Accessibility of Block-Based Introductory Programming Languages and a Tangible Programming Tool Prototype

Emmanuel Utreras and Enrico Pontelli

New Mexico State University, Las Cruces, NM 88003, USA
eutreras@nmsu.edu, epontell@cs.nmsu.edu

Abstract. Visual programming languages (VPLs) were designed to assist children in introductory programming courses. Unfortunately, despite the positive results in teaching, VPLs are believed to be inaccessible for children with visual impairments and low vision due to the dependency of visual graphics as both input and output methods. To identify the barriers that users with visual impairments and low vision face while using Block-based programming environments, as well as to acquire feedback regarding the design of a new tangible programming tool prototype, a usability study was conducted which involved nine adult participants with visual impairments and low vision. This paper presents the findings of this usability study and provides a list of features that are needed in order to make Block-based environments accessible. Furthermore, based on observations, interviews, and post-surveys this study demonstrates that our prototype can be used by users with visual impairments and low vision and provides a guideline for the design of tangible interfaces to teach programming concepts.

Keywords: Introductory programming · Low vision · User study · Visual impairment

1 Introduction

Learning programming concepts plays an important role in children’s learning process. However, teaching programming to children is not an easy task due to the complexity of the courses and languages syntax. To remedy this problem, researchers have developed visual programming languages (VPLs) (e.g., Scratch, Alice, Lego NXT, App Inventor) which make use of an intuitive graphical syntax and fun projects to encourage children in learning to code. One of the commonly used approaches of VPLs is Block-based programming, which allows the user to build code by dragging, dropping, and connecting visual blocks. The blocks represent different program constructs, and the shape of the blocks facilitate the correct combination of such constructs in the construction of a program.

Supported by the National Science foundation 1914635, 1401639 and 1757207.
Typically, the execution of such programs result in some form of visual output, such as graphics, animations, or control of the movements of a robot.

Unfortunately, despite the positive results shown in the teaching field, VPLs are believed to be largely inaccessible for children with visual impairments (VI) and low vision (LV) due to the dependency on visual constructs in both the construction of the program as well as in the output of the execution. Additionally, most VPLs are incompatible with assistive technologies such as screen readers [5]. These issues might have a negative impact on the learning process of children with VI and LV.

To identify the barriers that users with VI and LV face while using VPLs, and more specifically block-based introductory programming languages, we present a usability study designed to address the following research questions:

- **RQ1**: Can a VPL environment like Scratch\(^1\) be accessible to users with VI and LV?
- **RQ2**: What are the barriers that users with VI and LV face using Scratch?

These questions are aimed at informing the design of a novel teaching platform, a tangible programming tool (TPT) environment; our study includes the following questions about our first TPT prototype:

- **RQ3**: Is our TPT prototype accessible for users with VI and LV?
- **RQ4**: What barriers do the participants face while using the TPT prototype?

This study explores these questions by performing a usability study with nine participants with VI and LV. The study helps in identifying features that Scratch needs in order to be accessible. Furthermore, this study demonstrated that our TPT prototype can be used by users with VI and LV. In addition, the study provides guidelines for the design of tangible interfaces to teach programming to children with visual impairments.

2 Background

Block-based programming is an approach used by most educational VPLs. A block-based programming environment enables the construction of a program as the process of assembling visual blocks on the screen. Each block has a specific shape (like jigsaw puzzle pieces) and represents a specific programming constructs (e.g., conditional, loop). The shape of the block constrains how the block can be used in a program, thus facilitating the construction of programs that are “syntactically” correct.

VPLs have proven to be effective in the teaching field; indeed, there now exist a wide range of educational VPLs, focusing on a variety of application domains. The main objective of these environments is to aid students in introductory programming courses such as object-oriented programming [2] and data science [1,8]. Scratch [6] is a VPL environment designed for children in K-12 grades,

\(^1\) We use Scratch as a representative block-based language, due to its popularity.
that became very popular due to its wide availability and the presence of a rich literature and collections of educational modules. Scratch allows students to build animated novels, 2D games, and music projects. Unlike Scratch, Lego NXT-G is a block-based environment whose programs are used to control a robot’s movements instead of visual graphics.

