

TVET Innovation: Line Follower Robot Programming Approach Using Excel Platform

Mohd Fadrul Rizal B. Ab Rahim¹, W. M. Rizairie B. W. Md. Noor^{1*}, Farih B. Deraman¹

¹Department of Mechanical Engineering (Mechatronics Unit),
Politeknik Sultan Mizan Zainal Abidin, 23000 Dungun, Terengganu
Corresponding author E-mail: rizairie@psmza.edu.my

Abstract

The line follower robot serves as a fundamental element in robotics education as it introduces key concepts of automation, motor control, and sensor integration, which are essential foundations for modern intelligent systems. This project presents an innovative approach to simplifying the programming of line follower robots by utilizing Microsoft Excel as the primary platform for automatically generating Arduino code. The main issue identified is that many primary, secondary, and engineering students without programming backgrounds struggle with conventional coding languages such as C, while block-based programming platforms still require a deep understanding of specific functions, leading to confusion among new users. The objective of this project is to develop a more user-friendly method that allows students to simply input basic parameters such as direction (bearing), distance, and speed into an Excel spreadsheet without needing to understand complex code structures. These parameters can then be copied directly and pasted into the Arduino IDE, allowing the build and burn processes to proceed as usual. Findings show that this approach effectively accelerates the understanding of basic robotics, reduces syntax errors, and simplifies system modifications. Its impact is significant in empowering students with essential technical skills, strengthening STEM and TVET competencies, and supporting the development of capabilities aligned with the needs of the Fourth Industrial Revolution (IR 4.0). This approach not only simplifies the teaching of robotics to students without a technical background but also has great potential to expand access to robotics education at all levels.

Key Words: *Line Follower Robot, Programming Approach, Excel Automation, STEM and TVET Education*

1.0 Introduction

The Line Follower Robot (LFR) is a foundational tool in STEM and TVET education, offering hands-on experience with automation, sensor integration, and logic-based control. Programming the LFR can be approached in two major ways: direct coding using C language in Arduino IDE and block programming using visual platforms such as Scratch, mBlock, or TinkerCAD. Each method has distinct advantages and limitations, especially for students at the beginner level. Research has shown that early engagement with robotics enhances problem-solving skills, computational thinking, and motivation in STEM subjects (Benitti, 2012). Integrating robotics into primary curricula can foster collaborative learning and boost confidence (Mubin et al., 2013). Lack of Early Exposure to Programming. Many primary school students, especially those in rural or semi-private settings, have limited or no exposure to computer programming concepts, resulting in low STEM engagement from an early age (Wong & Lee, 2019).

Direct coding in C offers several pedagogical and practical advantages. First, it allows complete control over the hardware. Students can directly manipulate sensor inputs, motor speeds, PID tuning, and path logic, enabling them to engage deeply with how robots' function (Chan & Chiu, 2015). This level of control is essential for high-performance robotic behavior, and for developing complex applications beyond basic navigation. Furthermore, C is a standard language in embedded systems and industry applications. Students who master Arduino C coding are better equipped for real-world engineering, robotics, and automation careers, aligning their skills with the demands of the Fourth Industrial Revolution (Johnson, 2024). Additionally, direct coding typically results in more efficient execution, with faster processing and reduced resource consumption, which is important for time-sensitive robotic tasks.

However, direct C coding poses significant challenges for beginners. The syntax is rigid and complex, requiring understanding of loops, conditional logic, variables, and function structures. Minor syntax errors (e.g., missing semicolons or incorrect brackets) can prevent code compilation entirely, which can frustrate and demotivate students with no prior experience. For many young learners, the lack of visual support and the abstract nature of code make it difficult to grasp logical flow and robot behavior. Difficulty with Text-Based Syntax. Beginners often struggle with the syntax of text-based languages such as C (Ozkan & Barker, 2009), creating a barrier to entry for microcontroller-based robotics projects.

