SEBHA UNIVERSITY JOURNAL OF PURE & APPLIED SCIENCES VOL.22 NO. 3 2023
DOI: 10.51984/JOPAS.V22I3.2835

مجلة جامعة سبها للعلوم البحتة والتطبيقية
Sebha University Journal of Pure & Applied Sciences

Journal homepage: www.sebhau.edu.ly/journal/index.php/jopas

Unlocking the Potential of Programming Education: Enhancing Conceptual Understanding and Student Engagement with Sphero SPRK Robot

*Ibrahim Nnass[1], Juan-Carlos Muñoz[2], Michael A. Cowling[3], Roger Hadgraft[4]

[1]College of Technical Sciences, Ministry of Technical & Vocational Education, State of Libya
[2]ICT Lecturer, School of Engineering and Technology, CQUniversity, 160 Ann St Brisbane, Queensland 4000, Australia
[3]Mobile Computing & Apps CQUniversity, School of Engineering and Technology, 160 Ann St Brisbane, Queensland 4000, Australia
[4]Educational Innovation & Research, Faculty of Engineering & Information Technology University of Technology Sydney, Broadway, NSW 2007, Australia

Keywords:
Computer programming
Learning programming
Programming language
Robotic assistant
Interactive learning
Novice programmers

ABSTRACT

This research investigates the use of the 'Sphero SPRK+' robot as a programming assistant to enhance conceptual understanding and engagement in programming education. The research addresses the challenges of teaching programming and underlines the significance of students understanding programming structures, semantics, syntax and planning. The Sphero robot offers multiple programming options, including graphic blocks, a pictorial command line, and text commands, catering to diverse learning styles. The approach of the study is based on a cognitive learning model, with the Sphero robot providing immediate feedback throughout practising. The experiment incorporates two groups: one using Blocks and the other using JavaScript code with Sphero's assistance. Results suggest that using Sphero SPRK robot enhances programming learning outcomes, particularly for novice learners. Participants express satisfaction with the engaging and practical learning experience, favouring the ease of Blocks for learning programming logic. The study provides significance for educational settings and emphasises the value of robotics and block-based programming in promoting algorithmic thinking and computational skills.

توسيع إمكانات تعليم البرمجة: تحسين فهم مبادئ البرمجة للطلاب باستخدام الروبوت Sphero SPRK

*إبراهيم النعاس[1] و خوان كارلوس[2] و مايكل أ. كاولينج[3] و روجر هادجرافت[4]

[1]العلوم التقنية، وزارة التعليم التقني، دولة ليبيا.
[2]تكنولوجيا المعلومات والاتصالات، كلية الهندسة والتكنولوجيا، CQUniversity، أستراليا.
[3]الحوسبة المتنقلة والتطبيقات، كلية الهندسة والتكنولوجيا، CQUniversity، أستراليا.
[4] الابتكار التعليمي والبحوث، كلية الهندسة وتكنولوجيا المعلومات، University of Technology Sydney ، أستراليا.

الكلمات المفتاحية:
برمجة الحاسوب
تعليم البرمجة
لغة برمجة
مساعد آلي للتعليم التفاعلي
المبرمجين المبتدئين

الملخص

هذا البحث يبحث في استخدام الاداة/الروبوت "Sphero SPRK+" كمساعد برمجي للمساعدة ولتعزيز فهم البرمجة وايضا، المشاركة في تعليم البرمجة. كما يتناول البحث تحديات تدريس البرمجة ويؤكد على أهمية فهم الطلاب لهياكل البرمجة كدلالاتها وبناء الجملة والتخطيط. يوفر Sphero خيارات برمجة متعددة، بما في ذلك الكتل الرسومية وأسطر الأوامر المصورة والأوامر النصية، مما يلبي أنماط التعلم المتنوعة. يعتمد منهج الدراسة على نموذج التعلم المعرفي، حيث يقدم الروبوت Sphero ردود فعل فورية طوال التدريب. تتضمن التجربة مجموعتين: واحدة تستخدم Blocks والأخرى تستخدم أكود JavaScript وذلك بمساعدة Sphero. وتشير النتائج إلى أن استخدام Sphero SPRK+ يعزز نتائج تعلم البرمجة، وخاصة بالنسبة للمتعلمين المبتدئين. كما اعرب المشاركون عن رضاهم بخوض هذه التجربة التعليمية الجذابة والعملية،

