating, (b) testing and debugging, (c) reuse and remixing, and (d) abstraction and modularization. Three authors assessed the questions based on the categories 'difficult to do', 'requires physical assistance (overall assistance)', 'requires verbal instruction (partial assistance)', and 'can perform well (good job)'. Scores ranging from 0–3 points were assigned corresponding to the evaluation phrases. Over the course of the sessions, participants' changes in computational practices were depicted through four graphs, illustrating the total score (3 points × 2 questions = 6 points) for each of the four scales numerically (refer to Figures 3 and 4).

In contrast to the evaluation of computational concepts, computational practices were assessed using a changing criterion design, aiming to discern functional relationships through gradual alterations in target scores. The method of gradually changing the target score for each participant is explained in relation to graph interpretation as follows. The graph comprises two main phases. The first phase was the baseline, and the second phase was the intervention. The intervention phase differed slightly for each participant but consisted of two to four sub-phases, with a dotted vertical line separating them. Each sub-phase had an intermediate criterion. In this study, the participants' actual practice scores (data points) are displayed as dots, and the data points in each sub-phase are connected. Graph (a) of S1 in Figure 3, S1 shows an average of zero practice levels in the baseline. The first intermediate practice level was set to a score two points higher than the baseline average practice in consideration of the participant's level. If the goal was achieved in more than two sessions, the next intermediate criterion (3 points) was raised by one point from the sixth session. As the score remained higher than the target score for the sixth to eighth sessions, the next intermediate criterion (4 points) was raised by 1 point from the ninth session. The first author offered verbal praise and encouragement to participants upon meeting intermediate criteria. These procedures were continued until the end of all sessions.

In addition to the primary evaluation method, this study aimed to enhance the objectivity and reliability of the analysis results by incorporating supplementary evaluation data. Specifically, detailed field notes and videos of participants' program designs were utilized (Lindsay and Hounsell 2017). Videos were recorded solely during sessions where participants engaged in actual programming activities across the nine sessions. Subsequently, three authors independently assessed the outcomes, discussing the occurrence of computational concepts, alterations in target scores for computational practices, and rubric scoring. In cases where opinions differed, a final evaluation was conducted through an opinion adjustment process. The final inter-rater reliability was high, at 0.97.

The data collection and analysis method from computational perspectives is pre- and post-intervention survey. This study provided participants with a 5-point Likert scale survey related to participants' perception of computers, robots, and programming before and after the start of class. The pre-post intervention survey consisted of the same five questions used in previous studies (Merkouris and Chorianopoulos 2019). Each response was scored as strongly disagree (1 point), disagree (2 points), undecided (3 points), agree (4 points), or strongly agree (5 points). The score was calculated by assigning the corresponding score to each response, adding them up, and then dividing by the number of participants.

Additionally, emotion cards were employed as supplementary evaluation materials to glean further insights into changes in participants' perspectives. Considering the participants' limited reading and writing abilities, pictures were also used as they could express participants' perceptions and emotions more clearly than text (Lee et al. 2018). After completing the class, participants were asked to select an emotion card to express their emotions or feelings (Merkouris, Chorianopoulos, and Kameas 2017).

Results

Computational concepts

In this study, the data form that records whether participants' behavior using computational concepts occurred by session is shown in Table 6.

As can be observed in Table 6, as the sessions progressed, all the participants used the LEGO blocks

