Innovative Voice-Activated Robots for Computational Thinking Education: Design and Development

JUDY C.R. TSENG
Department of Computer Science and Information Engineering
Chung Hua University
Hsinchu
TAIWAN

WEI LI
Ph.D. Program in Engineering Science
Chung Hua University
Hsinchu
TAIWAN

Abstract: - With the advent of the digital age, countries worldwide have begun to emphasize computational thinking education, hoping to cultivate learners' abilities to meet the requirements of future talents. Currently, computational thinking education in young children is mainly based on visual programming on computers or robots. However, using computers requires a prior understanding of abstract thinking, which is difficult for young children to master. To meet the need for cultivating computational thinking in younger children, this study combines a tangible robot with a verbal user interface to develop a set of voice-activated programming robots suitable for younger children. The learner gives verbal commands to make the robot perform the specified actions and complete the problem-solving task. This approach allows children to ignore the syntax of the programming language and thus focus more on problem-solving.

Key-Words: - Computational Thinking, Educational Robot, Verbal Programming Language

Received: June 19, 2022. Revised: March 20, 2023. Accepted: April 16, 2023. Published: May 26, 2023.

1 Introduction

Information technology has changed how people work, live, learn, and play in this digital era. Countries worldwide have begun emphasizing computational thinking education to cultivate talent suited to this digital era, hoping to foster learners' problem-solving skills and creative thinking, [1]. Computational thinking uses computer logic for problem-solving and system design, [2]. In response to the rapidly changing digital era, many scholars believe that it's better to cultivate computational thinking as earlier as possible, [3], [4], [5]. Currently, the cultivation of computational thinking in young children is mainly based on visual programming languages such as Scratch. Visual programming languages are more manageable for young children than text-based programming languages such as C and Python, [6], [7]. For example, Scratch allows students to drag and drop blocks without having to write a program, overcoming the problem that beginners tend to get lost in programming code or syntax, thus allowing students to focus more on problem-solving and facilitating the development of computational thinking, [8], [9].

However, visual programming involves computer operation, and learners must have a certain level of abstract thinking, which is difficult for young children to comprehend, [10]. Since children aged 7 to 11 are in the concrete operational stage, they tend to solve problems based on concrete physical experiences, [11]. Using robots as learning vehicles for computational thinking can enable children to observe the results of programming through the robot's actions, thereby increasing their own concrete experiences and making it easier to understand the concepts, [12]. For example, scholars have used tangible graphical programming robots to teach computational thinking to children aged 5-9 years old and found that the children's computational thinking skills improved significantly, [13]. Many studies have applied robots to computational thinking learning, [14], [15], [16] and have achieved promising results. However, most of these studies only use

robots to show the results of programming. The programming work still needs to be done on a computer, which is not advantageous for young children whose abstract thinking is still developing. On the other hand, as visual programming mainly uses block-dragging to design programs, students may use guesswork, intuition, or trial-and-error methods to solve problems, disregarding systematic, holistic operational thinking to solve problems, [17].

Before children start speaking, their language system in the brain processes phonology, semantics, and pragmatics to fulfill the goal of communicating, [18], enabling them to understand what they are expressing and conveying the concepts they possess. Thus, verbal programming languages should be more suitable for cultivating young children's computational thinking than visual programming languages. Research has found that students use language as a tool to help them solve problems and think, [19]. However, there has yet to be a programming robot that integrates a physical robot and a verbal user interface to meet the needs of younger children in fostering computational thinking.

Given this, this study proposed a verbal programming language for young children, called the V language, and developed an educational robot for computational thinking based on the V language. The educational robot enables children to give voice commands by the V language grammar to make the robot perform specific actions to complete problem-solving tasks. If the commands are incorrect, appropriate feedback is provided to help the children learn the correct grammar. With the development of this educational robot, an innovative computational thinking learning approach suitable for young children is provided, which is expected to effectively improve the effectiveness of computational thinking education for young children.

2 Literature Review

In this section, a review of related literature related to this research is conducted. The literature is divided into three sub-sections, which successively discuss the related literature on computational thinking, education robots, and programming languages.