*Corresponding author:
E-mail addresses: Alnnass76@yahoo.com, (J-C Muñoz) j.munoz@cqu.edu.au, (M. A. Cowling) m.cowling@cqu.edu.au , (R. Hadgraft) Roger.Hadgraft@uts.edu.au

Article History : Received 24 June 2023 - Received in revised form 23 September 2023 - Accepted 02 October 2023

وصرحوا بانهم يفضلون الكتل الرسومية لسهولة تعلم منطق البرمجة من خلالها. توفر الدراسة إعدادات تعليمية مهمة وتؤكد على اهمية استخدام الروبوتات في تعلم البرمجة المبنية على الكتل لأهميتها في تعزيز التفكير الخوارزمي والمهارات الحسابية.

Introduction

Teaching programming poses significant challenges, irrespective of the programming language, strategy, or approach used [1, 2, 3]. However, computer programming offers valuable skills with extensive career opportunities and potential for personal and professional growth [4]. One of the primary difficulties in teaching programming lies in helping students understand programming structures, semantics, syntax, and planning [5, 6]. The critical conceptual understanding, encompassing logical context, steps, and operations, often remains obscured during programming education [4, 5]. Additionally, learners grapple with building structured solutions based on rigid syntax and confusing command names, leading to inflexible examples [3]. Understanding classes, objects, and program design becomes a challenge as students struggle to link and apply these concepts effectively [7].

To address these challenges, innovative approaches are necessary to assist novices in comprehending computational actions. One such solution is the use of a programming robot assistant like 'Sphero,' a programmable robotic ball controlled via smartphone and tablet [8]. Sphero offers three programming options, incorporating puzzle-like graphic blocks, a pictorial command line, and text commands, catering to various learning preferences and skill levels. By leveraging Sphero's interactive capabilities, students receive guidance and immediate feedback during programming practice.

This research explores the implementation and results of an educational robotic activity utilizing Sphero block programming to overcome the challenges of traditional programming education. The research aims to answer two key research questions: "RQ1: How does the use of robot technology and block programming impact students' interest in learning programming?" and "RQ2: How does the incorporation of this learning method assist students in enhancing their conceptual understanding of programming?" Through this investigation, the article sheds light on the potential of Sphero SPRK+ as an effective tool for fostering engagement and improving programming conceptualization.

The subsequent sections of this article delve deeper into the background and related work, providing insights into the difficulties faced by students learning programming and the existing research on teaching programming languages. The methodology section elucidates the cognitive learning model employed and outlines the utilisation of Sphero as a programming assistant. The experimental results section presents the outcomes of the study, showcasing the impact of Sphero SPRK+ on programming learning outcomes. Additionally, the discussion section analyses the results in light of the literature review, highlighting the significance of educational robotics and block-based programming in addressing learning barriers and enhancing computational thinking skills.

The following sections will provide a foundation for the significance of the research, exploring the challenges of programming education and the importance of problem-solving strategies and computational thinking skills. The role of robotics in providing tangible and hands-on experiences to make programming more concrete and engaging will be explored, leading to the introduction of Sphero SPRK as an educational robotics solution to foster improved learning outcomes in programming. The first section, "Background and Related Work," establishes the context by discussing the challenges faced in programming education and emphasising the significance of problem-solving strategies and computational thinking skills. It highlights the relevance of robotics in providing hands-on experiences to enhance programming learning. In the subsequent section, Sphero SPRK is introduced as a versatile programmable robotic tool emphasising the user-friendly nature of the Sphero EDU app, which combines visual blocks and Scratch-based coding, making it an ideal assistant programming tool for novice learners. The third section, Methodology, outlines the research approach, which involves the implementation of a cognitive learning model with a programming robot instructor present during practice. In the experimental Results section, the outcomes of the study are presented, demonstrating the positive impact of Sphero SPRK as an assistant programming tool. The research emphasises the value of Sphero SPRK as a cutting-edge method for enhancing programming education through the logical evolution of these portions. The Sphero SPRK platform makes learning computational thinking and problem-solving easier for students and equips them with the skills they need to succeed in programming.