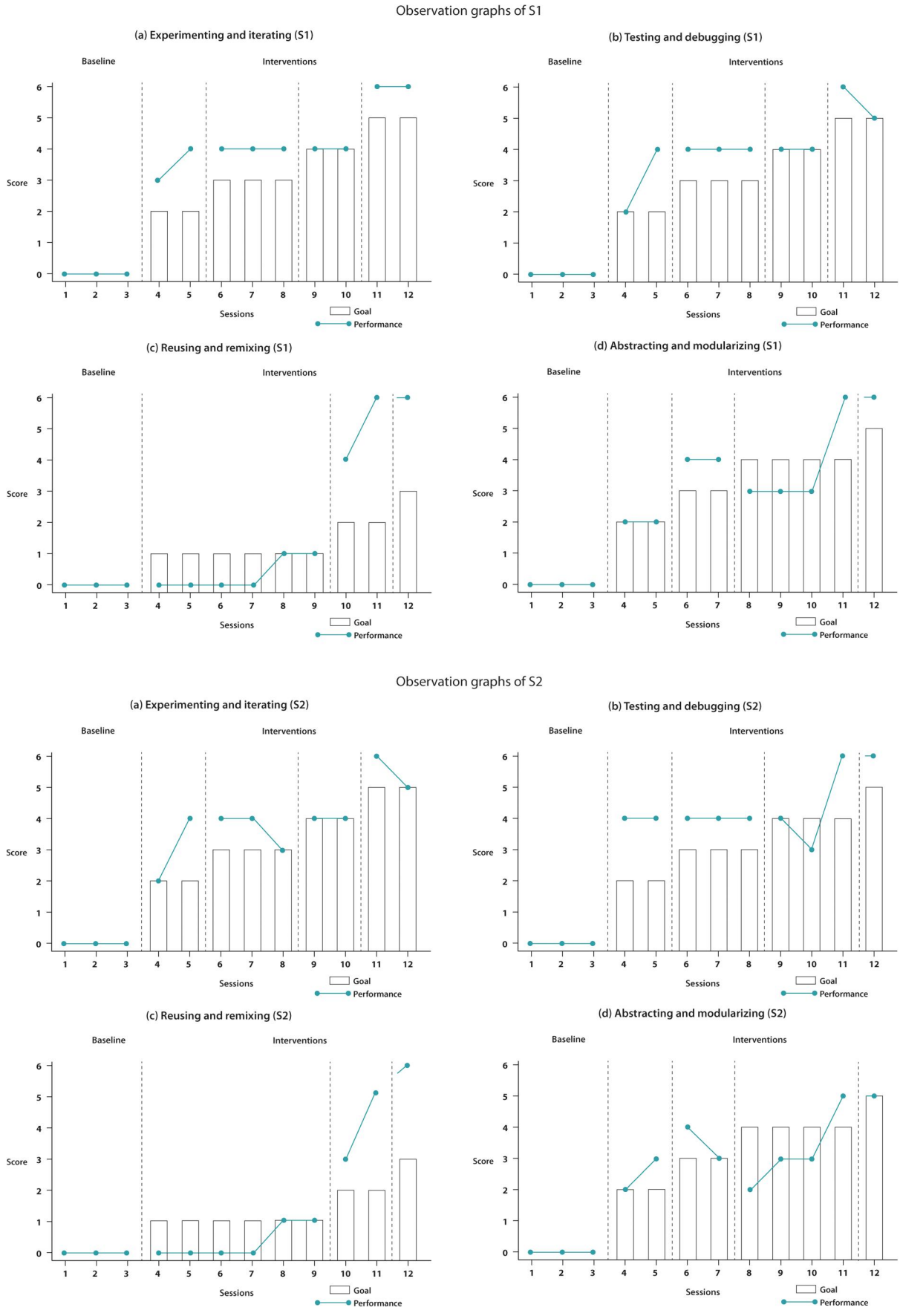


Figure 3. Observation graphs of S1 and S2.