2.1 Computational Thinking

Computational thinking skills enable students to analyze problems from different perspectives and solve them effectively, [20], and thus incorporating computational thinking into information education has become a global trend, [21]. Many countries have cultivated children in computational thinking, critical thinking, and problem-solving skills from a young age, [22]. Many scholars believe computational thinking and problem-solving skills development should begin as early as possible, [3], [4].

Many scholars have proposed different definitions of computational thinking, such as, [2], who believes that computational thinking is a way of thinking about problem-solving and system design using computer logic. Google refers to it as a problem-solving process, such as sorting and analyzing data logically and producing solutions through sequential steps, [23]. In summary, computational thinking can be defined as the ability to use computational methods and tools to solve problems.

Since programming is a great way to create computational works and demonstrate computational thinking abilities, [24], Several studies have found that learning programming can improve students' computational thinking, [13], [25]. As a result, most schools have students learn computer programming to develop their computational thinking abilities, [26]. However, learners must have a certain level of abstract thinking to operate computers, which is unfavorable for young students whose abstract thinking has yet to develop fully, [10].

2.2 Educational Robots

With the development of advanced technologies, robots that integrate various technologies have been developed and applied to various fields. The applications of robots in education have also become more and more versatile, [27], [28], [29]. By incorporating robots into teaching with the creativity of teachers, students can learn in a creative environment, thereby fostering their technological integration ability, problem-solving ability, and creativity, [30].

Research has indicated that robots can greatly assist education at all education levels, [31]. Besides increasing students' motivation, [32], they can also increase students' concrete experiences, making it easier for them to understand the learning material, [8]. Concrete experiences have varying degrees of impact on students of different ages. Piaget's theory of child cognitive development states that children aged 7 to 11 are in the concrete operations period and need to solve problems based on concrete experiences, [11]. Therefore, robots

can be a good learning vehicle for students at this stage.

Previous research has also found that arranging robot movements can cultivate students' computational thinking skills and stimulate hand-eye coordination, [3]. As a result, many scholars have tried to develop learning strategies for handling complex tasks through educational robots to promote the development of students' computational thinking, [11], [30]. Therefore, robots are also considered to be a good platform for learning computational thinking, [33]. However, most of these studies only use robots to show the results of programming, and the programming work still needs to be done on a computer, which still hinders young students' learning.

2.3 Programming Languages

Programming is a fundamental skill in information technology and a way of solving problems through programming languages, [34]. Programming languages nowadays can be divided into three types: textual, visual, and verbal.

Textual programming languages, including C, Java, Python, etc., are constructed using strings that conform to a specific syntax. For beginners in programming, learning the syntax of text-based programming languages is a big challenge and can also be one of the reasons for feeling discouraged, [35]. For beginners or younger learners, the learning threshold for text-based programming languages is higher and harder to start with, [36].

Visual programming languages were created to reduce the barriers to programming entry, eliminate syntax's complexity, and allow learners to concentrate on visualizing solutions to problems, [37]. Visual programming languages primarily use blocks of images to express solutions to problems and have simple operations, making them more suitable for beginners or younger learners compared to textual programming languages, [38]. Visual programming also enables learners to concentrate on systematic, logical thinking for problem-solving rather than being frustrated by a lack of familiarity with programming syntax, [39]. However, the simplicity of visual programming language's operations may lead learners to solve problems through guessing, intuition, or trial-and-error rather than systemic, computational thinking, [17].

Verbal programming languages are written through oral narrative. Before children start speaking, their language system in the brain processes phonology, semantics, and pragmatics to fulfill the goal of communicating, [18], enabling them to understand what they are expressing and conveying the concepts they possess. Thus, verbal programming languages should be more suitable for cultivating young children's computational thinking than visual programming languages. Its language structure is also too complex for young children to learn.

To help young children develop computational thinking, this study proposes a verbal programming language and creates an educational robot for computational thinking accordingly. The robot enables young children to use simple verbal programming language to give commands and complete specified tasks as instructed. By organizing the logical structure of speech commands and obtaining the concrete experience by the robot feedback, the solution to the problem can be constantly improved, thus enhancing their computational thinking ability.