RELATED WORK

One of the many obstacles faced by first-year students in information technology (IT) classes is being exposed to a subject they have no prior knowledge on nor studied in their previous learning [7]. Such an issue appears challenging In programming, when students encounter a number of concepts, computer declarations and components that include obtaining set skills to be developed in parallel while learning and practicing programming [9]. Understanding how programmes can be performed by putting together symbols that lead to computational activities and dealing with demanding syntax can also frustrate and discourage learners [10].

According to [5], the real challenge lies in the student transferring this learning syntax to solving a programming problem. Another challenge is to learn the complexity of concept details that implies logical reasoning, understanding abstract terms and variables that do not relate to real life [4]. Computer programing in particular is a concern for many researchers and academics. For example, [11] refers to the type of methodology, the language and the topics that should be used to teach programming. According to [4], the most used programming language in academia is Java, covered by object oriented concepts.

In teaching programming languages using traditional approaches, academics emphasize the importance of the language' syntax and semantics rather than the problem solving strategy itself to address programming problems [9]. For instance, [12] argued that novice programmers know the syntax and semantics of individual statements but do not know how to combine this feature into valid programs. [13] indicate that introductory programming books primarily present particular language information and knowledge.

Programming has consistently been demonstrated in research to improve children's algorithmic thinking [8]. Programming plays an important role in improving algorithmic knowledge by making underlying mathematical assumptions explicit and concrete [8,14]. When students translate problem-solving into software programs, they develop a deeper comprehension of algorithmic principles, resulting in improved mathematical problem-solving abilities [15]. Moreover, the understanding of algorithmic thinking contributes to the broader development of computational thinking skills [28, 24]

Robotic environments provide tangible and hands-on experiences, making programming more concrete (15). By engaging with robots, students gain a deeper understanding of programming concepts, further strengthening their computational thinking abilities.

However, only a small number of sources applied problem solving strategies linked to difficult concepts [16]. According to [16], the analysis of the problem statement within the programming training provides a better overview of the problem and therefore a conception of the programming solution. The problem-solving strategy can add another layer for problem interpretation and presentation when breaking down the complexity from problem statement through problem-solving phase to finally produce code [3].

Relating to this concept of problem solving, computational thinking is a fundamental skill for students and individuals' analytical ability alongside reading, writing and critical thinking that allows students to explore, dive and find information to understand complex concepts [17]. Using effective forms of communication in different settings is

the foundation of modern life. Managing, cognitive activities and problem solving in a range of contexts is highly rewarded in many fields including computing disciplines [18, 19].

Robotic educational applications and its features are mainly dedicated to computer simulations, robot programing, robot construction and artificial intelligence [8]. However, alternative solutions that embrace educational robotics exist [18, 20] and can enhance problem-solving and computational skills as well as engaging learners with diverse interest and learning styles.

Robotics can be used as a teaching assistant tool to assist students to assimilate complex scenarios as it engages them and therefore to increase their computational thinking skills [21, 19]. To strengthen the learning of complex or abstract topics, educational robotics has been introduced in many educational institutions as a novel learning environment to maximize advanced problem-solving skills [6, 22]. Consequently, the addition of learning methods connected to learning instructions using robots facilitates teamwork, enhances critical thinking, scaffolding learning and conceptual understanding [23].

A ROBOT-PROGRAMMING ASSISTANT: SPHERO SPRK

Sphero is a mobile app controlled robotic ball capable of moving in numerous directions [24] (see Figure 1). Users of Sphero have three options on programming and directing their robot. First choice is block programming, which uses a drag and drop interface. Draw programming where users can draw lines to program the robot, and text based code, for more advanced methods of programming ("programming with Sphero," 2017).

Figure 1: Sphero SPRK robot technology

One of the Sphero features is that it can convert blocks into a text JavaScript program (see Fig 2.).