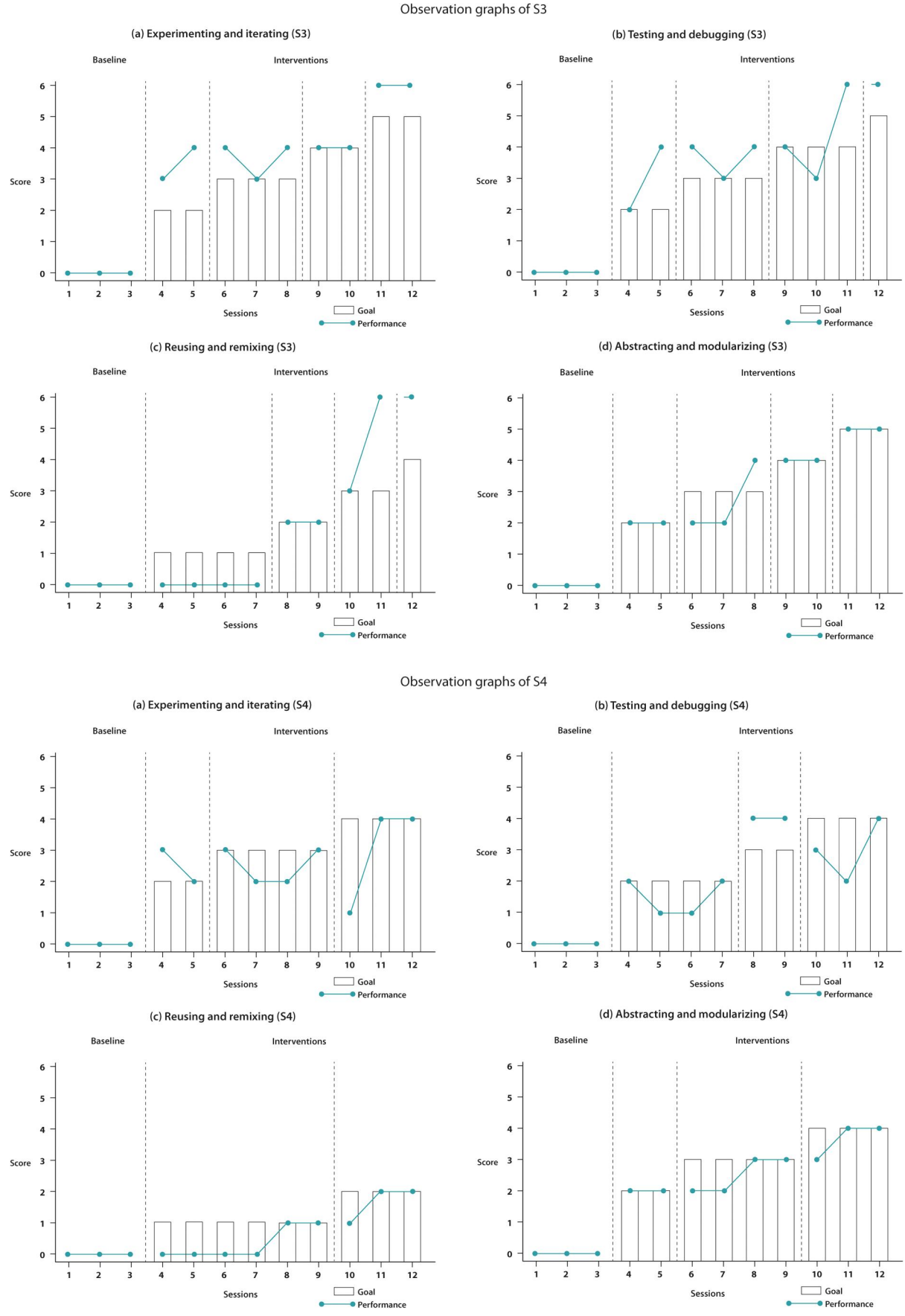


Figure 4. Observation graphs of S3 and S4.

related to the computational concepts according to their behavioral goals. A characteristic feature of this study's observation of behavior using computational concepts is its focus on acquiring the concept of focused learning. The participants first learned the concept of Loops in Session 4, but almost none of them used it well in Sessions 6 and 7. The concept of parallelism, in which two or more code instructions are executed simultaneously, was first learned in Session 5 but was not used by all participants in Session 6. This is because solving a given problem without necessarily using a block related to the computational concepts is possible.

Computational practices

In this study, the graphical results of changes in computational practices as the session progressed are shown in Figures 3 and 4.

In Figures 3, 4, S1, S2, and S3 had an average of 0 points in the baseline phase in all four computational practices but met the criteria in three to four consecutive phases. In the 11th and 12th sessions, project activities achieved more than the final goal by obtaining five and six points, respectively. In contrast, graphs (a), (b), and (d) of S1, S2, and S3 show similar data point trends as the regression increases, whereas graph (c) shows a different trend. In graph (c), zero points were continuously obtained during the fourth to seventh sessions because the three participants concentrated on concept learning, and the behavior of reusing and remixing the concepts or example codes learned in the previous session did not appear. The researcher changed the intervention method from the eighth session to provide a collection of blocks or example codes that represented the computational concepts learned in the previous session and demonstrated the specific method of reusing and remixing. After the change in the training method in the eighth session, it was observed that S1, S2, and S3 performed the behaviors related to reusing and remixing, and the highest score (6 points) was obtained in the last session.

As shown in Figure 4, the target score of S4 was lower than that of S1, S2, and S3. Similar to the other three participants, S4 met the intermediate criterion in 2–3 consecutive phases of all graphs, and in the final session, 100% of the target score was obtained. In graph (c), S4 exhibited relatively low practice, similar to the other three participants. However, the behavior of reusing and remixing through the data point connection line according to the session increased. That is, students who need more support from teachers learn at a slower rate than those who need less support. However, if given more time, they may exceed the target score. Finally, all participants showed an improvement from a minimum of two points to a maximum of six points compared to their baseline in all four graphs.

