Programming teaching with robotic support for people who are visually impaired: a systematic review

Juliana Damasio Oliveira[1], **Darlan A. Jurak**[1], **Robson F. Bittencourt**[1], **Márcia de B. Campos**[2], **Alexandre M. Amory**[1]

School of Technology– Pontifical Catholic University of Rio Grande do Sul (PUCRS)
90.619-900 – Porto Alegre – RS – Brazil

[2]Inedi College – CESUCA –94.935-630 – Cachoeirinha – RS – Brazil

juliana.damasio@acad.pucrs.br, darlan.jurak@edu.pucrs.br

marciabcampos@hotmail.com, alexandre.amory@pucrs.br

Abstract. *Robotics has been used to draw students' attention to computing and engineering. Unfortunately, most of the hardware and software features used to teach programming and robotics are not suitable for students who are visually impaired. Most programming environments are highly visual interfaces which makes them inaccessible for the visually impaired. This paper aims to present the studies regarding the use of robotics for the teaching of programming for people who are visually impaired that were published between 2016 and 2019, through a systematic literature review. As a result, we identified the teaching methodologies, the robotics and programming kits used, the good practices and difficulties faced in programming teaching with robotic support.*

1. Introduction

The use of robots are used as a facilitator and incentive to learn programming [Benitti et al. 2009, Meira et al. 2016]. In this sense, several robotic environments are being proposed to facilitate programming teaching, such as Lego Mindstorms [Klassner and Anderson 2003] and NaoBlocks [Sutherland and MacDonald 2018]. However, these initiatives do not include all students, especially those who are visually impaired (VI) (e.g., blind or low vision) [Barros et al. 2014]. This is because many programming environments are based on graphical interfaces, which makes them inaccessible to this group of users [Barros et al. 2014, Kane and Bigham 2014, Ludi and Reichlmayr 2011], its use becomes more difficult for those who use assistive technology in terms of screen reader software or screen magnifiers, for example.

Also, the traditional teaching techniques are mainly based on visual models to aid in the understanding of complex information using diagrams, flowcharts, tables, and images. Unfortunately, this type of resource is not useful for students who are visually impaired as they do not allow the use of concrete resources or assistive technology [Al-Ratta and Al-Khalifa 2013]. People who are visually impaired need complementary support of non-visual stimuli to perceive the environment and to be able to construct mind maps of the environment. Receiving information from space through other senses, such as hearing and touch, collaborates with the creation of mind maps as a representation of the environment [Lahav et al. 2008]. The inclusion of multimodal

DOI: 10.5753/cbie.sbie.2019.1231

interfaces increases the user's ability to guide and navigate the robot within an environment [Lahav and Mioduser 2003, Yu et al. 2006]. The programming environment should use different soundtracks to describe the movements, the location of the robot, and the objects around it. Furthermore, the robot should be easy to handle, so that the blind user can recognize the parts and confirm the sound information through tactile feedback [Ludi and Reichlmayr 2011].

Based on this scenario, the main objective of this research is to identify the state of the art in programming teaching techniques for people who are visually impaired, using robotics as an aid tool. In 2016, Oliveira et al. [2017] carried out a systematic literature review (SLR), in which they located nine papers on this topic. The current work aims to replicate the protocol used in this study, identifying the publications between 2016 and 2019. Because the area of robotics and programming evolves rapidly, the intention is to verify new technologies and ways of teaching.

This document is organized as follows: Section 2 presents the planning and execution of the systematic literature review; Section 3 shows the results obtained; and finally, Section 4 presents the conclusion of this work.

2. Methodology

An SLR is a methodical process to "identify, evaluate and interpret the available scientific evidence relevant to a specific topic of interest" [Brereton et al. 2007]. The purpose of this systematic review was to update the studies of [Oliveira et al. 2017], which aimed to help "identify and understand methodological procedures which make use of robotics as support to the teaching of programming to people who are visually impaired". In this way, the following steps were applied: planning, execution, and results.

2.1. Planning

We used the same protocol created by [Oliveira et al. 2017]. Thus, the objective of the research, search strategies, and selection criteria will be presented. The study has as primary research questions (RQ1) and secondary ones (a, b, c, d), which follow:

- RQ1 - Which methodological procedures are being used in the teaching of programming with robots for people with visual disabilities?
  - a) Which are the methodologies for teaching robot programming to people with visual disabilities?
  - b) What are the characteristics of the programming environments used by people with visual disabilities?
  - c) What are the examples of good practices for teaching robot programming to people who are visually impaired?
  - d) Which were the difficulties/limitations in the use of robotics as support to the teaching of programming to people who are visually impaired?

