International Journal of Research in Special Education

E-ISSN: 2710-3870
P-ISSN: 2710-3862
Impact Factor (RJIF): 6.69
IJRSE 2026; 6(4): 60-64
© 2026 IJRSE
www.rehabilitationjournals.com
Received: 26-01-2026
Accepted: 28-02-2026

Flavio Cezar Amate
Associate Professor, Department of Informatics, Federal Institute of Education, Science and Technology of São Paulo (IFSP), Bragança Paulista, SP, Brazil

Caique Kevin Piaçaroli Maccei Mendes
Student, Department of Informatics, Federal Institute of Education, Science and Technology of São Paulo (IFSP), Bragança Paulista, SP, Brazil

Corresponding Author:
Flavio Cezar Amate
Associate Professor, Department of Informatics, Federal Institute of Education, Science and Technology of São Paulo (IFSP), Bragança Paulista, SP, Brazil

Development of a tactile educational toolkit for teaching programming logic to students with visual impairments

Caique Kevin Piaçaroli Maccei Mendes and Flavio Cezar Amate

DOI: https://www.doi.org/10.22271/27103862.2026.v6.i4a.168

Abstract
Programming education remains highly dependent on visual representations, creating significant barriers for students with visual impairments. Traditional programming environments and teaching materials frequently rely on graphical interfaces and visual flow representations, which are difficult to access through assistive technologies such as screen readers. This study presents the design and preliminary validation of a tactile educational toolkit aimed at supporting the teaching of programming logic through physical representation of algorithm structures. The prototype was developed based on classical flowchart conventions and fabricated using modular tactile components that allow the assembly of sequential and conditional algorithm structures. A laboratory-based evaluation was conducted with four programming specialists who tested the prototype by assembling a flowchart representing a conditional algorithm. Usability was assessed using the System Usability Scale (SUS) and an additional structured questionnaire evaluating symbolic coherence, assembly stability, branching representation, and structural readability. The results indicated a mean SUS score of 88.75, corresponding to an excellent level of usability. Specialists also reported high agreement regarding the clarity and pedagogical consistency of the tactile representation. The findings suggest that the proposed toolkit provides a coherent and intuitive physical model for representing algorithm structures, demonstrating strong potential as an inclusive educational resource for introductory programming instruction. Future work will involve empirical evaluation with visually impaired learners in authentic educational contexts.

Keywords: Inclusive education, accessible computing education, visual impairment, tactile educational toolkit, algorithm representation, usability evaluation

1. Introduction
Computer Science education has increasingly embraced inclusive policies aiming to provide equitable access to students with disabilities. However, programming and algorithm education remain particularly challenging for people with visual impairments (PwVI), especially blind and low-vision learners. Learning to program is cognitively demanding even for sighted students, and these challenges are amplified when instructional tools rely predominantly on visual representations [4].

Empirical investigations into the educational experiences of blind programmers reveal persistent barriers in accessing development environments, understanding code structure, and participating in collaborative learning contexts [3]. These barriers are often rooted in the visual design of programming interfaces and instructional materials.

Recent systematic investigations have shown that although multiple initiatives attempt to promote accessibility in programming education, empirical evidence remains limited and fragmented [8]. A systematic mapping study covering the period from 2014 to 2024 identified only a small number of validated interventions combining pedagogical tools with structured HCI evaluation methods [8]. Similarly, systematic reviews focusing on programming accessibility highlight that most solutions remain experimental and lack longitudinal validation [7].

Traditional programming environments depend heavily on graphical user interfaces and text-based code editors. Screen readers, currently the most common assistive technology used by blind programmers, translate visual interfaces into linear auditory streams. However, this model presents intrinsic limitations, particularly in handling indentation, punctuation, and hierarchical structure [5]. Such constraints increase cognitive load and hinder structural comprehension [2].

Moreover, block-based programming systems such as Scratch and Blockly, while effective for sighted novices, remain largely inaccessible to blind learners due to their visual and spatial interaction paradigms [6].

Given these structural and pedagogical challenges, there is a clear need for alternative instructional strategies that go beyond simply adapting visual tools. Research on tactile teaching strategies demonstrates that multisensory approaches can positively impact learning outcomes among visually impaired students [1]. Therefore, structured tactile representations may offer more inclusive pathways for algorithm comprehension and logical reasoning.

This study presents the results of a scientific initiation project aimed at developing and evaluating tactile educational resources to support the teaching of programming logic and algorithmic thinking to visually impaired students. The research contributes empirical evidence to an area that still lacks robust experimental validation [8].

2. Theoretical Background

2.1 Barriers in Programming Education for PwVI

Multiple studies have documented accessibility barriers in programming education for blind and low-vision learners. Santos *et al* [8] identified a limited number of empirically evaluated tools and emphasized that most initiatives target early education contexts, leaving higher education relatively underexplored.