Computational perspectives

Table 7 shows the results of measuring the participants' perceptions of computers, robots, and programming before and after class. Finally, the participants were asked, 'How do I feel after all the robot programming classes are completed?' The selection of the emotion card is shown in Figure 5.

Table 7 shows that the participants' interest in programming and robots increased after the education. In addition, the participants became aware of learning more about CS education or having a job in the computer field from a more positive perspective. However, interest in computers decreased. These findings are judged to result from participants not directly experiencing computers in actual education, as participants mainly think of a desktop as a computer device. This can be seen from the fact that participants' perceptions of learning computers and computer jobs mentioned in Questions 4 and 5 were more positive in the post-intervention survey. As shown in Figure 5, participants selected an emotion card containing a stable, positive, great, or fun expression. All participants expressed positive feelings about the robot programming.

Table 7. Survey results.

Order	Question	Pre-intervention survey ($N = 4$) Rating (total score/N)	Post-intervention survey ($N = 4$) Rating (total score/N)
1	How interested are you in computers?	4.5	4.25
2	How interested are you in programming?	2.25	3.75
3	How interested are you in robots?	2.75	4.5
4	Do you want to learn computer, programming, and robot from now on?	3.25	4
5	Are you thinking of getting a computer job in the future?	2.75	3.25

*Perfect score = 5 points.

Figure 5. Results of emotion card selection.

Discussion

Computational concepts

According to the analysis results presented in Table 6, participants' computational concepts ultimately demonstrated improvement. The computational concepts the participants found difficult in the early stages of learning were parallelism, conditionals, and operators. Specifically, the difficulty of the term itself (parallelism) and the difficulty of recognizing and explaining concepts related to the block (conditionals, operators), apart from using the block, required more learning and practice time than other concepts and teacher help. These results are consistent with the difficulties experienced in understanding some concepts (conditional statements) in a study that reported that most of the 22 elementary school students used sequences fully and accurately, had limited or no consistent understanding of conditionals, and had difficulty constructing loops apart from understanding and recognizing loops as cumulative effects (Luo, Israel, and Gane 2022). While 20 middle school students correctly used concepts of sequences, events, loops, and parallelism in Minecraft environments, variables, conditionals, and operators are also consistent with the difficulties experienced in understanding some concepts (e.g. conditionals, operators) compared to studies reported as less successful (Kutay and Oner 2022).

On the other hand, all participants in this study easily understood the concept of loops. This differs from research showing that participants are sequences are easy to learn but that struggle with loops. This may mean that a LEGO Boost coding environment, such as Minecraft, facilitates loop structure learning (Grover 2014). It has been pointed out that certain programming environments may affect the computational concepts students use when solving computing tasks because of the differences between the Alice and Scratch programming environments (Allsop 2019; Kutay and Oner 2022).

To address challenges in grasping concepts, this study assessed participants' understanding and provided feedback through task analysis-based scenario worksheets. Leveraging the ability to construct problem scenarios aligned with the computational concepts learned, the study ensured precise comprehension. Utilizing task analysis, all participants successfully tackled the two final project problems tailored to their proficiency levels. The results of this study are better than those of other studies that have reported that if elementary school students' knowledge of individual concepts is not yet established, they are likely to have difficulty solving problems related to combined programming concepts (Luo, Israel, and Gane 2022).

Computational practices

Based on the analysis results depicted in Figures 3, 4, participants' computational practices exhibited improvement. This study's findings are characterized by two key aspects. First, the participants spent most of their time testing and debugging graph (b) when designing the actual program. Reusing and remixing graph (c) showed a completely different appearance from graphs (a), (b), and (d), which are gradually improving. Not all participants applied reusing and remixing at the beginning of class. Therefore, considerable feedback from the teacher was required. These results are consistent with previous findings that Minecraft-based coding artifacts are used most of the time for testing and debugging, whereas reuse and remixing are used the least (Kutay and Oner 2022).

Second, the participants' successful learning of computational concepts using task analysis showed that it was connected to the improvement of their actual computational practices. Specifically, the

participants thought about the overall appearance of the program to be designed once more while producing the scenario. Participants' self-records prepared in the class summary stage were found to help explain how participants programmed, what computational concepts were used, what were the most important parts of the program, and whether the program design met the needs of the problem or reduced actual program errors.