The research sources used were the following digital libraries: ACM Digital Library[1], ScienceDirect[2], Scopus[3] e IEEExplore[4]. Table 1 shows the search string, with

[1] http://dl.acm.org
[2] http://www.sciencedirect.com/
[3] https://www.scopus.com/
[4] http://ieeexplore.ieee.org

Table 1. Search string

Keywords	Synonyms and additional terms
Visual impairment	*(blind OR blindness OR visually impaired OR visual impairments OR student disability OR unsighted pupils OR visual disability)* AND
Robotics	*(robot OR robotic OR robotics)* AND
Programming	*(program OR programming)*

Table 2. Selection Criteria

Inclusion Criteria	Exclusion Criteria
I1) The result must be in the English language.	E1) In case of similar or duplicate studies, only the most recent will be considered.
I2) The result must be completely available.	E2) Results that deal with teaching programming comprehensively and not aimed at people who are visually impaired.
I3) The result must have in the title, keywords or the abstract some relation to the theme of this research (programming, robotics, and people who are visually impaired).	E3) Results that deal with comprehensive and non-targeted robotics for people who are visually impaired.
	E4) The result is not from the field of Computer Science or Engineering.

keywords (terms and synonyms) combined with boolean operators. We adapted this string for each digital library, without changing its logic value. Table 2 presents the selection criteria used for inclusion or exclusion of publications.

2.2. Execution

In this step, we selected the publications based on the protocol shown in the planning stage. Thus, we extract the studies according to the following procedure: the search string was applied in June 2019 in each database, restricting the search to published papers from the year 2016 until 2019. The result was exported to the Bibtex format so that it could be imported into the StArt tool[5], software that supports the organization of systematic reviews. First filtering was performed on the results by one of the authors, checking the title, keywords, and summary, applying the selection criteria. Next, second filtering was performed by another author who confirmed the selected articles. The articles with the accepted status were extracted to be read entirely.

3. Results

In this step, we present the answers to the primary research question previously quoted in this paper: "Which methodological procedures are being used in the teaching of programming with robots for people with visual disabilities?" For this, the answers to the secondary questions will be provided in order to answer the primary research question. The search in each digital library brought a total of 141 articles. After applying the selection criteria, 6 articles were accepted, 122 rejected, and 13 were duplicates. Table 3 shows the selected articles and publication places. Among the 6 accepted papers, 5 are from the digital library Scopus and 1 from ACM Digital Library.

Concerning **question a**, workshops for programming teaching with robotics continue to be one of the most used methodologies [Paramasivam et al. 2017,

[5]http://lapes.dc.ufscar.br/tools/start_tool

Table 3. Selected publications

Reference	Conference/Journal
[Motoyoshi et al. 2016]	Lecture Notes in Computer Science
[Paramasivam et al. 2017]	ACM Technical Symposium on Computer Science Education
[Barros et al. 2017]	IEEE Latin America Transactions
[Molins-Ruano et al. 2018]	Computers in Human Behavior
[Ludi et al. 2018]	ACM Technical Symposium on Computer Science Education
[Tsuda et al. 2018]	Lecture Notes in Computer Science

Barros et al. 2017, Molins-Ruano et al. 2018, Ludi et al. 2018]. The number of workshops was varied in each work, and the use of 1-7 workshops was reported. In most of the works the students were organized into groups of 2-3 students to carry out the activities [Paramasivam et al. 2017, Molins-Ruano et al. 2018, Ludi et al. 2018].

The authors [Paramasivam et al. 2017, Molins-Ruano et al. 2018] included in their workshops students with different disabilities. A report of two editions of a robot programming teaching workshop with duration of 5 and 6 days was presented in [Paramasivam et al. 2017]. The activities were carried out with students who had several disabilities, such as deafness, low vision, blindness, cerebral palsy, muscular dystrophy, Ollier's disease, attention deficit, Asperger's syndrome and other aspects of autism or learning ability. The total of participants was 11 students (8 men and 3 women), of whom 4 had some prior knowledge of programming and 7 had no prior programming experience. As methodology, students were initially taught basic concepts of robotics and programming. The learning methodology consisted of minimizing expository class time and maximizing practical activities. Initially, students were introduced to the robotic platforms that would be used and to some examples of program and program execution in the robot. After the examples, the students received off the shelf programs and were instructed to make their modifications to the programs and execute them on the robots. The students worked in groups or individually.