The challenge of accessibility of teaching programming environments has been addressed by researchers. Blocks4All [7] is an environment that provides the visually impaired students with audio cues to connect the blocks. P-Cube [3] is a TPT that uses physical cubes to program a robot’s movements. The robot provides audio feedback announcing its position. On the other hand, Torino [9] is a tangible tool that uses audio sounds and audio story as an output method, and enables collaboration between students with diverse VI and students with no VI.

3 Tangible Programming Tool

Although the purpose of this paper is to report the findings in the user study, we will briefly overview our prototype design. Our project aims to develop a TPT to aid children with VI and LV to learn basic programming concepts using music as an output method. To build our prototype we used modified Duplo Lego blocks (height = 25.80) and a Lego panel (8 rows × 9 columns). Each 2 × 2 block is modified with a 3.5 mm male audio jack connector which connects to a female audio jack placed in the Lego panel. Each Lego block represents a different instruction to build a program. For the identification of each block we used a 3D printed symbol on top and a Braille label in front. The Lego panel is divided into cells to facilitate the block’s connection and the physical assembling of a program. The semantics of program is described in terms of music and the execution of a program is a melody.

4 Usability Study

4.1 Evaluation

The study consists of two parts. The first part is aimed at assessing the accessibility of Scratch, a block-based programming environment. The study consists of a training session, where the moderator teaches the participants how to use Scratch’s interface and basic programming concepts, such as loops, conditional statements, and variables. A task completion section was implemented to evaluate the following features of the Scratch interface: (1) labels of the blocks (names of the blocks), (2) Directional arrows (rotational arrow symbols), (3) the indication of correct placement for a block connection (block’s background changes to gray, simulating shadow), (4) parameter form (Scratch provides a geometric form to connect parameters), and (5) graphic feedback to debug code.

The tasks included different scenarios which present a problem to the participants and pre-built programs where they must complete or fix it. Each task...
had a time limit of five minutes. To evaluate Scratch environments we used success (completed with no help), partial success (completed with help regarding concepts), and failure (not able to complete) of the tasks. The tasks’ order was set by using balanced latin square to eliminate bias in results. The screens were recorded in video format while time to complete the task was measured manually. Lastly, a post-survey was provided to participants to gauge their overall experience and ask for suggestions related to Scratch’s environment. A screen reader (NonVisual Desktop Access) and screen magnifier were provided to the users.

In the second part of the study, the participants assessed a preliminary version of our TPT prototype, performed three different tasks, and were interviewed by the moderator.

The first step involved allowing the users the opportunity to explore the prototype. This provided us information regarding exploration techniques of the participants to identify and associate physical objects such as Lego blocks, panel, and audio jack connectors. It additionally gave us feedback on hardware’s improvements that must be implemented. The second step involved completion of the following tasks:

(T1) **Lego panel connectors and cells’ height** – The participants plugged and unplugged a block into cells with different heights (8.32 mm, 11.49 mm, 14.61 mm, 17.97 mm, 21.10 mm) and chose which height was easier to connect the block. The connection’s correctness is important to read the blocks effectively. Otherwise, it will be difficult to debug a code where blocks are not in the right position.

(T2) **Block’s location** – The participant located a Lego block placed in a specific cell and identified its location.

(T3) **3D printed block’s symbols** – Before starting this task, the participants were taught about basic programming concepts (*variable, constant, loop, conditional, functions*). Once we were confident they understood the concepts, three symbols per concept were given to them. Participants chose the best symbol to represent each concept. Additionally, they were allowed to provide their symbol ideas.

Once tasks and interview sessions were completed, we showed them an example of our idea which uses tangible tools to create music as a code output. Lastly, a self-reported survey was provided for prototype evaluation and to collect their suggestions regarding the hardware and music output. Results have been collected in a video-audio format. The videos provide vital information regarding hand movements, block’s location, and tactile exploration.