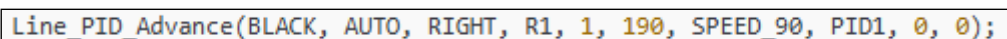
In contrast, block programming presents a more accessible entry point for novice learners. Visual environments allow students to drag and drop code blocks that represent movement, control, and logic operations. This reduces the cognitive load associated with syntax, allowing learners to focus on algorithmic thinking and logical structure (Resnick et al., 2009). The use of colors, icons, and guided interfaces makes the programming experience more intuitive, interactive, and engaging, especially for primary and lower secondary students (Sapounidis, 2023). Unfamiliarity with Block-Coding Paradigms. Although block-coding environments (e.g., Scratch, Blockly) can lower the entry threshold, many students remain unfamiliar with them and find the transition to “real” robot code abrupt (Papert, 1980).

Despite its ease of use, block programming has inherent limitations. The predefined blocks may restrict creative or advanced programming. Students cannot easily modify low-level hardware functions or introduce custom algorithms. As such, block programming is less suited for complex robotics applications or integration with industrial protocols (Weintrop et al., 2016). This gap may hinder the transition of students from beginner-friendly platforms to more advanced environments, unless a scaffolded transition strategy is in place.

In summary, both direct C coding and block programming have a valuable role in robotics education. Block programming is ideal for introducing basic concepts and building confidence, while direct coding should be introduced progressively as students develop their understanding and seek to implement more complex or customized robotic behaviors. A blended approach beginning with visual tools and transitioning to direct coding may be the most effective strategy in fostering technical competence and long-term interest in robotics.

One of the often-overlooked challenges in teaching programming, especially robotics beginners, is lack of imagination or mental visualization. For many students, especially at the primary and early secondary levels, understanding the relationship between code (syntax) and real-world robot behavior is difficult due to limited ability to visualize how specific instructions translate into physical actions. This cognitive gap has been shown to significantly affect student interest, motivation, and confidence in learning programming (Bers, 2018).

Programming a Line Follower Robot (LFR) involves abstract concepts such as motion control, conditional logic, and sensor-based decision making. In environments like Arduino IDE where syntax is written in C, a student must mentally imagine how a line of code is shown in Figure 1 that translates to the robot's physical movement that means turning right at a specific junction, with a particular speed and timing.



```
Line_PID_Advance(BLACK, AUTO, RIGHT, R1, 1, 190, SPEED_90, PID1, 0, 0);
```

Figure 1: Syntax C to instruct LFR

For students without strong spatial or kinetic imagination, this process is overwhelming and unintuitive. According to Papert (1980), effective learning in computing requires the learner to develop a “syntonic relationship” with the machine as sense of empathy or mental identification

with how the robot “feels” or behaves. When this is missing, the learner cannot connect abstract symbols to expected outcomes.

This disconnect between abstract syntax and tangible behavior is especially pronounced in younger learners who may not yet have fully developed abstract reasoning skills, which typically begin to mature around age 12 (Piaget, 1972). Without scaffolding or visual aids, many students find it difficult to understand why a robot behaves in a certain way after uploading the code. This not only creates frustration but often leads to disengagement from programming altogether (Weintrop et al., 2016).

Furthermore, the invisibility of code execution adds to this challenge. Unlike video games or animations where changes are immediately visible, robotics code runs inside a microcontroller with delayed or indirect feedback. Students must wait to observe the outcome, and when the robot does not behave as intended, they often struggle to trace the error back to their imagination or code logic. This results in trial-and-error learning without real understanding, which reduces long-term retention and undermines self-efficacy in STEM learning (Zhong, 2020).