The visual programming environment utilizes the Sphero EDU app, powered by Scratch, to program robots. The platform involves "blocks" that contain snippets of code, which users connect using the app. Users can manipulate these blocks visually in the blocks view editor, where colored graphical blocks represent code. Additionally, the editor provides a source view that presents a textual representation of the program for editing purposes. This combination of a visual editor and the Sphero EDU app offers a versatile programming experience, combining visual blocks and Scratch-based coding for robot programming.

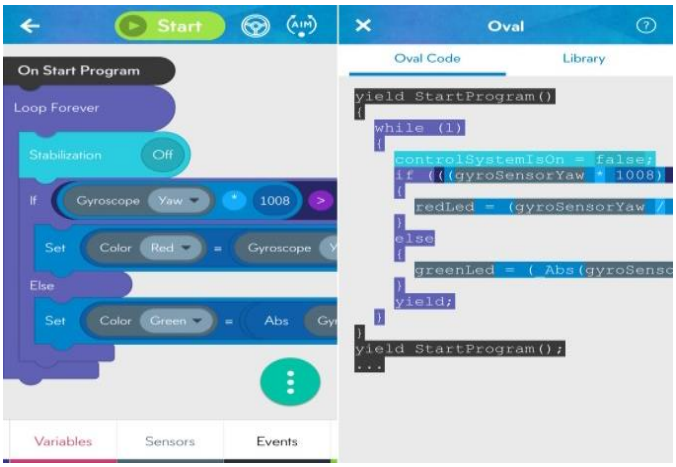

Figure 2: Text code associated with a block program.

As programming abilities become more valuable and in demand, the learnability of end-user programming skills becomes increasingly important. However, much research on learning barriers in programming systems has concentrated on languages, ignoring possible barriers in the environment and related libraries. According to [25], there are six categories of learning barriers: design, selection, coordination, use, comprehension, and information. These barriers motivate a new metaphor that takes a more learner-centric approach to programming [15].

Learning Barrier	Description
Design Barriers	Inherent cognitive difficulties in solving programming problems, making it hard to visualize and conceive solutions.
Selection Barriers	Challenges in determining which programming interfaces are capable of achieving a specific behavior.
Coordination Barriers	Limitations in combining programming interfaces to achieve complex behaviors; learners struggle with coordination.
Use Barriers	Difficulties in understanding how to use programming interfaces and their effects, including syntactical and parameter-related challenges.
Understanding Barriers	Challenges in evaluating program behavior, including difficulties with interpreting error messages and unexpected outcomes.
Information Barriers	Difficulties in acquiring information about a program's internal behavior, such as variable values or function calls, due to unclear tools or how to use them effectively.

These barriers highlight the various challenges learners face during programming education, ranging from conceptualization to implementation, coordination, and understanding of code behavior.

METHODOLOGY

The proposed methodology for this study is based on a cognitive learning model where a programming robot instructor is utilised during the practise of programming to provide immediate feedback based on user actions. The approach aims to create an engaging learning experience for students by enabling them to build programmes using blocks, similar to playing with LEGO, and benefit from the physical aspects of using a robotics assistant like Sphero. The study involved eight participants, including a mix of students from the host university and members of the public who were either novice programmers or interested in learning programming. The teaching activity was conducted face-to-face in a computer lab, utilising Microsoft PowerPoint and a whiteboard for clarifications. Additionally, Apple devices (iPads and iPhones) were provided to participants without electronic devices to interact with the Sphero SPRK robot during the experiment, alongside Sphero SPRK+ with the Sphero Edu App. The participants were divided into two groups: Group A and Group B. Initially, all participants were given an overview of programming and introduced to the Sphero SPRK robot as an assistant programming tool. They were familiarised with how to use the Sphero SPRK robot to learn programming through blocks and codes, and examples were presented to demonstrate the capabilities of the Sphero SPRK robot technology. Subsequently, the two groups worked separately. Group A's task was to learn to build their own programme using blocks and then test it on the Sphero SPRK robot. On the other hand, Group B was assigned to learn to build their own programme using JavaScript code and test it on the Sphero SPRK robot. To assess the learning outcomes, a quiz was conducted. Group A was tested using Blocks with the assistance of the Sphero SPRK robot, while Group B was tested on JavaScript code with the assistance of the same robot. Additionally, qualitative data on participants' perceptions of the learning process was collected through a survey. The experiment's primary objectives were twofold: to understand the programming difficulties faced by participants and to gauge the effectiveness of Sphero SPRK robot technology as an assistant programming tool in improving participants' programming learning outcomes. By applying this methodology, the study sought to explore the potential benefits of utilising Sphero SPRK in programming education and its impact on novice programmers' learning experiences. The combination of the cognitive learning model and the hands-on experience with Sphero SPRK aimed to provide insights into the effectiveness of educational robotics as a tool for enhancing computational thinking skills and problem-solving abilities in the context of programming education.