The authors [Molins-Ruano et al. 2018] held a robotic workshop lasting 90 min, and students worked in groups of 2-3. It was attended by 19 students. None of them reported having any previous knowledge about programming or robotics. It was a diverse group, as 12 of them were people with functional or intellectual disabilities. There were students who are blind, intellectual disability, deaf-mute, motor deficiency, and autism spectrum disorders. The objective was to introduce the group to programming concepts, such as understanding that a computer executes orders previously given by a programmer; basic use of predefined variables and functions; understanding of flow control structures: (a) Conditional jumps, (b) deterministic loops; grouping and reusing code defining functions; realize that these notions are the backbone of all existing complex software.

The authors [Ludi et al. 2018] performed 3 workshops lasting 4 days, where the students also were organized in groups of 2-3. The authors organized the Lego's pieces on a tray so that they were easily identifiable by visually impaired students. The students in the study formed groups so that they could compare, contrast and identify individual parts of the Lego. Students were able to practice how the pieces interact with each other and their functions. The pilot experiment created detailed instructions in a text about how to build the robot models that were tested by the visually impaired students group. Three groups of students (17 men and 2 women in total) 42% were blind, and the rest had vision

limitations. Most had little or no experience with robotics. According to the authors, programming robots requires students to decide what the program will accomplish, including instructions in the correct order and understanding how to use the syntax and commands of a specific language. This is also called top-down planning. Programming Top-down refers to what the program is designed to do. Programming Bottom-up refers to the syntax of the programming language; What programming commands do and how to use them; steps for programming and debugging; programs that need to work together. Regarding the methodology, both definitions should be considered in planning and programming teaching for people who are visually impaired.

In the work [Barros et al. 2017] workshops were carried out with and without the programming tool. They carried out 7 workshops lasting 3 hours each. In each of these moments, they followed three phases of participant observation: descriptive observation, focused observation, and selective observation. Workshop without the tool was performed with 7 students, only one who is visually impaired, aged between 5 and 7 years. The methodology consisted of weekly meetings lasting three hours and is divided into three stages. In the first, the teacher explains the subject addressed in that workshop and proposes a challenge to be solved. In the second moment, the teacher asks the children if the robot can help solve the proposed challenge. Finally, it is proposed a model of robot assembly that must be assembled and programmed or controlled to solve the challenge. In the third stage of observation, called selective observation, they searched for requirements for the tool. In the workshops with the tool: once developed CardBot 2.0, they made four new experiments following the same phases of the observation of the participant: the first with a class of non-disabled students performed in a robotics event; the second with a totally blind disabled person; the third with a poor with low vision; and the fourth with a mixed class of fifteen students in which five are blind.

In the studies [Motoyoshi et al. 2016, Tsuda et al. 2018], experiments were carried out. In [Motoyoshi et al. 2016] they evaluated the P-CUBE tool which is a programming language created in Japan that uses wooden programming blocks with tactile information, with 7 people between 14 and 21 years old. These users experimented with programming exercises with P-CUBE and Arduino Sketch. In [Tsuda et al. 2018] they performed an experiment to evaluate P-CUBE2, an evolution of P-CUBE. Eight students with no programming skills participated in the experiment. The students performed activities with P-CUBE2 and Scratch, and answered questionnaires about the use.

Regarding **question b**, some works have created their own robots using low cost materials [Motoyoshi et al. 2016, Tsuda et al. 2018, Molins-Ruano et al. 2018]. The works [Ludi et al. 2018, Barros et al. 2017] use the Lego Mindstorm NXT robot, whereas the authors [Paramasivam et al. 2017] have used Turtlebot. Figure 6 shows some of the robots constructed (Figure 1 a, b) of the Lego Mindstorm NXT (Figure 1 c), and Turtlebot (Figure 1 d, e).

For programming, tangible materials continue to be used as the wooden programming blocks [Tsuda et al. 2018, Motoyoshi et al. 2016], but there have been added more functions [Tsuda et al. 2018]. The authors [Barros et al. 2017] innovated by using tangible assistive languages that use cards in geometric forms, a mobile application. Languages based on the Logo were also used [Paramasivam et al. 2017, Barros et al. 2017, Molins-Ruano et al. 2018]. Table 4 shows the programming commands for each work.

Figure 1. (a) controllable object [Tsuda et al. 2018], (b) Tortoise robot [Molins-Ruano et al. 2018], (c) CardBot2 [Barros et al. 2017], (d) Turtlebot with dust brusher, and (e) Turtlebot with tray to transport objects.