To address this, educational researchers recommend tools and strategies that externalize the mental model. Block-based programming, simulations, and step-by-step animations can help bridge the gap by allowing students to see and manipulate abstract ideas more concretely (Resnick et al., 2009). Similarly, the innovation of using Microsoft Excel as a visual parameter-entry tool where each cell represents a part of the robot’s journey has proven to significantly improve understanding and imagination. Instead of writing code, students describe movement using everyday terms such as “Turn Left,” “Move Forward,” and “Speed 90,” and then see the code generated from those instructions (Chan & Chiu, 2015), (Németh, 2021).

In conclusion, weak imaginative ability is a cognitive barrier that contributes to low interest and perceived difficulty in coding, especially in robot-related projects like LFR. Overcoming this requires designing learning experiences that help students visualize robot behavior before coding. By doing so, we not only make programming more accessible but also cultivate deeper cognitive engagement with computational thinking, a core component of 21st-century education.

The challenge of teaching robotics and programming to beginners lies not only in the complexity of programming languages like C, but also in cognitive barriers, such as students’ limited ability to visualize how code maps onto real-world robot actions (Bers, 2018; Papert, 1980). Text-based programming languages present steep learning curves for novices. Block-based platforms alleviate syntax errors and enable learners to focus on logical structures (Kong, 2022). However, block environments still require digital literacy that some young learners lack (Weintrop et al., 2016).

While block programming platforms such as Scratch and mBlock offer easier syntax through visual drag-and-drop commands, they are constrained by limited functions and lack the fine-grained control available in direct C coding (Looi, 2024). To address these challenges, this project introduces a novel method that leverages Microsoft Excel as a bridge between user-friendly interfaces and the full power of Arduino-based C programming. Spreadsheets such as Microsoft Excel are ubiquitous in school settings and familiar to both teachers and students. Prior studies demonstrate that using spreadsheets to teach algorithmic thinking can be effective: learners map computations to cells and formulas, reinforcing the concept of step-by-step instructions (Bruun & Freitas, 2017).

This project was developed in response to the challenges faced by beginner programmers, particularly primary school students, in understanding and applying conventional robot programming using text-based languages. The approach centers on creating a simplified yet powerful platform for robot instruction generation using Microsoft Excel. The project is guided by the following three core objectives:

1. Designing an Excel-Based Interface to Abstract Programming Syntax

The primary objective of the project is to design a user-friendly interface using Microsoft Excel that allows students to program the Line Follower Robot (LFR) without having to understand or write complex text-based syntax. Rather than requiring knowledge of C-language programming, the system enables students to input key movement parameters such as direction (LEFT, RIGHT, FWD), speed, sensor type, and turn behavior into a structured Excel table. Formulas embedded in the spreadsheet then automatically generate syntactically correct Arduino C-code. This design ensures that syntax errors are avoided and that the students can focus on the logic of robot movement instead of programming rules. The interface bridges the gap between abstract code and practical robotics by providing a visual, parameter-driven environment that lowers the entry barrier to coding.

2. Evaluating the Interface's Effectiveness in Enhancing Conceptual Understanding

A secondary but equally important goal of the project is to evaluate how well the Excel-based interface improves students' conceptual understanding of robotic control logic. Unlike conventional programming, which often hides the relationship between code and behavior, this method makes the connection between input and output explicit. Each cell in the Excel table directly maps to a line of robot behavior, enabling students to grasp the cause-effect relationships of each command. By removing the distractions of syntax and focusing on logical sequencing, the system encourages students to think critically about how the robot navigates a circuit, detects junctions, and responds to sensor input. The project measures conceptual gains through pre- and post-tests that assess students' knowledge of robot movement and decision-making.

3. Measuring Changes in Motivation and Confidence in Programming

The third objective of the project is to assess the impact of this innovative method on students' motivation and confidence levels in programming. For many young learners, programming can seem intimidating due to its abstract nature and steep learning curve. By offering an accessible tool that feels familiar and achievable since most students have prior exposure to Excel. The project aims to foster positive emotional engagement with coding. To measure this, students completed pre-and-post intervention surveys using Likert scales, which gauged their enjoyment of robotics, perceived ease of learning, confidence in using Excel, and interest in learning more about coding. The project hypothesizes that by experiencing success early through the generation of working robot code. Students will be more motivated to explore advanced programming and continue learning in STEM fields.