Here is summarizing the main steps of the proposed methodology:

Step	Description
1. Cognitive Model	Adoption of a cognitive learning model
2. Participant	Selection of eight participants
3. Experimental Setup	Conducting the teaching activity in a computer lab
4.Introduction to Sphero SPRK	Familiarizing participants with the robot
5. Group Division	Dividing participants into Group A and Group B
6. Programming Tasks	Assigning programming tasks to each group
7. Assessment	Conducting a quiz to evaluate learning outcomes
8. Data Collection	Gathering both quantitative and qualitative data
9. Analysis	Analysing the collected data
10. Experiment Discussion	Discussing the results and implications

EXPERIMENTAL RESULTS

This study experiential learning activity revealed that programming topics were more engaging and easily understood by novice learners compared to traditional explanations. The proposed methodology combines hands-on learning with Sphero SPRK and a cognitive learning model to explore the impact of educational robotics on novices' programming skills. By using Sphero SPRK as an assistant programming tool, the research aims to enhance computational thinking skills and address programming challenges. The results, presented in a tabular form, provide valuable insights into students' satisfaction, experiences, and suggestions with Sphero SPRK. This innovative approach holds promising potential to revolutionize programming education and empower learners with an interactive and effective platform for skill development.

Table 1. Specific programming issues facing participants

Understanding issues	No. of participants
Loops statements	5
Condition statements	4
Syntax	4
Semantics	2
Data types	2

Table 2 summarizes the difficulties that participants faced in basic programming principles (loops statements, condition statements & syntax, semantics & data types).

Table 2. Participants' Comments

Classification Comments	Participants Comments
Participants satisfied	Sphero App and Sphero robot are amazing Sphero technology is interesting
Advantages use Sphero SPRK	Sphero Edu App and Sphero robot are very easy to use for learning programming especially learning programming logic Sphero robot is fun and interactive way to start learn about programming Learnt something new about programming Sphero robot is so practical method to learn and explain programming concepts It makes me want to go back and study programming Sphero robot is way to understand the basic principles of programming and to appreciate the importance of writing a correct code.
Disadvantages use Sphero SPRK	The Blocks technique is restrictive; for instance, you can not modify your code at any point when using Blocks. There were connectivity issues with some mobile devices
Frustrations with the experience	There is no enough time to play with Sphero robot Writing code in a cell phone is painful
Suggestions for improvements	Explanation or help text for each option For beginners, it would good idea to explain (variables, loops and methods) Not to primarily rely on Sphero because this could lead beginners to assume that programming is less complex than it actually is.
Programming skills needed	Programming logic understanding Principle programming concepts
Experience with Sphero SPRK robot technology	It is something that really could help beginners to learn programming in an easy way and very attractive not boring In contrast to codes, blocks are significantly simpler to understand, and learners should have no concerns about syntax and semantics. Blocks is easy to associate a concept with an action like dragging a box and seeing how the robot changed in real time Sphero robot and Sphero Edu App looks easy to understand and then easy to manage but first you need to understand how to deal with it Blocks method assists beginners in understanding what they are attempting to accomplish, and if there is a flaw in their programme, they can quickly identify it. Drag and drop functionality in the blocks technique eliminates the need for repetitive coding. The Sphero robot makes it achievable for anyone to learn programming on the go with greater flexibility. The technique is enjoyable and gives motivation to learn Block use does not require programming experience is the fastest and easiest in correct program design