3 The Verbal Programming Language for The Computational Thinking Educational Robot

To develop a voice-activated educational robot suitable for young children to cultivate computational thinking, this research uses the ASUS Zenbo robot as a platform and, based on the actions that the Zenbo robot can perform, a verbal programming language is designed called the V language. Children can use spoken commands to give instructions, which follow the syntax of the V language, to the voice-activated computational thinking educational robot developed in this research. The Zenbo will perform the specified actions to complete the problem-solving task. This research uses BNF grammar to define the syntax of the V language. The following sections will explain the built-in actions of Zenbo and the designed V language in order.

3.1 Built-in Features of Zenbo

Zenbo's built-in features are divided into two parts, speech, and action. Speech-related features include listening and speaking, while action-related features include movement, rotation, and head motion. Details are as follows:

1. Listening: When Zenbo is in listening mode, the user can give Zenbo a voice command, and Zenbo will analyze the voice command spoken by the user and take subsequent actions based on the command.

2. Speaking: The speaking function allows Zenbo to speak the text users specify, providing feedback or informing the user of the content of an error message.

3. Movement: The movement function allows Zenbo to move forward or backward in a longitudinal motion.

4. Rotation: The rotation function allows Zenbo to rotate to the left or right.

5. Head Motion: The head motion function allows Zenbo to perform head motions, such as tilting, turning left or right, or stopping head sway.

As the educational robot for teaching computational thinking developed in this study uses Zenbo as a platform, the verbal programming language designed only focuses on Zenbo's built-in actions. Corresponding statements have been designed to enable students to instruct Zenbo to perform various actions and complete assigned problem-solving tasks.

3.2 V Language

This research adopts BNF (Backus Normal Form) to define the syntax of the V language. BNF is widely used to represent syntax in programming languages, instruction sets, and communication protocols, [40]. Most programming languages adopt BNF grammar to define their syntax. The notations of the BNF grammar used in this study are listed in Table 1.

Table 1. Notations of the BNF grammar

Notation	**Meaning**
::=	Defined as
\| | choice |
< > | Non-terminal |

For example, "<seq::=<move>|<dance>|<turn>" means that <seq> is defined as either <move>, <dance>, or <turn>, not as <move><dance>, <dance> <turn>, or a combination of <move><dance> <turn>. The form of <move>, <dance>, and <turn> are non-terminal symbols and need to be defined separately.

This study designs the syntax of the V language based on the actions that the Zenbo robot can execute. The main purpose is to allow students to use verbal commands to control the educational robot developed for computational thinking using syntax that conforms to the V language so that Zenbo can perform the corresponding actions to complete the problem-solving tasks assigned by the teacher. To reduce the cognitive burden of young children and avoid the frustration and decreased learning motivation that can result from frequent syntax errors, this study considered the problem-solving requirements designed by code.org for young children when designing the syntax of the V language. Three main instructions are provided in V language: sequential, for loop, and while loop. In addition to setting parameters, each statement can be nested with other statements, allowing the user to design complete program codes using the V language. The BNF Grammar of the V language is shown in Table 2.

Table 2. The BNF Grammar of the V language

P1	<program> ::= <statements>
P2	<statements > ::= <statement>|<statement>< statements >
P3	<statement> ::= <action>|<repeat>|<while>
P4	<action> ::= FORWARD|TURN-LEFT| TURN-RIGHT
P5	<repeat> ::= <for>|<while>
P6	<for> ::= REPEAT <number> TIMES EXECUTE <statements> END
P7	<number> ::= 1|2|3|4|5
P8	<while> ::= REPEAT < statements >UNTIL STOP

Sequential structure refers to the program structure in which statements are sequentially executed in the order they are written in the code. The program statements were simplified according to the needs of young children to accomplish their tasks, and the complexity of the syntax was reduced by removing parameters that could be adjusted for action range and replacing them with fixed parameters. The sequential instructions in V language include "forward," "turn left," and "turn right." Each forward movement is 0.6 meters, and the left and right turn angles are 90 degrees. If the "forward" command is issued, Zenbo will walk forward 0.6 meters; when the "turn-left" command is issued, Zenbo will turn left 90 degrees; and when the "turn-right" command is issued, Zenbo will turn right 90 degrees. However, if the command "forward 3" is issued, it will result in a syntax error, and an error message will appear.