2.0 Research Methodology

Intervention Design

Students filled in a standardized Excel (Figure 2) sheet listing movement direction, sensor configuration, speed parameters, PID profile, and timing. A set of Excel formulas concatenated these entries into Arduino `Line\_PID\_Advance ()` statements, which were copied into the Arduino IDE for uploading to the robot hardware.

LINE	COMMAND	LINE COLOR	MODE	DERECTION	SENSOR	TURN	MAX SPEED	SPEED TURN	PID
1	LINE	(BLACK,	AUTO,	LEFT,	L1,	1,	190,	SPEED90,	PID1,
2	LINE	(BLACK,	AUTO,	LEFT,	L1,	1,	190,	SPEED90,	PID2,
3	LINE	(BLACK,	AUTO,	RIGHT,	R1,	1,	190,	SPEED90,	PID1,
4	LINE	(BLACK,	MOTION,	FWD,	L1,	1,	190,	SPEED90,	PID1,
4	LINE	(BLACK,	AUTO,	LEFT,	L1,	2,	190,	SPEED30,	PID1,
5	LINE	(BLACK,	UTURN,	LEFT,	L1,	2,	190,	SPEED90,	PID1,
6	LINE	(BLACK,	MOTION,	LEFT,	L1,	1,	190,	SPEED60,	PID1,
7	LINE	(BLACK,	AUTO,	LEFT,	L1,	1,	190,	SPEED90,	PID1,
8	LINE	(BLACK,	AUTO,	LEFT,	L1,	1,	190,	SPEED90,	PID1,
9	LINE	(BLACK,	AUTO,	RIGHT,	R1,	1,	190,	SPEED90,	PID1,
10	LINE	(BLACK,	AUTO,	RIGHT,	R1,	1,	190,	SPEED90,	PID1,
11	LINE	(BLACK,	AUTO,	LEFT,	L1,	1,	190,	SPEED90,	PID1,
12	LINE	(BLACK,	AUTO,	FWD,	R1,	1,	190,	SPEED90,	PID3,

Figure 2: Excel sheet uses by student to input the parameter

The flowchart in Figure 3 illustrates a step-by-step procedure for using Microsoft Excel as a code generation interface to program a Line Follower Robot (LFR) without requiring direct interaction with C programming syntax. This method was specifically designed to support beginners in robotics and programming by simplifying the coding process into a structured and visual experience.

The process begins with launching a pre-designed Excel spreadsheet that acts as an interactive platform for inputting robot parameters. The first few steps involve the user selecting various control parameters, including the direction of the robot (e.g., forward, left, right), the speed at which it should move, the PID control profile to be used (for line stability), and the type of sensor or junction to detect. These selections are made via dropdown lists or guided cells within the Excel sheet. Each parameter corresponds to a part of the robot's movement logic, allowing users to visualize the path and behavior of the robot in a step-by-step format.