Table 4. Language Commands

Reference	Commands
[Motoyoshi et al. 2016]	*Movement (forward, right, left, backwards), Timer (1st,2nd, 3rd, 4th), If e Loop.*
[Paramasivam et al. 2017]	*goTo(location), say(text), displayMessage(message), askMultipleChoice(message, choices, timeout), moveForward(speed), moveBack(speed), turnLeft(speed), turnRight(speed)*
[Barros et al. 2017]	F - front, D - right, E - left, P – stop, La – turn on blue light e Lv – turn on red light.
[Molins-Ruano et al. 2018]	*forward(units 1= x), back(units = x), right(degrees = x), left(degrees = x), if, pen_up(), pen_down(), obstacle(), while times*
[Ludi et al. 2018]	*start_robot(), move_up(X), turnleft(X), turnright(X), stop_robot()*
[Tsuda et al. 2018]	*Hiragana, Function, If, Loop*

Among the programming blocks, the authors [Motoyoshi et al. 2016] devised two tools, the P-CUBE (Figure 2 b) and PRO-TAN consisting of a program panel, cards, and a PC (Figure 2 a). The program panel is made of expanded polystyrene boards and a magnet sheet and has ten frames that have RFID readers under the panel. The programming panel is smaller and lighter than the P-CUBE programming belt. The programming boards have four types of cards, such as the P-CUBE programming blocks. Unfortunately, since programming boards do not have tactile information like the P-Cube, the use of visually impaired individuals is not considered. The authors consider that PRO-TAN has the same characteristics as P-CUBE because it has the same system configurations and operations. Only P-CUBE was evaluated at work.

The same previous research group [Tsuda et al. 2018] presented P-CUBE2 (Figure 3), an evolution of P-CUBE. In addition to the commands already in the P-CUBE as selection (IF) and Loop, the block Hiragana [6] is added so that the user can learn the concept of subroutines. The shape of the Hiragana block is a beveled wooden cube at the top of it. A Hiragana block is assigned to a line of characters between 50 letters of Hiragana. Each surface of the Hiragana block is attributed to a Hiragana pronunciation. For example, the upper surface of the block indicates the “KA” sound and each side surface indicates “KI”, “KU”, “KE”, and “KO”. Also, function blocks have been added, F1 to F4 are defined in the side function blocks as the function name. The functional block has the same shape as the Hiragana block. By these operations, the user can control a robot that produces sound by the controlled object.

[6]Japanese characters from which it is possible to write all Japanese words

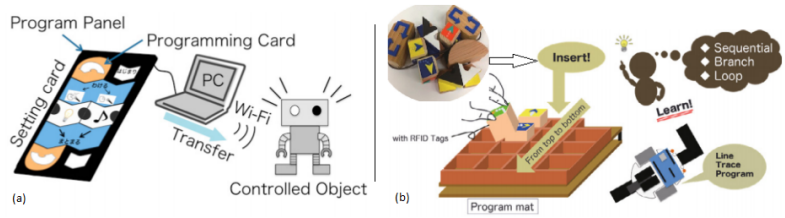

Figure 2. (a) PRO-TAN and (b) P-CUBE [Motoyoshi et al. 2016]

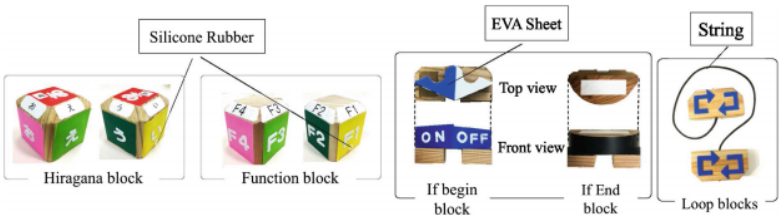

Figure 3. P-CUBE2 programming blocks [Tsuda et al. 2018]

The authors [Barros et al. 2017] described CardBot an environment for creating tangible assistive languages that use cards in geometric shapes and a mobile application. The teacher creates the alphabet of the language that will be used by the students. From this, the student develops a program by organizing the geometric cards. These cards have markings on the actions that the robot must perform. These markings are read using a mobile application in order to compute the location of each card.