Computational perspectives

According to the analysis results in Table 7 and Figure 5, the participants' computational perspectives changed positively. All participants expressed heightened interest and positive sentiments towards careers in robot, programming, and computing compared with the pre-intervention survey results. These results support studies that showed feelings of excitation, self-esteem, and happiness when students with learning disabilities assembled robots, wrote codes, and observed the robots' movements (Kert, Yeni, and Fatih Erkoç 2022).

Conclusion

This study aimed to investigate the enhancement of CT skills through LEGO robot programming education among middle school students with ASD or ID, with developmental disabilities. The conclusions drawn from the discussion of the research findings are as follows.

First, middle school students with ASD or ID demonstrated positive improvement in all three dimensions of CT through robot programming education, mirroring trends observed among non-disabled students. All participants exhibited immediate benefits from the intervention, with these effects seemingly sustained even after the intervention concluded. Post-intervention, participants expressed newfound interest in programming, extending beyond the Vernie robots provided by LEGO Boost to include other models such as Guitar 4000, Frankie the Cat, AutoBuilder, and M.T.R.4 (Multi-Tooled Rover 4). Additionally, some participants (e.g. S1) ventured into advanced programming using Mindstorms kits. This underscores the role of robot teaching tools in cultivating enthusiasm and engagement in programming, ultimately contributing to the enhancement of CT skills. Furthermore, the effectiveness of individualized teaching and learning strategies tailored to students' cognitive abilities, level of support, and specific needs highlights their efficacy in programming education.

Second, it suggests that even students with severe intellectual disabilities may have partial or full participation in programming learning using robot teaching aids. In the education of abstract computational concepts, students with low abilities can sense and feel the concept with their bodies, while competent students can proceed to more intellectual learning stages, which is a similar conclusion to the study that suggests that it provides alternative learning opportunities for all students (Merkouris and Chorianopoulos 2019). WeDo and Boost have Scratch-based coding environments, but they differ from Scratch in that the block level is at the basic level. Considering the programming environment, which differs in the level and function of the block, and the difficulty of content knowledge (e.g. mathematics), etc., it is judged that learners who want to learn basic computational concepts should be introduced through robot teaching aids with easy coding environments.

The significance of this study lies in its analysis of how LEGO robot programming education, using individualized teaching and learning strategies, enhances CT skills among students with ASD or ID. Moreover, it provides valuable insights and recommendations for improving accessibility in the design of future programming classes.

Limitations

The limitations of this study are as follows. First, the sample size is limited owing to the recruitment of participants through convenience sampling. Given the diverse abilities and tasks among individuals with ASD or ID, the findings may not be generalizable to all individuals within these populations. Future research should aim to overcome this limitation by conducting programming education studies with more diverse samples to gather additional results and evidence.

Second, the results may be influenced not only by the intervention but also by the Hawthorne effect. Although efforts were made to minimize this effect by ensuring stable and low baseline measurements before the intervention, participants' awareness of being filmed and their perceptions of the research situations could still impact the validity of the results.

Third, comparative group studies are warranted. To gain a more comprehensive understanding of the effects observed in this study, further research using appropriate comparison groups, such as a group of

students with the same disability or a group of non-disabled students of the same chronological age, is necessary.

Ethical approval

All procedures performed were in accordance with the ethical standards of the institutional and with the 1964 Helsinki Declaration and its later amendments or comparable ethical standards.

Informed consent

Informed consent was obtained from all individual participants and parents who participated in the study.

Disclosure statement

No potential conflict of interest was reported by the authors.

ORCID

MiJeong Kim http://orcid.org/0000-0002-2139-9976
JaMee Kim http://orcid.org/0000-0002-5949-9753
WonGyu Lee http://orcid.org/0000-0001-5335-2913

Data availability statement

All data generated or analyzed during this study are included in this published article.

References

Adams, Kim, and Al Cook. 2014. "Access to Hands-On Mathematics Measurement Activities Using Robots Controlled Via Speech Generating Devices: Three Case Studies." *Disability and Rehabilitation. Assistive Technology* 9 (4): 286–298. https://doi.org/10.3109/17483107.2013.825928.

Alberto, Paul A., and Anne C. Troutman. 2017. *Applied Behavior: Analysis for Teachers*. 10th ed. Boston: Pearson.