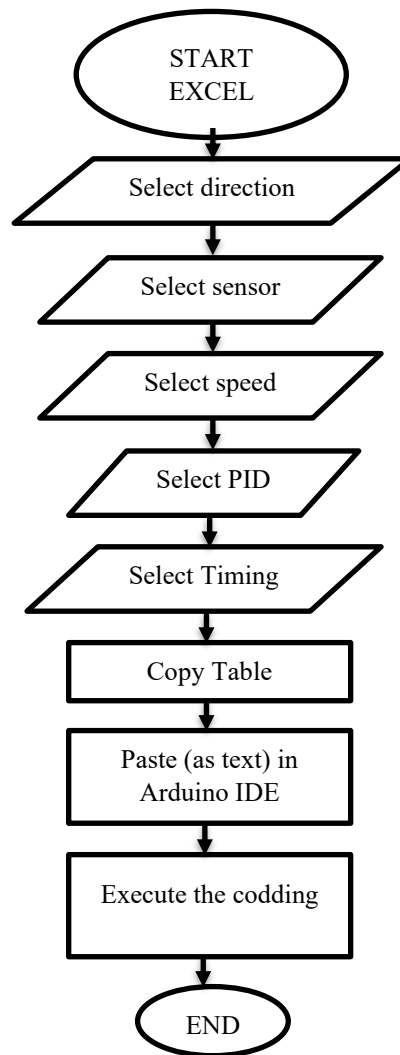


Figure 3: A step-by-step of using Microsoft Excel for coding

Once all relevant values are entered, the spreadsheet automatically generates Arduino-compatible code by combining the selected parameters into a structured function call (typically `Line_PID_Advance(...)`). This eliminates the need for users to write code manually or worry about syntax accuracy. The user then proceeds to copy the generated code from Excel and paste it as plain text into the Arduino IDE. After passing, the user uploads (executes) the code to the microcontroller on the robot, completing the process.

This workflow effectively bridges the gap between visual planning and executable programming. It avoids the cognitive overload associated with manual coding, reduces syntax errors, and enables a smooth learning curve for students unfamiliar with traditional programming environments. Moreover, because the output is still standard C code, it retains the flexibility and power of direct programming, enabling more advanced control when needed. In summary, this method promotes intuitive programming while maintaining technical rigor, making it ideal for use in educational settings, particularly at the primary and lower secondary levels.

Testing

To determine the effectiveness of the proposed Excel-based programming approach for Line Follower Robots (LFR), a structured experimental study was conducted involving 96 primary school students from two different institutions in Dungun, Terengganu. These participants included 40 students from Sekolah Rendah Agama Bersepadu Zainal Abidin, a semi-private school located in Gong Pasir, and 56 students from Sekolah Kebangsaan Sura, a public primary school. The selected students were in Year 2 and Year 3, representing the lower primary level, where students begin to develop more abstract reasoning and logical thinking. To comprehensively evaluate both cognitive and affective outcomes, the researchers employed two main assessment tools: Knowledge Test: A set of five multiple-choice questions was designed to assess students' understanding of the key concepts related to the LFR system. These included the robot's purpose, the role of Arduino IDE, the function of Excel as a parameter input tool, command interpretation (e.g., "RIGHT" or "FWD"), and the overall advantages of this innovative method. These questions were aligned with the lesson content and tested basic comprehension and logic mapping between parameter input and robot behavior.

Likert Scale Survey: To assess motivation, confidence, and perceived learning ease, a five-item Likert scale questionnaire was developed. The statements measured students' enjoyment of robotics, their confidence in using Excel, how helpful the table-based interface was in understanding instructions, their interest in learning more about coding, and how much the method simplified their understanding compared to conventional approaches. Each item was rated on a 5-point scale, ranging from "Strongly Disagree" to "Strongly Agree." Testing Procedure: The evaluation followed a pre- and post-test design to measure the impact of the innovation over a short-term learning intervention. Before any exposure to the Excel interface, students were administered both the knowledge test and the Likert scale survey. This baseline data provided insight into the students' prior understanding of robot programming and their initial attitudes toward coding and robotics.

Students were then introduced to the Excel-based interface, where they learned how to input movement parameters for the LFR based on a visual layout of the circuit. After entering data into the Excel table, students copied the automatically generated Arduino C code and uploaded it into the Arduino IDE to run the robot. This hands-on experience provided immediate feedback and reinforced the link between abstract input and physical robot behavior. Upon completion of the programming session and first successful robot run, students were re-administered the same knowledge test and Likert scale survey. This allowed for a direct comparison of learning outcomes and attitudinal shifts.