The results of the study using Sphero SPRK as an assistant programming tool demonstrated that programming topics became more interesting and engaging for novice learners compared to traditional conceptual explanations. Participants expressed high satisfaction with the Sphero App and robot, emphasising their ease of use and interactive nature. The advantages of using Sphero SPRK included its practicality in explaining programming concepts and its ability to motivate learners to study programming further. The Blocks method offered by Sphero SPRK was well-received as a user-friendly approach, allowing novices to understand programming logic without worrying about syntax and semantics.

However, some limitations were identified, such as the restriction on editing code using the Blocks method and connectivity issues with certain mobile devices. Participants also emphasised the need to balance the use of Sphero with other learning methods to avoid oversimplifying programming concepts. The feedback suggested that a more comprehensive explanation of variables, loops, and methods within the Sphero Edu App could enhance the learning experience, especially for beginners.

The results highlighted the importance of understanding programming logic and fundamental programming concepts, which were effectively facilitated by the Blocks method. Participants found the visual association of concepts with actions, such as dragging and dropping blocks, to be beneficial in understanding the flow and logic of the programming activity.

Overall, the experience with Sphero SPRK was deemed enjoyable and motivating, offering an attractive and easy way for beginners to learn programming. The study underscored the significance of incorporating tangible and hands-on learning experiences, like using Sphero SPRK, in programming education to make it accessible and engaging for learners of all levels. The positive outcomes demonstrated the potential of Sphero SPRK as an assistant programming tool to enhance learning outcomes and overcome challenges in programming education.

EXPERIMENT DISCUSSION

The research experiment focuses on programming language comprehension and program learning. The Sphero quiz was applied to two groups: Group A included those participants who used the Blocks with the assistance of the Sphero SPRK robot. Group B included those using JavaScript code with the assistance of the Sphero SPRK robot. The data was first analysed through the comments presented in Table 2.

The majority of participants reported that learning programming was straightforward when utilising the Sphero robot, especially when studying programming logic and learning innovative coding techniques. Students expressed great satisfaction with this experience and said that using blocks rather than text-based coding made it simpler for them to learn and comprehend programming logic and its

principles. This supports the finding by [12] that language is not a good way to start teaching programming and the indication by [14] that a focus should instead be put on critical thinking skills.

The experiment feedback also included some suggestions for improving the experiment design in terms of including a brief text explanation for each option in the Sphero Edu App and adding more explanation to some programming topics especially variables, loops, and methods. This supports the assertion by [26] that these are still important skills to learn, even when there is less focus on language. Indeed, this is supported by looking at the data in Table 1, which indicates that despite a downturn in the number of syntax and semantic problems that students encountered, there is still a desire to address problems relating to programming logic such as loop statements and condition statements.

Also, some of the participants reported that they should avoid relying only on Sphero because this may make novices think that programming is simpler than it is in reality. In a way, this supports [16] in indicating the importance of problem-solving skills to the problem and also [27], which indicates that despite the use of these novice languages, the most common industry language is still Java and that eventually students will need to learn this language to be competitive in the industry.

Specific qualitative comments also support the data found in the literature review. For instance, comments such as how useful the blocks are and how easy they are to understand support the assertions by [9] and [17] about using these tools to understand complex concepts.

Overall, in addition to the feedback obtained from participants, observation also generally showed that participants in Group A, who were using the Block method, advanced progressively in performing their given quiz programme, whereas those in Group B, who used JavaScript codes, struggled. Group B participants could not design programmes correctly because they faced difficulties with some programming concepts (such as Syntax and Semantics in JavaScript). It's clear from this that using Sphero SPRK robot technology, and particularly block programming, as an assistant programming tool can encourage novice programmers to learn programming with minimal frustration, as learning is more flexible and can be used on their own mobile devices. This supports the assertions by [18] and [19] about how these technologies help with learning and the value of these actions in helping learners gradually build a programme and have exposure to programming logic without engaging in semantic errors that may confuse them in the early stages of learning.