Allsop, Yasemin. 2019. "Assessing Computational Thinking Process Using a Multiple Evaluation Approach." *International Journal of Child-Computer Interaction* 19: 30–55. https://doi.org/10.1016/j.ijcci.2018.10.004.

Barr, Valerie, and Chris Stephenson. 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. https://doi.org/10.1145/1929887.1929905.

Bell, Tim, Ian H. Witten, and Mike Fellows. 1998. *Computer Science Unplugged: Off-Line Activities and Games for All Ages*. Christchurch, New Zealand: Computer Science Unplugged.

Brennan, Karen, and M. Resnick. 2012. "New Frameworks for Studying and Assessing the Development of Computational Thinking." In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association* (Vol. 1, 25). Vancouver: AERA.

Conti, Daniela, Grazia Trubia, Serafino Buono, Santo F. Di Nuovo, and Alessandro Di Nuovo. 2021. "An Empirical Study on Integrating a Small Humanoid Robot to Support the Therapy of Children with Autism Spectrum Disorder and Intellectual Disability." *Interaction Studies. Social Behaviour and Communication in Biological and Artificial Systems* 22 (2): 177–211. https://doi.org/10.1075/is.21011.con.

Desideri, Lorenzo, Marco Negrini, Massimiliano Malavasi, Daniela Tanzini, Aziz Rouame, Maria Cristina Cutrone, Paola Bonifacci, and Evert-Jan Hoogerwerf. 2018. "Using a Humanoid Robot as a Complement to Interventions for Children with Autism Spectrum Disorder: A Pilot Study." *Advances in Neurodevelopmental Disorders* 2 (3): 273–285. https://doi.org/10.1007/s41252-018-0066-4.

Díaz-Boladeras, Marta, Ada Claver i Díaz, and Marta García-Sánchez. 2023. "Robots for Inclusive Classrooms: A Scoping Review." *Universal Access in the Information Society: International Journal* 23: 1–25. https://doi.org/10.1007/s10209-023-01065-z.

Fronza, Ilenia, Nabil El Ioini, and Luis Corral. 2017. "Teaching Computational Thinking Using Agile Software Engineering Methods: A Framework for Middle Schools." *ACM Transactions on Computing Education* 17 (4): 1–28. https://doi.org/10.1145/3055258.

González-González, C. S., P. Caballero-Gil, A. García-Holgado, F. J. García-Peñalvo, J. Molina, J. M. del Castillo-Olivares, and Ramos. S. 2021. "COEDU-IN Project: An Inclusive Co-educational Project for Teaching Computational Thinking and Digital Skills at Early Ages." In *2021 International Symposium on Computers in Education (SIIE)* 1–4. https://doi.org/10.1109/SIIE53363.2021.9583648.

González-González, Carina S., Erika Herrera-González, Lorenzo Moreno-Ruiz, Nuria Reyes-Alonso, Selene Hernández-Morales, Maria D. Guzmán-Franco, and Alfonso Infante-Moro. 2019. "Computational Thinking and Down Syndrome: An Exploratory Study Using the KIBO Robot." *Informatics* 6 (2): 25. https://doi.org/10.3390/informatics6020025.

Grover, Shuchi, and Roy D. Pea. 2013. "Computational Thinking in K–12: A Review of the State of the Field." *Educational Researcher* 42 (1): 38–43. https://doi.org/10.3102/0013189X12463051.

Grover, Shuchi. 2014. *Foundations for Advancing Computational Thinking: Balanced Designs for Deeper Learning in an Online Computer Science Course for Middle School Students*. Stanford, CA: Stanford University.

Grover, Shuchi, Satabdi Basu, Marie Bienkowski, Michael Eagle, Nicholas Diana, and John Stamper. 2017. "A Framework for Using Hypothesis-Driven Approaches to Support Data-Driven Learning Analytics in Measuring Computational Thinking in Block-Based Programming Environments." *ACM Transactions on Computing Education* 17 (3): 1–25. https://doi.org/10.1145/3105910.

Islam, Mohd Ariful, Md Mehedi Hasan, and Shamim Ahmed Deowan. 2023. "Robot-Assisted Training for Children with Autism Spectrum Disorder: A Review."

Journal of Intelligent & Robotic Systems 108 (3): 1–36. https://doi.org/10.1007/s10846-023-01872-9.