Analysis Method

To analyze the results, descriptive statistics were computed for both instruments. For the knowledge test, the total number of correct responses for each question was tallied across all participants, allowing researchers to observe the percentage of improvement for each item. For the Likert scale, the average score (mean) for each of the five items was calculated both before and after the intervention. To determine whether the observed differences were statistically significant, paired sample t-tests were performed on both sets of data with a significance level of $\alpha = 0.05$. This method tested whether there was a meaningful improvement in knowledge and motivation following the Excel-based activity.

3.0 Finding

Table 1 below illustrates the number of correct answers (out of 96 students) for each of the five knowledge-based questions, showing significant improvement in all areas after the intervention.

Table 1:Pre-and-Post Test Correct Responses for Knowledge Questions

QUESTION	PRE (N=96)	POST (N=96)	% IMPROVEMENT
Q1	45	85	+44.8%
Q2	38	78	+41.7%
Q3	30	80	+52.1%
Q4	41	88	+47.9%
Q5	35	76	+44.8%

All five knowledge items saw statistically significant gains ($p < .001$), with an overall mean increase from 37.8 to 81.4 correct answers per item ($t(95) = 18.27$, $p < .001$). This suggests a strong conceptual uptake of LFR programming via the Excel interface.

The bar chart at Figure 4 below presents average student scores on Likert-scale statements related to enjoyment, confidence, comprehension, and motivation in coding before and after the Excel-based robotics activity.

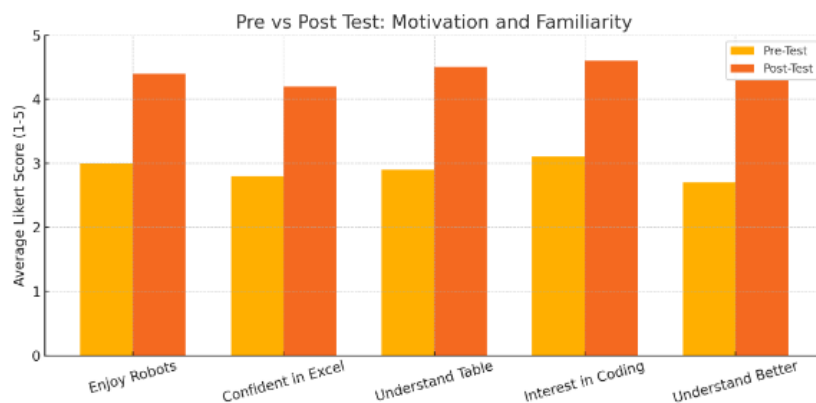


Figure 4: Pre Vs Post Test: Motivation and Familiarity

Paired t-tests confirmed significant improvements across all five items (all $p < .001$). The largest gains occurred in perceived ease of understanding and overall interest, underscoring the motivational impact of the platform.

4.0 Conclusion

This project introduces an innovative Excel-based interface that simplifies the programming of Line Follower Robots (LFR) by enabling students to input essential parameters such as direction, turning angle, speed, junctions, and PID settings into a structured spreadsheet that visually represents the robot's movement along a predefined path. These parameters are then

automatically converted into syntactically correct Arduino C code, which can be directly executed in the Arduino IDE.

This approach offers several key advantages. First, it bypasses the complexity of C syntax, allowing beginners to focus on the logic of robot movement without encountering frustrating code errors. Second, it overcomes the limitations of block-based programming platforms by preserving the full flexibility and power of text-based code, including advanced functionalities like PID tuning and custom sensor behavior. Third, it addresses a major cognitive barrier by providing a visual scaffold that connects programming logic with physical robot actions, enhancing understanding and mental modeling. Furthermore, the method encourages constructive and exploratory learning, fostering curiosity, confidence, and computational thinking. Finally, because the output remains editable, it supports scalability and customization, making it suitable not only for novices but also for more advanced learners.