### 4.2 Participants

Nine adults (Female = 5, Male = 4) with an average age of 52.6 years and differing VI and LV levels participated in this study (see Table 1). The participants were recruited through the Asociación Puertorriqueña de Ciegos inc. and by email.
Table 1. Summary of participants of the usability study. The labels were assigned randomly and do not represent the order of evaluations.

| Participants | Age | Gender | Programming experiences | Assistive technology |
|--------------|-----|--------|-------------------------|----------------------|
| P1           | 74  | M      | No                      | Screen reader        |
| P2           | 60  | M      | No                      | Screen reader and magnifier |
| P3           | 50  | F      | No                      | Screen reader        |
| P4           | 25  | F      | No                      | Screen reader        |
| P5           | 37  | F      | No                      | Screen reader        |
| P6           | no  | F      | No                      | None                 |
| P7           | 48  | F      | No                      | Screen reader        |
| P8           | 54  | M      | No                      | Screen reader and magnifier |
| P9           | 53  | M      | Yes                     | Screen reader and magnifier |

The study was performed in the facilities of the Asociación Puertorriqueña de Ciegos and library Antonio Roig located in Puerto Rico. Each participant completed the study individually. Eight participants used screen readers and three of them used magnifiers to access visual information. As Table 1 shows, participant six (P6) did not use assistive technology due to a lack of familiarity with computers. Eight participants had no experience in programming, while one was an experienced programmer. However, none of them had experience with Block-based programming or Scratch. Before the study, five participants had never heard about TPT, three knew about them but never tried, and one participant has used it.

5 Results

5.1 Part I

In the Scratch interface evaluation, only two participants were able to successfully complete the tasks. From now on we will refer to them as participant (a) and participant (b). To complete the study participant (a) used screen magnification while participant (b) used screen reader and screen magnification. As Table 2 shows both participants were able to complete all the tasks. Also, we can see that task three and four were the most time-consuming for both participants. Those tasks are related to blocks and parameter connections. Participant (a) suggested the use of a button or key to navigate through the interface since it was difficult to keep track of the mouse position while using screen magnifier. Additionally, participant (a) indicated that using sound when connecting a block will help the participant know when a block is connected. Participant (b) recommended keyboard shortcuts to facilitate the block’s connections.
Table 2. Results of the first part of the user study. Success means that participants finished the task with no help, partial success that finished with help, and fail means that participant does not finish the task in five minutes.

| Tasks | Participant (a) | Participant (b) |
|-------|-----------------|-----------------|
|       | Time (s) | Success | Partial success | Fail | Time (s) | Success | Partial success | Fail |
| 1     | 46       | x       |                 |       | 69       | x       |                 |       |
| 2     | 47       | x       |                 |       | 35       | x       |                 |       |
| 3     | 300      | x       |                 |       | 290      | x       |                 |       |
| 4     | 285      | x       |                 |       | 285      | x       |                 |       |
| 5     | 196      | x       |                 |       | 58       | x       |                 |       |

The results of the post-survey shows that although it was difficult to connect the blocks, both participants liked the use of graphics as an output.

Based on observations, results, and post-survey of the first part we found the following:

A) Focus Navigation: A navigation button or keyboard shortcut should be added to move through Scratch’s interface (Programming blocks, work-space, animation window). Participant (a) showed difficulty finding the mentioned sections in the interface while using a screen magnifier.

B) Auditory Cues: The Block-based environment of Scratch relies on text labels, symbols, and colors to identify blocks. The participants who could not complete this section claimed the incompatibility of screen readers with block labels and symbols and the absence of an alternative way to move blocks, such as a keyboard, as the main reasons for the failure.

C) Keyboard Shortcuts: The interface of Scratch depends on mouse or touch-pad devices to drag and drop the virtual blocks. Participant (b) suggested the use of keyboard shortcuts to move and connect the blocks.

D) Connection Validation: Scratch programming consists of connecting blocks to build a program. Audio feedback that indicates the right position of the connection would help users with VI and LV to know where to drop blocks.