Kang, Chunhua, Na Liu, Yinrui Zhu, F. Li, and Pingfei Zeng. 2023. "Developing College Students' Computational Thinking Multidimensional Test Based on Life Story Situations." *Education and Information Technologies* 28 (3): 2661–2679. https://doi.org/10.1007/s10639-022-11189-z.

Kert, Serhat Bahadir, Sabiha Yeni, and Mehmet Fatih Erkoç. 2022. "Enhancing Computational Thinking Skills of Students with Disabilities." *Instructional Science* 50 (4): 625–651. https://doi.org/10.1007/s11251-022-09585-6.

Knight, Victoria F., John Wright, and Andrea DeFreese. 2019. "Teaching Robotics Coding to a Student with ASD and Severe Problem Behavior." *Journal of Autism and Developmental Disorders* 49 (6): 2632–2636. https://doi.org/10.1007/s10803-019-03888-3.

Kolne, Kendall, Sunny Bui, and Sally Lindsay. 2021. "Assessing the Environmental Quality of an Adapted, Play-Based LEGO® Robotics Program to Achieve Optimal Outcomes for Children with Disabilities." *Disability and Rehabilitation* 43 (25): 3613–3622. https://doi.org/10.1080/09638288.2020.1743776.

Kutay, Emine, and Diler Oner. 2022. "Coding with Minecraft: The Development of Middle School Students' Computational Thinking." *ACM Transactions on Computing Education* 22 (2): 1–19. https://doi.org/10.1145/3471573.

K-12 Computer Science Framework Steering Committee. 2016. *K-12 Computer Science Framework. Technical Report*. New York, NY: Association for Computing Machinery. https://doi.org/10.1145/3079760.

Lee, Shuk Ching Clara, Stephen H. F. Lam, Sally T. K. Tsang, Cheong M. C. Yuen, and Carmen K. M. Ng. 2018. "The Effectiveness of Technology-Based Intervention in Improving Emotion Recognition Through Facial Expression in People with Autism Spectrum Disorder: A Systematic Review." *Review Journal of Autism and Developmental Disorders* 5 (2): 91–104. https://doi.org/10.1007/s40489-017-0125-1.

Lee, Irene, Fred Martin, Jill Denner, Bob Coulter, Walter Allan, Jeri Erickson, Joyce Malyn-Smith, and Linda Werner. 2011. "Computational Thinking for Youth in Practice." *ACM Inroads* 2 (1): 32–37. https://doi.org/10.1145/1929887.1929902.

Lindsay, Sally, and Kara Grace Hounsell. 2017. "Adapting a Robotics Program to Enhance Participation and Interest in STEM Among Children with Disabilities: A Pilot Study." *Disability and Rehabilitation. Assistive Technology* 12 (7): 694–704. https://doi.org/10.1080/17483107.2016.1229047.

Lindsay, Sally, and Ashley Lam. 2018. "Exploring Types of Play in an Adapted Robotics Program for Children with Disabilities." *Disability and Rehabilitation. Assistive Technology* 13 (3): 263–270. https://doi.org/10.1080/17483107.2017.1306595.

Luo, Feiya, Maya Israel, and Brian Gane. 2022. "Elementary Computational Thinking Instruction and Assessment: A Learning Trajectory Perspective." *ACM Transactions on Computing Education* 22 (2): 1–26. https://doi.org/10.1145/3494579.

Merkouris, Alexandros, and Konstantinos Chorianopoulos. 2019. "Programming Embodied Interactions with a Remotely Controlled Educational Robot." *ACM Transactions on Computing Education* 19 (4): 1–19. https://doi.org/10.1145/3336126.

Merkouris, Alexandros, Konstantinos Chorianopoulos, and Achilles Kameas. 2017. "Teaching Programming in Secondary Education Through Embodied Computing Platforms: Robotics and Wearables." *ACM Transactions on Computing Education* 17 (2): 1–22. https://doi.org/10.1145/3025013.

Papert, Seymour. 1980. *Mindstorms: Children, Computers, and Powerful Ideas*. New York, NY: Basic Books.

Perez, J., M. Azuaje, C. Leon, and O. Pedroza. 2021. "Effects of Social Robotics on Episodic Memory in Children With Intellectual Disabilities." *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje* 16 (4): 393–399. https://doi.org/10.1109/RITA.2021.3125899.

Pollak, Micheal, and Martin Ebner. 2019. "The Missing Link to Computational Thinking." *Future Internet* 11 (12): 263. https://doi.org/10.3390/fi11120263.