The semantic representation of the learning process with the Sphero SPRK robot and block-based programming:

1. Participants engage in the learning process by interacting with the Sphero robot.
2. They use the Sphero Edu App on their mobile devices to access programming tools.
3. Participants manipulate colored blocks representing code in the Blocks view.
4. Optionally, they can also use JavaScript code in the Source view.
5. The Sphero Edu App transfers the program instructions to the Sphero robot.
6. The Sphero robot executes the program instructions and provides feedback to the participants.
7. Participants actively learn programming language comprehension and program learning through this interaction.
8. Participants provide feedback on the learning experience, including suggestions for improvement.
9. Researchers analyze the comments and feedback from participants.
10. Researchers derive findings from the analyzed data and compare them with existing research through a literature review.
11. Recommendations are provided based on the findings and compared with prior research to enhance the learning process.
12. The learning process continues, and participants can iterate through the steps to further develop their programming skills.

This steps captures the key aspects of the learning process with the Sphero SPRK robot and block-based programming, focusing on the participants' interaction, feedback, and the role of researchers in analyzing data and providing recommendations.

CONCLUSIONS

This study provides a foundation for exploring educational robotics' potential in programming education. Understanding Sphero SPRK's impact can optimize programming pedagogy, preparing students for a technology-driven future. Addressing limitations and future research will contribute to advancing educational robotics, shaping skilled and innovative programmers. In this current study, we determined the utilisation of a robotic programming assistant that could be utilised to aid students explore programming in an interactive manner by assembling small elements to construct a whole model. This involved the use of a Sphero robot, a smartphone, and the Sphero Edu App, which is used to gather building blocks that lead to coding. In accordance to the observations, beginning programmers who implemented the blocks method in the Sphero Edu App made substantial advancements in their comprehension of the programming language. Future study is projected to concentrate on areas of the curriculum where conceptual difficulties necessitate a more comprehensive explanation.

Limitations:

Despite the promising findings, this study also has some limitations that should be acknowledged. This study's limitations include a relatively small sample size, potentially limiting generalizability. The controlled environment and focus on Sphero SPRK may not represent all educational robotics tools or settings. The study primarily targeted novice programming learners, leaving room for investigation at different skill levels.

Future Research:

Future research should employ larger and more diverse samples in longitudinal studies to assess long-term impacts. Comparative studies with various educational robotics platforms can elucidate their effectiveness. Integrating educational robotics into different programming languages and curricula can explore adaptability. Investigating educational robotics' impact on learners with diverse learning styles can ensure inclusivity. Interdisciplinary studies can demonstrate cross-domain problem-solving abilities, while integrating educational robotics with traditional methods can optimise learning experiences.

References

[1] Xinolagos, S (2011). Object-Oriented Programming - What do Students Think of Objects and Classes?, Proceedings of the 14th IASTED International Conference on Computers and Advanced Technology in Education.

[2] Miliszewska, I & Tan, G (2007). Befriending computer programming: a proposed approach to teaching introductory programming, Issues in Informing Science and Information Technology, 4, 277-289.

[3] Malik, IS., & Coldwell-Neilson, J (2017). Impact of a New Teaching and Learning Approach in an Introductory Programming Course, Journal of Educational Computing, 55(6), 789-819.

[4] Piteire, M., & Costa, C., (2013). Learning Computing Programming: Study of difficulties in learning programming. ISDOC, 75-79.

[5] Black, TR (2006). Helping Novice Programming Students Succeed, Journal of Computing Sciences in Colleges, 22(2), 109-114.

[6] Mubin, O., Stevens, CJ., Shahid, S., Al Mahmud, A., & Dong, JJ (2013). A review of the Applicability of Robotics in Education, Technology for Education and Learning, 1, 1-7.

[7] Buttler, M., & Morgan, M (2007). Learning challenges faced by novice programming students studying high level and low feedback concepts, ASCILITE, 99-107.

[8] Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W & Erickson, J (2011). Computational thinking for youth in practice, ACM Inroads, 2, 32-37.