Qualitative studies investigating blind programmers' educational trajectories report recurring difficulties related to inaccessible interfaces, limited instructor preparedness, and inadequate structural navigation tools [3]. Additionally, literature reviews on accessible programming education highlight that many development environments remain incompatible with screen readers or rely on visual metaphors that are difficult to translate into auditory representations [4, 6].

Mountapmbeme [6] further identifies specific challenges in block-based programming environments, including navigation inefficiencies, code comprehension barriers, and difficulties in debugging tasks.

2.2 Structural Limitations of Screen Reader-Based Programming

Lock, Brown, and Kölling [5] argue that the paradigm of "reading the screen" is inherently limited for programming education. Screen readers linearize two-dimensional code structures into sequential audio streams, which may obscure hierarchical relationships and semantic prioritization.

Research on blind students' access to visual information suggests that structural and spatial relationships are essential for mental model construction in Computer Science education [2]. When code structure must be reconstructed cognitively rather than perceived spatially, cognitive load increases significantly.

As discussed by Lock, Brown, and Kölling [5], the limitations documented in the literature include inconsistent narration of punctuation, ambiguity in indentation-based languages, confusion between semantically distinct elements, and linear narration incompatible with different reading strategies.

These findings reinforce that accessibility cannot rely solely on screen reader adaptation but requires structural rethinking of representation models.

2.3 Tangible and Multimodal Approaches

Systematic mappings identify tactile flowcharts, tangible programming kits, and haptic models as promising alternatives for accessible algorithm instruction [8]. Such approaches externalize program structure into manipulable physical representations, reducing dependence on visual perception.

Empirical evidence indicates that tactile teaching strategies can positively influence achievement and engagement among visually impaired learners [1]. Furthermore, multimodal approaches combining tactile and auditory feedback may enhance structural comprehension and collaborative inclusion [2, 4].

However, despite promising preliminary results, robust experimental validation remains scarce, particularly in higher education settings [8].

2.4 Research Gap

Despite growing awareness, empirical validation remains scarce, highlighting the need for controlled studies that assess not only usability but also learning effectiveness [8].

Despite growing interest in inclusive programming education, significant gaps persist in the development of experimentally validated tactile tools for algorithm instruction, the structured assessment of learning outcomes, the application of accessible methodologies in higher education contexts, and the systematic integration of logical reasoning with accessible design principles.

This study addresses these gaps by designing and structurally validating a tactile educational prototype focused on programming logic representation, as a preliminary step toward future empirical evaluation with visually impaired learners.

3. Materials and Methods

3.1 Research Design

This study adopted a design and development research approach aimed at conceptualizing and prototyping a tactile educational system for algorithm representation. Rather than conducting an intervention with visually impaired learners, the present phase of the research focused on the structural design, fabrication, and laboratory-based validation of the prototype.

The objective of this stage was to evaluate the technical feasibility, structural coherence, and pedagogical consistency of the tactile model before future empirical application with the target population.

3.2 Prototype Design

3.2.1 Conceptual Foundation

The prototype was based on classical flowchart representations widely used in introductory programming education. The design sought to translate algorithmic structures into physically manipulable components while preserving logical hierarchy and execution flow.

The conceptual model was guided by principles of structural clarity of control flow, modular representation of sequential and conditional logic, physical differentiation between algorithmic functions, and hierarchical organization of execution steps.

3.2.2 Fabrication Process

The tactile components were manufactured using laser-cut MDF panels to ensure dimensional accuracy and tactile

consistency. Each geometric form corresponds to a conventional flowchart symbol, including terminator, process, input/output, decision, connectors, and directional bars.