Santos, Laura, Silvia Annunziata, Alice Geminiani, Alessia Ivani, Alice Giubergia, Daniela Garofalo, Arianna Caglio, et al. 2023. "Applications of Robotics for Autism Spectrum Disorder: A Scoping Review." *Review Journal of Autism and Developmental Disorders* 10: 1–22. https://doi.org/10.1007/s40489-023-00402-5.

Saqr, Mohammed, Kowk Ng, Solomon Sunday Oyelere, and Matti Tedre. 2021. "People, Ideas, Milestones: A Scientometric Study of Computational Thinking." *ACM Transactions on Computing Education* 21 (3): 1–17. https://doi.org/10.1145/3445984.

So, Wing-Chee, Miranda Kit-Yi Wong, Carrie Ka-Yee Lam, Wan-Yi Lam, Anthony Tsz-Fung Chui, Tsz-Lok Lee, Hoi-Man Ng, Chun-Hung Chan, and Daniel Chun-Wing Fok. 2018. "Using a Social Robot to Teach Gestural Recognition and Production in Children with Autism Spectrum Disorders." *Disability and Rehabilitation. Assistive Technology* 13 (6): 527–539. https://doi.org/10.1080/17483107.2017.1344886.

Statter, David, and Michal Armoni. 2020. "Teaching Abstraction in Computer Science to 7th Grade Students." *ACM Transactions on Computing Education* 20 (1): 1–37. https://doi.org/10.1145/3372143.

Syriopoulou-Delli, Christine K., and Eleni Gkiolnta. 2022. "Review of Assistive Technology in the Training of Children with Autism Spectrum Disorders." *International Journal of Developmental Disabilities* 68 (2): 73–85. https://doi.org/10.1080/20473869.2019.1706333.

Taylor, Matthew S. 2018. "Computer Programming with Pre-K Through First-Grade Students with Intellectual Disabilities." *The Journal of Special Education* 52 (2): 78–88. https://doi.org/10.1177/0022466918761120.

Tedre, Matti, and Peter J. Denning. 2016. "The Long Quest for Computational Thinking." In *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*, 120–129. Koli, Finland: ACM. https://doi.org/10.1145/2999541.2999542.

Tucker, Allen. 2003. *A Model Curriculum for K–12 Computer Science: Final Report of the ACM K–12 Task Force Curriculum Committee*. New York: Association for Computing Machinery. https://doi.org/10.1145/2593247.

Warren, Zachary E., Zhi Zheng, Amy R. Swanson, Esubalew Bekele, Lian Zhang, Julie A. Crittendon, Amy F. Weitlauf, and Nilanjan Sarkar. 2015. "Can Robotic Interaction Improve Joint Attention Skills?" *Journal of Autism and Developmental Disorders* 45 (11): 3726–3734. https://doi.org/10.1007/s10803-013-1918-4.

Wing, Jeanette M. 2006. "Computational Thinking." *Communications of the ACM* 49 (3): 33–35. https://doi.org/10.1145/1118178.1118215.

Wing, Jeanette M. 2014. "Computational Thinking Benefits Society." *40th Anniversary Blog of Social Issues in Computing* (blog), January 10, 2014. https://socialissues.cs.toronto.edu/index.html%3Fp=279.html.

Witherspoon, Eben B., Ross M. Higashi, Christian D. Schunn, Emily C. Baehr, and Robin Shoop. 2017. "Developing Computational Thinking Through a Virtual Robotics Programming Curriculum." *ACM Transactions on Computing Education* 18 (1): 1–20. https://doi.org/10.1145/3104982.

Yadav, Aman, Hai Hong, and Chris Stephenson. 2016. "Computational Thinking for All: Pedagogical Approaches to Embedding 21st Century Problem Solving in K-12 Classrooms." *TechTrends: Linking Research & Practice to Improve Learning* 60 (6): 565–568. https://doi.org/10.1007/s11528-016-0087-7.

Yadav, Aman, Chris Stephenson, and Hai Hong. 2017. "Computational Thinking for Teacher Education." *Communications of the ACM* 60 (4): 55–62. https://doi.org/10.1145/2994591.

Zigmond, Naomi P., and Amanda Kloo. 2017. *General and Special Education Are (and Should Be) Different.* New York, NY: Taylor and Francis Inc. https://doi.org/10.4324/9781315517698.