Loop structure refers to the program structure in which a block of statements can be repeatedly executed. Usually, a loop condition is defined in the loop structure, and the loop structure will only continue to execute as long as the loop condition evaluates to true. This study's loop structure of V language includes “for-loop” and “while loop.”

The syntax format of the for-loop instruction is: repeat (number) times execute (statements) end. The number after "repeat" is limited to 1 to 5, and

the statements after "execute" can be any combination of instructions. For example, if the instruction "repeat 2 times execute forward, turn left end" is issued, Zenbo will repeat the action of forward 0.6 meters and then turn left 90 degrees 2 times. But if the command "repeat end" is given, it will result in a syntax error, and an error message will appear.

The syntax format of the while loop instruction is repeated (statements) until stop, where the statements after "repeat" can be any combination of instructions. For example, if the instruction "repeat forward until stop" is issued, Zenbo will keep moving forward 0.6 meters each time until it encounters an obstacle and stops. However, if the instruction "repeat until stop" is issued, it will result in a syntax error, and an error message will appear.

4 Development of the Voice-activated Educational Robot

To apply educational robots to young students' learning of computational thinking, this study develops a voice-activated computational thinking educational robot based on V language. We use JAVA, HTML, CSS, JavaScript, jQuery, and PHP as the programming language for system development, Bootstrap as the frontend framework, and Android Studio as the development environment. Zenbo SDK toolkit and phpMyAdmin database management tool are also used to develop and deploy the educational robot system. The system architecture of the educational robot developed in this study is shown in Fig. 1.

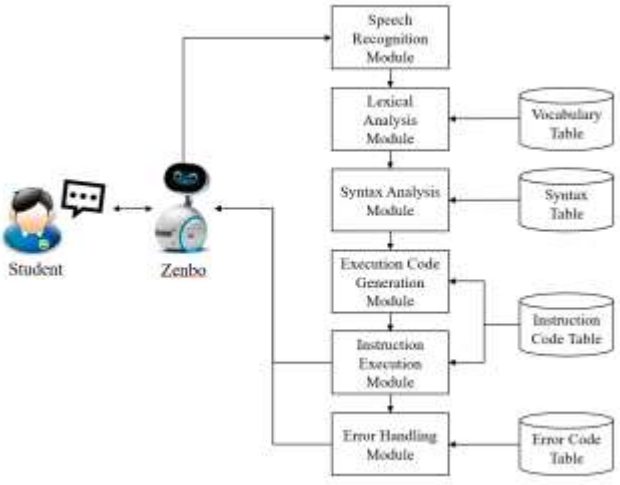

Fig. 1: System Architecture

The system includes 4 data tables: vocabulary table, syntax table, instruction code table, and error code table, as well as six functional modules: speech recognition module, lexical analysis module, syntax analysis module, execution code generation module, instruction execution module, and error handling module.

The vocabulary table stores the legal vocabulary in the syntax and its corresponding vocabulary code. The syntax table keeps the BNF-Grammar syntax of the V language. The instruction code table keeps the instruction codes for the robot to perform various actions. The error code table stores the error codes and corresponding error messages for syntax errors.