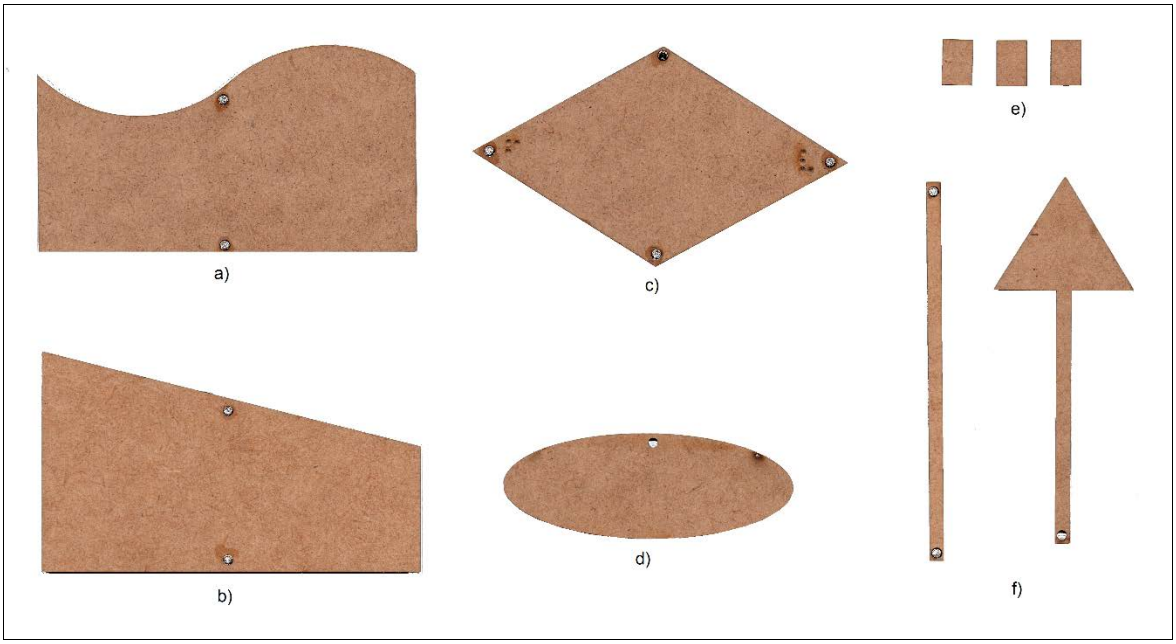


Fig 1: Tactile components of the modular algorithm representation system

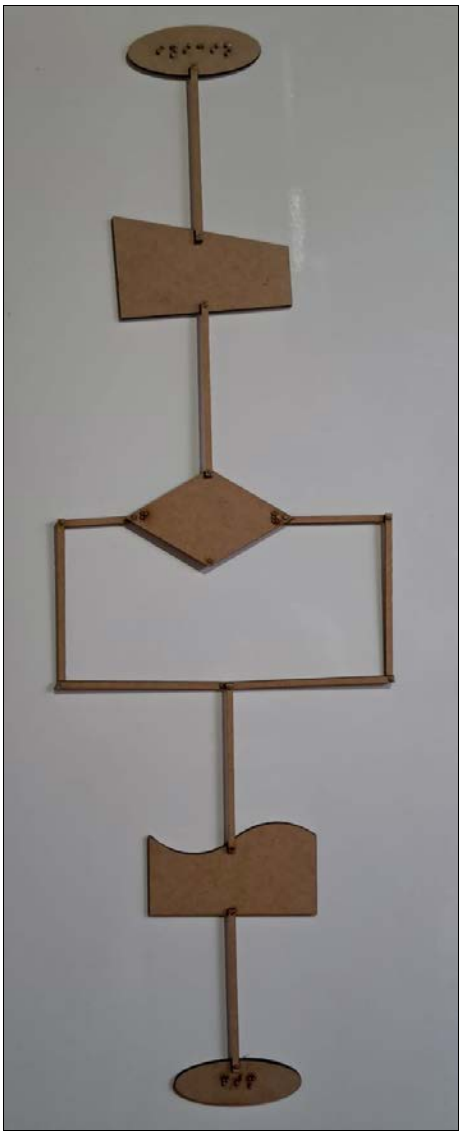


Fig 2: Example of assembled tactile flowchart representing a conditional algorithm structure

Mechanical connectors allow vertical and lateral assembly, enabling representation of sequential execution and conditional branching structures.
Figures 1 and 2 illustrate the individual components and a complete assembled prototype, respectively.

3.3 Laboratory-Based Validation
The validation phase was conducted in a controlled laboratory environment and followed an exploratory expert-based evaluation approach. The objective was not to measure learning outcomes but to assess the structural and pedagogical coherence of the prototype prior to future application with visually impaired learners.
The assembled models were examined by instructors and programming specialists to verify alignment with established algorithm representation principles. The evaluation considered structural fidelity to conventional flowchart standards, mechanical stability and modularity of assembly, clarity of hierarchical organization, and logical consistency of branching mechanisms.
Observational analysis focused on the coherence between symbolic function and geometric representation, the ease of physical assembly and reconfiguration, the adequacy of branching structures, and the overall structural readability of complete algorithm sequences. The validation process was qualitative and exploratory, aiming to identify potential structural inconsistencies before empirical testing with the target population.

3.4 Scope and Limitations
This study does not claim empirical validation of learning effectiveness among visually impaired students. Its contribution is limited to the design, development, and structured laboratory validation of a tactile educational prototype. Future research will involve empirical evaluation with visually impaired learners to assess usability, cognitive load, and instructional impact.

4. Results & Discussion
A usability evaluation was conducted with four programming specialists (n = 4) who tested the tactile educational prototype by assembling a flowchart representing a conditional algorithm. After completing the task, participants responded to two evaluation instruments: the System Usability Scale (SUS) and a structured questionnaire assessing the pedagogical and structural characteristics of the prototype.