[9] Mubin, O., Stevens, CJ., Shahid, S., Al Mahmud, A & Dong, JJ (2013). A review of the applicability of robots in education, Technology for Education and Learning 1(1), 1-7.

[10] Kelleher, C., & Pausch, R (2003). Lowering the Barriers to Programming: a survey of programming environments and languages for novice programmers, Journal Computer Surveys, 37(2), 83-137.

[11] Giangrande, E (2007). CSI Programming Language Options, Journal of Computing Sciences in Colleges, 3, 153-160.
[12] Winslow, LE (1996). Programming pedagogy – a psychological overview, ACM SIGCSE Bulletin, 28(3), 17-25.
[13] Robins, A., Rountree, J & Rountree, N (2003). Learning and teaching programming: A review and Discussion, Computer Science and Education, 13(2), 137-172.
[14] Programming with Sphero EDU. (2018, February 2). Retrieved from https://support.sphero.com/support/solutions/articles/9000100576-programming-with-sphero-edu-
[15] Booth, T., & Stumpf, S. (2013). End-user experiences of visual and textual programming environments for Arduino. In End-User Development: 4th International Symposium, IS-EUD 2013, Copenhagen, Denmark, June 10-13, 2013. Proceedings 4 (pp. 25-39). Springer Berlin Heidelberg.
[16] De Raadt, M (2008). Teaching programming strategies explicitly to novice programmers, Thesis Dissertation-USQ Faculty of Business, 1-124.
[17] Wing, JM (2006). Computational thinking, Communications of the ACM - Self Managed Systems, 49(3), 33-35.
[18] Atmatzidou, S & Demetriadis, S (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences, Robotics and Autonomous Systems, 75, 661-670.
[19] Wing, JM (2008). Computational thinking and thinking about computing, Philosophical Transactions of the Royal Society A Mathematical, Physical and Engineering Science , 366, 3717-3725.
[20] Benitti, FB (2012). Exploring the educational potential of robotics in schools: A systematic review, Journal of Computers & Education, 58(3), 978-988.
[21] Mitnik, R., Recabarren, M., Nussbaum, M & Soto, A (2009). Collaborative robotic instruction: A graph teaching experience, Journal of Computers and Educational Computing, 53, 330-342.
[22] Blanchard, S., Freiman, V & Lirrete-Pitre, N (2010). Strategies used by elementary schoolchildren solving robotics-based complex task: Innovative Potential of Technology, Social Behavioural Science, 2, 2851-2857.
[23] Chambers, JM., Carbonaro, M., Rex, M & Grove, S (2007). Scaffolding knowledge construction through robotic technology: A middle school case study, Journal of Integration Technology and Education, 6, 55-70
[24] Futschek, G., & Moschitz, J (2011). Learning algorithmic thinking with tangible objects eases transition to computer programming. In Informatics in Schools. Contributing to 21st Century Education: 5th International Conference on Informatics in Schools: Situation, Evolution and Perspectives, ISSEP 2011, Bratislava, Slovakia, October 26-29, 2011. Proceedings 5 (pp. 155-164). Springer Berlin Heidelberg.
[25] Ko, A. J., Myers, B. A., & Aung, H. H. (2004, September). Six learning barriers in end-user programming systems. In 2004 IEEE Symposium on Visual Languages-Human Centric Computing (pp. 199-206). IEEE.
[26] Repenning, A., Webb, D & Loannidou, A (2010). Scalable Game Design and the Development of a Check List for Getting Computational Thinking into Public Schools, Symposium of Computing Science and Education, 265-269.
[27] Schulte, C & Bennedsen, J (2006). What do teachers teach in introductory programming?, Proceedings the 2006 International workshop on Computing Education Research ICER, 17-28.
[28] Fanchamps, N. L., Slangen, L., Hennissen, P., & Specht, M. (2021). The influence of SRA programming on algorithmic thinking and self-efficacy using Lego robotics in two types of instruction. International Journal of Technology and Design Education, 31, 203-222.
[29] Mhashi, MM. & Alakeel, AM (2013). Difficulties Facing Students in Learning Computer Programming Skills at Tabuk University, Conference on Recent Advances in Modern Educational Technology, 1-10.