The speech recognition module converts verbal instructions into textual instructions when given to the Zenbo robot. The speech recognition module utilizes the Google Speech Recognizer API to transform voice commands into text by calling Google's speech recognition engine. The text strings are then delivered to the lexical analysis module for processing. The lexical analysis module will tokenize the text string and check whether the token generated after tokenization is legitimate. After lexical analysis, if the token is legal, it is coded according to the lexical list and passed to the syntax analysis module for further syntax analysis. If it is illegal, it is sent back and the user is informed of the lexical error by the Zenbo robot. The syntax analysis module analyzes the lexical code sequence obtained from the lexical analysis module according to the syntax table. If the analysis result matches the syntax, the execution code or error code will be generated later using the execution code generation module. The execution code generation module generates execution codes or error codes based on the syntax analysis results. If the analysis results in a correct syntax, the word code is converted into an execution code sequence and then sent to the instruction execution module for instruction execution. If the analysis results in a syntax error, a negative error code is generated according to the error so that the error handling module can report the error message. Based on the execution code generated by the execution code generation module, the instruction execution module calls the corresponding API in the Zenbo SDK to let Zenbo perform the action that matches the user's instruction. Suppose there is an error occurred in the above modules. In that case, the error handling module will convert the error code into corresponding error messages and calls the speak API in the Zenbo SDK to let the user know where the instruction is wrong and give correct examples.

The overall process of the system is as follows:

Step 1: When a student issues a verbal instruction to Zenbo, the speech recognition module will convert the received verbal instruction into a textual instruction and send it to the lexical analysis module.

Step 2: The lexical analysis module separates the textual instruction into word segments, converts each into lexical code, and then sends the lexical code sequence to the syntax analysis module. Suppose any error is encountered during lexical analysis, such as invalid vocabulary used in the instruction. In that case, the corresponding error codes will be sent to the error handling module to prompt suitable error messages to the user.

Step 3: The syntax analysis module analyses the lexical code sequence according to the syntax table to see if it complies with the grammar of V language. If the grammar is complied with, the lexical code sequence will be passed to the execution code generation module. If not, the corresponding error codes will be sent to the error handling module to prompt suitable error messages to the user.

Step 4: The execution code generation module generates an execution code according to the instruction code table and sends it to the instruction execution module.

Step 5: The instruction execution module calls the corresponding API in the Zenbo SDK based on the executable code to make Zenbo perform actions that comply with the user's instructions.

Step 6: If an error occurs in the above steps, the error handling module will convert the error code received into the error message according to the error code table and calls the speak API in the Zenbo SDK to let Zenbo speak out the error message, allowing the user to know where the error lies in the instruction.

The scenario in which the educational robot developed in this research is applied in the learning activity of computational thinking education is shown in Fig. 2.

5 Conclusions and Future Works

This study developed a voice-activated computational thinking educational robot for young children, using the ASUS Zenbo robot as the platform. Based on the Zenbo robot's actions, a verbal programming language called V language was defined by BNF grammar. While the learner speaks an instruction in V language to the developed educational robot, the robot will perform the designated actions to execute the instruction. The problem-solving tasks for cultivating computational thinking can be completed through a sequence of verbal instructions issued by the learner.

This study attempts to develop verbal programming robots for fostering computational thinking in younger children. Compared to graphical programming environments, Verbal programming robots incorporate the features of social robots, and children can communicate and interact with the robots using conversations. As a result, this learning approach is more innovative than other programming robots in cultivating children's computational thinking, freeing them from programming syntax and allowing them to focus more on problem-solving. The developed educational robot is expected to improve the effectiveness of computational thinking cultivation.

Nevertheless, whether verbal robots can facilitate the development of computational thinking in children needs to be further investigated. Therefore, this study will design and integrate a series of field experiments to investigate the effects of the developed verbal robot on the computational thinking of younger children.

Fig. 2: The scenarios of computational thinking learning activity incorporating the developed educational robot

References:

[1] Nouri, J., Zhang, L., Mannila, L., & Norén, E., Development of computational thinking, *digital competence and 21st century skills when learning programming in K-9*, Education Inquiry, vol. 11, no. 1, 2020, pp. 1-17.

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

[3] Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A., Computational thinking and tinkering: Exploration of an early childhood robotics curriculum, *Computers & Education*, vol. 72, 72, 2014, pp. 145-157.

[4] Boticki, I., Pivalica, D., & Seow, P., The use of computational thinking concepts in early primary school, *Science*, vol. 2, 2018,

[5] Lai, Y. H., Chen, S. Y., Lai, C. F., Chang, Y. C., & Su, Y. S., Study on enhancing AIoT computational thinking skills by plot image-based VR, *Interactive Learning Environments*, vol. 29, no. 3, 2021, pp. 482–495.