In summary, this hybrid solution bridges the gap between ease of use and programming depth. By combining the familiarity of Excel with the capabilities of C programming, it empowers students to engage meaningfully with robotics, preparing them for more complex challenges in STEM and TVET education. This work builds on prior research and extends the potential of spreadsheet-driven robotics as an accessible and pedagogically powerful tool.

References

- Benitti, F. B. V. (2012). Exploring the educational potential of robotics in schools: A systematic review. *Computers & Education*, 58(3), 978–988. <https://doi.org/10.1016/j.compedu.2011.10.006>
- Bers, M. U. (2018). *Coding as a playground: Programming and computational thinking in the early childhood classroom*. Routledge. <https://doi.org/10.4324/9781315398945>
- Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, Vancouver, Canada.
- Bruun, A., & Freitas, S. (2017). Spreadsheets as a tool for developing algorithmic thinking. *Journal of Educational Computing Research*, 55(6), 801–817. <https://doi.org/10.1177/0735633116684797>
- Chan, H. C., & Chiu, P.-L. (2015). Spreadsheet-driven robotics: Engaging novices with parameterized robot control. *International Journal of STEM Education*, 2(1), 9. <https://doi.org/10.1186/s40594-015-0017-8>
- Johnson, R. (2024). Encouraging high school students to pursue STEM degrees and careers: The role of early exposure. *Athens Journal of Education*, 11(2), 123–138. <https://www.athensjournals.gr/education/2024-6096-AJE-EDU-Johnson-02.pdf>
- Kong, S. C., & Lai, M. (2022). The effectiveness of Scratch-based programming on students' computational thinking skills: A meta-analysis. *Journal of Educational Computing Research*, 60(1), 3–29. <https://doi.org/10.1177/07356331211051900>
- Looi, C.-K., et al. (2024). Computational thinking in primary mathematics classroom activities. *Frontiers in Education*, 9, 1414081. <https://doi.org/10.3389/feduc.2024.1414081>
- Mubin, O., Stevens, C. J., Shahid, S., Al Mahmud, A., & Dong, J.-J. (2013). A review of the applicability of robots in education. *Technology for Education and Learning*, 1(1), 1–7. <https://doi.org/10.2316/Journal.209.2013.1.209-0015>
- Németh, Z., & Molnár, G. (2021). Developing computational thinking skills with algorithm-driven spreadsheeting. *Informatics in Education*, 20(2), 229–246. <https://doi.org/10.15388/infedu.2021.12>

- Ozkan, S., & Barker, P. (2009). Teaching “C” programming to novices: Promoting active learning using peer instruction and improvisational exercises. *ACM SIGCSE Bulletin*, 41(1), 195–199. <https://doi.org/10.1145/1508865.1508928>
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books.
- Piaget, J. (1972). *The psychology of the child*. Basic Books.
- Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... & Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM*, 52(11), 60–67. <https://doi.org/10.1145/1592761.1592779>
- Sapounidis, T., Demetriadis, S., & Stamelos, I. (2023). The effects of educational robotics in STEM education: A multilevel meta-analysis. *International Journal of STEM Education*, 10(1), 1–20. <https://doi.org/10.1186/s40594-024-00469-4>
- Silva, A. L., & Lopes, A. P. (2021). Active learning in the context of the teaching/learning of computer programming: A systematic review. *Education and Information Technologies*, 26(5), 5911–5937. <https://doi.org/10.1007/s10639-021-10571-2>
- Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, 25(1), 127–147. <https://doi.org/10.1007/s10956-015-9581-5>
- Wong, P. K., & Lee, L. S. (2019). Impact of early coding exposure on STEM career ambition in primary school children. *Asian Journal of Education and Training*, 5(2), 145–152. <https://doi.org/10.20448/journal.522.2019.52.145.152>
- Zhong, B., & Xia, L. (2020). The effects of educational robotics in STEM education: A multilevel meta-analysis. *International Journal of STEM Education*, 7(1), 1–20. <https://doi.org/10.1186/s40594-020-00238-0>