In the study [Paramasivam et al. 2017], the authors' interest in boosting students' learning through the use of robots capable of performing practical tasks that can assist them in daily life, such as guiding people from a place to the other, to load and deliver objects, to clean areas of an environment and to annotate user requests. For the execution of these tasks, programming teaching activities are performed using a Turtlebot [7]. This robot is widely used for prototyping and testing activities in various areas of mobile robotics, and usually, through programming, this robot becomes capable of performing autonomous navigation in real environments, acting manipulating objects in the environment and also interacting with other robots or human beings. However, programming of these activities generally requires a high level of programming knowledge. In this study, the authors make it easy to program these activities in Turtlebot using an extra layer of software abstraction that allows programming using the CodeIt language. Also, Turtlebot has been equipped with utensils useful for loading objects (e.g., basin - see Figure 1e) and tools (e.g., dust brusher - see Figure 1d).

In the study [Ludi et al. 2018], the JBrick software is used for programming the Lego Mindstorm NXT robots. The language used was NXC, similar to the C language. JBrick is a text-based programming interface, unlike other block-based beginner languages. JBrick has accessibility also for users who are visually impaired. Accessibility includes screen reader compatibility, upgradeable braille displays, listenable line number, audio feedback, keyboard shortcuts, highlighting of the current line (for navigation), and jump directly to line with error compilation. Besides, it also has several tutorials for

[7] https://www.turtlebot.com/

introducing programming using JBrick and working with Lego Mindstorm NXT robots.

The authors [Molins-Ruano et al. 2018] have created Phogo a Logo-based language that use Python, and by a robot in 3D printing and Arduino, that is easy to build and control. The robot called Tortoise has a pen for drawing shapes and can be commanded from a computer using a transparent wireless link for students (see Figure 1b).

Concerning **question c**, in the study [Oliveira et al. 2017] 34 good practice recommendations were identified. In the current study, 9 new recommendations to be added in the initial recommendations have been identified. These have already been organized into the existing categories.

- **Workshop preparation (4)**
  - Organize the robot parts in a sectioned tray, grouping similar pieces so that they are more easily identifiable by students who are visually impaired [Ludi et al. 2018].
  - Provide different tactile and visual information so that students with different visual impairments can more easily identify the different types of robot parts [Ludi et al. 2018].
  - Organize the environment with learning stations for each category of parts, when there are large groups of students [Ludi et al. 2018].
  - Provide a non-intrusive web-based programming environment so students can perform programming using their computers with assistive tools such as zoom, screen reader, and more [Paramasivam et al. 2017].
- **Content and activities (2)**
  - Provide robots capable of performing more complex, realistic and useful activities to increase engagement and motivation [Paramasivam et al. 2017].
  - Offer different activities so that students can choose to participate in those that interest them most or that is more appropriate for their abilities [Molins-Ruano et al. 2018].
- **Work Dynamics (2)**
  - Guide individually and personalized each student, when working in groups, to solve their programming difficulties and gradual evolution in the level of difficulty [Molins-Ruano et al. 2018].
  - Encourage collaboration among students in the same group, thus increasing the chances of success [Barros et al. 2017].
- **Data acquisition and instruments (1)**
  - Observe and note usability issues and level of student engagement related to the following steps: robot design, construction, and programming [Ludi et al. 2018, Barros et al. 2017].

Regarding **question d**, the authors [Paramasivam et al. 2017] reported that the flow of activities in the first edition (2015) was slow due to students' reliance on instructors to execute their codes on the robot. This is improved in the second edition where students can autonomously run their tests on the robots, without the need to rely on the instructor.

In the study [Barros et al. 2017], they report that students who are visually impaired had difficulty knowing in which part of the lane the robot was. This issue was

minimized when the student sat down by the lane to listen to the robot's engines and know where it was. Another problem presented was the use of smartphones with assistive technology that ended up distracting and even confusing some of the inexperienced students in assistive technology. In the study [Molins-Ruano et al. 2018] they report that because of the use of a textual programming language (i.e., Python), their platform is less suitable for use by young people under the age of 14.

4. Conclusion

The results obtained in this literature review showed that the most used methodologies are still programming and robotics workshops, with activities in groups. Understanding the top-down and bottom-up planning concepts were essential to effectively map students' learning, as well as interviews and questionnaires at all stages. Also, make time for students to explore the robot pieces in groups and allow interaction between them.

Regarding the characteristics of the environments, some works have reported the use of own robots of low cost. Meanwhile, Lego Mindstorm NXT material is still used. As a novelty, authors are interested in using modern robots and programming languages with more significant support by the community of users, developers, and maintainers, and are also concerned with developing more practical and useful activities for the students who are visually impaired daily life. This was our differential from the work of [Oliveira et al. 2017], in which the use of more playful activities with the users was encouraged. Also, students were able to use their computers in the activities, accessing the environment via the web.