[6] Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E., The Scratch programming language and environment, *ACM Transactions on Computing Education (TOCE)*, vol. 10, no. 4, 2010, pp. 1-15.

[7] Nouri, J., Zhang, L., Mannila, L., & Noren, E., Development of computational thinking, digital competence and 21st century skills when learning programming in K-9, *Education Inquiry*, vol. 11, no. 1, 2020, pp. 1-17.

[8] Montiel, H., & Gomez-Zermeño, M. G., Educational challenges for computational thinking in k–12 education: A systematic literature review of "scratch" as an innovative programming tool, *Computers*, vol. 10, no. 6, 2021, pp. 69.

[9] Sun, L., Hu, L., & Zhou, D., Which way of design programming activities is more effective to promote K-12 students' computational thinking skills? A meta-analysis, *Journal of Computer Assisted Learning*, vol. 37, 2021, pp. 1048-1062.

[10] Shadiev, R., Hwang, W. Y., Yeh, S. C., Yang, S. J. H., Wang, J. L., Han, L., & Hsu, G. L., Effects of unidirectional vs. reciprocal teaching strategies on Web-based computer programming learning, *Journal of Educational Computing Research*, vol. 50, no. 1, 2014, pp. 67-95.

[11] Piaget, J., & Inhelder, B., *The psychology of the child*. New York: Basic Books,1969

[12] Benitti, F. B. V., Exploring the educational potential of robotics in schools: A systematic review, *Computers & Education,* vol. 58, no. 3, 2012, pp. 978-988.

[13] Relkin, E., de Ruiter, L. E., & Bers, M. U., Learning to code and the acquisition of computational thinking by young children, *Computers & Education*, vol. 169, 2021, pp. 1-15.

[14] Chevalier, M., Riedo, F., & Mondada, F., Pedagogical uses of thymio II: How do teachers perceive educational robots in formal education?, *IEEE Robotics & Automation Magazine*, vol. 23, no. 2, 2016, pp. 16-23.

[15] Chalmers, C., Robotics and computational thinking in primary school, *International Journal of Child-Computer Interaction*, vol. 17, 2018, pp. 93-100.

[16] Qu, J. R., & Fok, P. K., Cultivating students' computational thinking through student-robot interactions in robotics education, *International Journal of Technology and Design Education,* vol. 32, 2021, pp. 1983-2002.

[17] Çakıroğlu, Ü., & Mumcu, S., Focus-fight-finalize (3F): problem-solving steps extracted from behavioral patterns in block based programming, *Journal of Educational Computing Research*, vol. 58, no. 7, 2020, pp. 1279-1310.

[18] Bloom, L., & Lahey, M., *Language development and language disorders*, New York: John Wiley,1978

[19] Moore, T. J., Brophy, S. P., Tank, K. M., Lopez, R. D., Johnston, A. C., Hynes, M. M., & Gajdzik, E., Multiple Representations in Computational Thinking Tasks: A Clinical Study of Second-Grade Students, *Journal of Science Education and Technology*, vol. 29, no. 1, 2020, pp. 19-34.

[20] Kong, S. C., Chiu, M. M., & Lai, M., A study of primary school students' interest, collaboration attitude, and programming empowerment in computational thinking education, *Computers & Education*, vol. 127, 2018, pp. 178-189.

[21] Rich, P. J., Browning, S. F., Perkins, M., Shoop, T., Yoshikawa, E., & Belikov, O. M., Coding in K-8: International Trends in Teaching Elementary/Primary Computing, *TechTrends*, vol. 63, no. 3, 2018, pp. 311-329.

[22] Hsu, T. C., Chang, S. C., & Hung, Y. T. How to learn and how to teach computational thinking: Suggestions based on a review of the literature, *Computers & Education*, vol. 126, 2018, pp. 296-310.