4.1 System Usability Scale Results
The SUS results indicated a mean usability score of 88.75, which falls within the range considered excellent usability according to standard SUS interpretation guidelines. Individual SUS scores ranged from 85 to 97.5, demonstrating consistent positive evaluations among the specialists.
These results suggest that the prototype presents a high level of perceived usability, indicating that the physical representation of algorithm structures is intuitive and easy to manipulate.

4.2 Structural and Pedagogical Evaluation
The second questionnaire evaluated specific characteristics of the prototype, including symbolic coherence, modular assembly, branching representation, and structural readability.
The results revealed very high agreement among specialists. The dimension related to symbolic coherence achieved the maximum mean score (5.0), indicating that the geometric shapes used in the prototype accurately correspond to traditional flowchart symbols used in programming education.
The assembly and modularity dimension obtained a mean score of 4.75, suggesting that the physical structure of the prototype allows stable assembly and easy reconfiguration when representing different algorithms.
Similarly, the branching structure evaluation achieved a mean score of 4.75, confirming that conditional structures such as IF–ELSE statements can be clearly represented using the tactile components.
The structural readability dimension achieved a mean score of 4.83, indicating that the assembled flowcharts allow clear identification of execution flow and hierarchical organization of algorithm steps.
Finally, the global evaluation dimension reached a mean score of 4.88, showing strong agreement among specialists regarding the pedagogical potential of the prototype. Participants indicated that the toolkit could be used as an introductory instructional resource and that it presents promising potential for future application with visually impaired students.
Overall, the results demonstrate that the proposed tactile toolkit provides a coherent and usable physical representation of algorithm structures, supporting its feasibility as an accessible educational resource.

5. Conclusions
This study presented the design and preliminary validation of a tactile educational toolkit aimed at supporting the teaching of programming logic through physical algorithm representation.
The usability evaluation conducted with programming specialists demonstrated high levels of perceived usability, with a mean System Usability Scale (SUS) score of 88.75, indicating excellent usability. Additionally, the structured evaluation questionnaire revealed strong agreement regarding symbolic coherence, assembly stability, branching representation, and structural readability of the prototype.
The results suggest that the tactile toolkit is capable of accurately representing traditional flowchart structures while maintaining ease of manipulation and modularity. These characteristics indicate that the system has significant potential as an inclusive educational tool for introducing algorithmic thinking.
Although this study did not involve visually impaired learners, the expert-based evaluation indicates promising feasibility for future pedagogical applications. Future work will include empirical testing with visually impaired learners in authentic educational settings to evaluate learning outcomes and instructional impact.

6. Acknowledgments
The authors would like to thank the Federal Institute of Education, Science, and Technology and the Systems Analysis and Development Program for their support of this research.

References
1. Alotaibi H, Al-Khalifa HS, AlSaeed D. Teaching programming to students with vision impairment: Impact of tactile teaching strategies on students' achievements and perceptions. Sustainability. 2020;12(13):1–19. https://doi.org/10.3390/su12135287

2. Baker CM. Understanding and improving blind students' access to visual information in computer science education [doctoral dissertation]. University of Washington; 2017.
3. Baker CM, Bennett CL, Ladner RE. Educational experiences of blind programmers. In: Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19). ACM; 2019.
4. Hadwen-Bennett A, Sentance S, Morrison C. Making programming accessible to learners with visual impairments: A literature review. International Journal of Computer Science Education in Schools. 2018;2(2):1–20.
5. Lock JG, Brown NCC, Kölling M. Programming education for blind and low vision users: Beyond reading the screen. In: Proceedings of the 25th Koli Calling International Conference on Computing Education Research (Koli Calling '25). ACM; 2025. https://doi.org/10.1145/3769994.3770018
6. Mountapmbeme A. Understanding and addressing accessibility barriers faced by people with visual impairments on block-based programming environments [doctoral dissertation]. University of North Texas; 2022.
7. Mountapmbeme A. A systematic review on developing computer programming solutions for individuals with visual impairments. 2023.
8. Santos DS, Shibata NN, Pinto VHSC. Teaching algorithms and programming for people with blindness and visual impairments: A systematic mapping study. In: Proceedings of the 30th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2025). ACM; 2025. https://doi.org/10.1145/3724363.3729110
9. Sujata VG, Saini BK. Impact of COVID-19 pandemic situation on persons with visual impairment. International Journal of Research in Special Education. 2021;1(1):23–27.
10. Jung I. A study on basic software education applying a step-by-step blinded programming practice. 2019.
11. Konecki M, Lovrenčić A, Kudelić R. Making programming accessible to the blinds. 2016.
12. Papazafiropulos N, *et al*. Tactile 3D-printed models for teaching algorithms to visually impaired students. 2016.