Acknowledgments

The work was financed by CNPq - 20/2016-6 - 442126 and CAPES - Finance Code 001.

References

Al-Ratta, N. M. and Al-Khalifa, H. S. (2013). Teaching programming for blinds: A review. In *International Conference on Information and Communication Technology and Accessibility*, pages 1–5.

Barros, R., Burlamaqui, A., Azevedo, S., Sá, S., Gonçalves, L., and Burlamaqui, A. (2017). Cardbot-assistive technology for visually impaired in educational robotics: Experiments and results. *IEEE Latin America Transactions*, 15(3):517–527.

Barros, R. P., Torres, V. P., Burlamaqui, A. M. F., and Natal, R. (2014). Cardbot: Tecnologias assistivas para imersao de deficientes visuais na robótica educacional. In *Workshop de Robótica Educacional*, pages 11–16.

Benitti, F. B. V., Vahldick, A., Urban, D. L., Krueger, M. L., and Halma, A. (2009). Experimentação com robótica educativa no ensino médio: ambiente, atividades e resultados. In *Workshop de Informática na Escola*, volume 1, pages 1811–1820.

Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M., and Khalil, M. (2007). Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software*, 80(4):571–583.

Kane, S. K. and Bigham, J. P. (2014). Tracking stemxcomet: teaching programming to blind students via 3d printing, crisis management, and twitter. In *ACM Technical Symposium on Computer Science Education*, pages 247–252.

Klassner, F. and Anderson, S. D. (2003). Lego mindstorms: Not just for k-12 anymore. *IEEE Robotics & Automation Magazine*, 10(2):12–18.

Lahav, O. and Mioduser, D. (2003). A blind person's cognitive mapping of new spaces using a haptic virtual environment. *Journal of Research in Special Educational Needs*, 3(3):172–177.

Lahav, O., Schloerb, D. W., Kumar, S., and Srinivasan, M. A. (2008). Blindaid: A learning environment for enabling people who are blind to explore and navigate through unknown real spaces. In *Virtual Rehabilitation*, pages 193–197.

Ludi, S., Bernstein, D., and Mutch-Jones, K. (2018). Enhanced robotics! improving building and programming learning experiences for students with visual impairments. In *49th ACM Technical Symposium on Computer Science Education*, pages 372–377.

Ludi, S. and Reichlmayr, T. (2011). The use of robotics to promote computing to pre-college students with visual impairments. *Trans. Comput. Educ.*, 11(3):20:1–20:20.

Meira, M. C., Lima, M. S. S., and Borges, M. A. F. (2016). Torneios baseados em robocode para incentivar jovens a aprender programação. In *Congresso da Sociedade Brasileira de Computação*, pages 2403–2412.

Molins-Ruano, P., Gonzalez-Sacristan, C., and Garcia-Saura, C. (2018). Phogo: A low cost, free and "maker" revisit to logo. *Computers in Human Behavior*, 80:428–440.

Motoyoshi, T., Tetsumura, N., Masuta, H., Koyanagi, K., Oshima, T., and Kawakami, H. (2016). Tangible gimmick for programming education using rfid systems. *IFAC-PapersOnLine*, 49(19):514–518.

Oliveira, J. D., de Borba C., M., de Morais A., A., and Harb M., I. (2017). Teaching robot programming activities for visually impaired students: A systematic review. In *International Conference on Universal Access in Human-Computer Interaction*, pages 155–167.

Paramasivam, V., Huang, J., Elliott, S., and Cakmak, M. (2017). Computer science outreach with end-user robot-programming tools. In *ACM Technical Symposium on Computer Science Education*, pages 447–452.

Sutherland, C. J. and MacDonald, B. A. (2018). Naoblocks: A case study of developing a children's robot programming environment. In *International Conference on Ubiquitous Robots*, pages 431–436.

Tsuda, M., Motoyoshi, T., Sawai, K., Tamamoto, T., Masuta, H., Koyanagi, K., and Oshima, T. (2018). Improvement of a tangible programming tool for the study of the subroutine concept. In *International Conference on Computers Helping People with Special Needs*, pages 611–618.

Yu, W., Kuber, R., Murphy, E., Strain, P., and McAllister, G. (2006). A novel multimodal interface for improving visually impaired people's web accessibility. *Virtual Reality*, 9(2-3):133–148.