[23] Google (2015). Exploring Computational Thinking. Retrieved from https://www.google.

com/edu/resources/programs/exploring-computational-thinking/
[24] Grover, S., & Pea, R. Computational thinking in K–12: A review of the state of the field, *Educational Researcher*, vol. 42, no. 1, 2013, pp. 38-43.
[25] Cheng, L. C., Li, W., & Tseng, J. C. Effects of an automated programming assessment system on the learning performances of experienced and novice learners, *Interactive Learning Environments*, 2021, pp. 1-17.
[26] Kalelioğlu, F. A new way of teaching programming skills to K-12 students: Code.org, *Computers in Human Behavior*, vol. 52, 2015, pp. 200-210.
[27] Chang, C. W., Lee, J. H., Wang, C. Y., & Chen, G. D. Improving the authentic learning experience by integrating robots into the mixed-reality environment, *Computers & Education,* vol. 55, no. 4, 2010, pp. 1572-1578.
[28] Atman Uslu, N., Yavuz, G. Ö., & Koçak Usluel, Y. A systematic review study on educational robotics and robots, *Interactive Learning Environments,* 2022, pp. 1-25.
[29] Xia, L., & Zhong, B. A systematic review of teaching and learning robotics content knowledge in K–12, Computers & Education, vol. 127, 2018, pp. 267-282.
[30] Sun, L., & Zhou, D. Effective instruction conditions for educational robotics to develop programming ability of K-12 students: A meta-analysis, *Journal of Computer Assisted Learning*, vol. 39, no. 2, 2023, pp. 380-398.
[31] Rusk, N., Resnick, M., Berg, R., & Pezalla-Granlund, M. New pathways into robotics: Strategies for broadening participation, *Journal of Science Education and Technology*, vol. 17, no. 1, 2008, pp. 59-69.
[32] Chin, K. Y., Hong, Z. W., & Chen, Y. L. Impact of using an educational robot-based learning system on students' motivation in elementary education, *IEEE Transactions on Learning Technologies*, vol. 7, no. 4, 2014, pp. 333-345.
[33] Fagin, B., & Merkle, L. Measuring the effectiveness of robots in teaching computer science, *ACM SIGCSE Bulletin,* vol. 35, no. 1, 2003, pp. 307-311.
[34] Winslow, L. E. Programming pedagogy—a psychological overview, *ACM SIGCSE Bulletin,* vol. 28, no. 3, 1996, pp. 17-22.
[35] Brito, M. A., & de Sá-Soares, F. Assessment frequency in introductory computer programming disciplines, *Computers in Human Behavior*, vol. 30, 2014, pp. 623-628.
[36] Lye, S. Y., & Koh, J. H. L. Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior,* vol. 41, 2014, pp. 51-61.
[37] Sáez-López, J.-M., Román-González, M., & Vázquez-Cano, E. Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools, *Computers & Education*, vol. 97, 2016, pp. 129-141.
[38] Jiang, B., & Li, Z. X. Effect of Scratch on computational thinking skills of Chinese primary school students, *Journal of Computers in Education,* vol. 8, no. 4, 2021, pp. 505-525.
[39] Zhao, L., Liu, X., Wang, C., & Su, Y. S. Effect of different mind mapping approaches on primary school students' computational thinking skills during visual programming learning, *Computers & Education*, vol. 181, 2022
[40] Deng, Y., Zhang, S., & Huang, B. A concise BNF syntax for OpenFlow, *IEEE Communications Letters*, vol. 21, no. 1, 2017, pp. 196-199.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

All authors contributed to the study's conception and design. [Judy C.R. Tseng] and [Wei Li] performed material preparation and data collection. The first draft of the manuscript was written by [Judy C.R. Tseng] and [Wei Li], and all the authors have revised the manuscript. All authors read and approved the final manuscript.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

This study was partly supported by the National Science and Technology Council of Taiwan under grant numbers MOST 109-2511-H-216-001-MY3 and the Ministry of Education of Humanities and Social Science Project of the People's Republic of China [21YJA880027]. The authors would like to thank Xin Ci Wen for her assistance in developing the system.

Conflict of Interest

The authors declare no conflicts of interest.