5.2 Part II

All participants were able to successfully complete the first and second tasks, interview, and post-survey. The blocks used contain a 3.5 mm stereo audio jack inside, which requires precision to connect them. To facilitate the connections, Lego bricks were placed in the panel, which created cells and functioned as a frame. The results of the first task show that participants have difficulty connecting the blocks in cells with height of 8.32 mm and 11.49 mm. None of the participants choose the first two heights. Four participants choose 14.61 mm, two choose 17.97 mm, and three choose 21.10 mm. Two of the participants claimed that the highest cell was better, other participants commented that it was uncomfortable to disconnect the blocks due to the pressure on their fingers. Results of the
second task show that using the rows and columns method is intuitive for the participants. Once they found the block, they proceed to start counting columns and rows to indicate its position automatically. The third task was a challenge for the participants, according to their reports during the interviews. However, they were excited with the 3D printed symbols. Symbol ideas were collected from people with no experience in programming. Two participants did not complete the task due to low touch sensitivity. They recommended that in addition to 3D symbols we could use different block sizes (heights) to identify them. Additionally, they recommend the use of buttons to enable multi-instruction in each block. Another participant commented that geometric figures are an easier and faster way to identify blocks. Results show that participants identify the programming concepts with the symbol of the first letter of its name (e.g., Loop (L), Variable (V), etc.). Furthermore complex symbols such as printed words, combined geometric figures in one block, or object symbols were time consuming and difficult to identify. The idea of using music as an output for the built code was accepted by the participants. For this step we used an environment called FoxDot [4] to present our idea of creating music with coding. All of the participants agreed or strongly agreed with using coding for music.

Based on results of tasks completion, post-survey, and interviews we created a guideline for the final design of our TPT:

A) Lego Panel Design: A Manhattan map design in a tangible tool prototype simplifies the movement around the panel. Counting the number of cells (horizontally and vertically), all participants were able to find a specific given coordinate. This feature is extremely important to provide good programming feedback to the users.

B) Height of the Cell Divisions: We found that a specific height is needed for two reasons: to act as a frame for the block connections as well as to highlight the cell divisions. Our results showed cell’s height of 14.61 mm are preferred by the users.

C) Connection Type: As a result of the post-survey, four participants totally agreed with the statement that connecting and disconnecting blocks was an easy task. Four participants agreed with the statement and one voted neutral.

D) Block Identification: Using 3D printing in our blocks was a good approach and a challenge as well. The majority of the participants were very excited about using a 3D printed symbol to identify each block. However, it was also a challenge to find symbols that represent programming concepts. Our results showed that simple symbols (e.g., geometric figures) are better and enhance the identification speed.

E) Music as Code Output: Although the TPT was in a preliminary stage, we showed our idea of using music as an output method. Five participants agreed and 4 totally agreed to the recommendation of music as an output method from execution of a program.
6 Conclusion

Based on the results of the first part of the study, we can conclude that it is possible to make VPL accessible to users with VI and LV by including the following: focus navigation to move between interface sections; audio effects to indicate a valid connection; and compatibility of screen readers with block labels and symbols (e.g., directional arrows). Some of the barriers that participants faced while using Scratch included navigating between interface sections as well as getting feedback regarding valid positions to connect a block.

The second part of this study establishes guidelines for building a successful TPT and demonstrates that our prototype can be used by users with VI and LV. Some of the barriers that participants encountered while using our prototype include the identification of blocks with complex 3D symbols and connecting blocks in the Lego panel with low cell heights. The next step of this ongoing project is to implement participants’ recommendations, including the music environment to build an efficient TPT.

References

1. Broll, B., et al.: A visual programming environment for learning distributed programming. In: Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education, pp. 81–86 (2017)
2. Cooper, S., Dann, W., Pausch, R.: Using Animated 3D Graphics to Prepare Novices for CS1 (2010). https://doi.org/10.1076/csed.13.1.3.13540, https://www.tandfonline.com/action/journalInformation?journalCode=ncse20
3. Kakehashi, S., Motoyoshi, T., Koyanagi, K., Ohshima, T., Kawakami, H.: P-CUBE: Block type programming tool for visual impairments. Proceedings - 2013 Conference on Technologies and Applications of Artificial Intelligence, TAAI 2013 pp. 294–299 (2013). https://doi.org/10.1109/TAAI.2013.65
4. Kirkbride, R.: FoxDot: live coding with Python and supercollider. In: Proceedings of the International Conference on Live Interfaces, pp. 194–198 (2016)
5. Ludi, S.: Position paper: towards making block-based programming accessible for blind users. In: Proceedings - 2015 IEEE Blocks and Beyond Workshop, Blocks and Beyond 2015, pp. 67–69. Institute of Electrical and Electronics Engineers Inc., December 2015. https://doi.org/10.1109/BLOCKS.2015.7369005
6. Maloney, J., Resnick, M., Rusk, N., Silverman, B., Eastmond, E.: The scratch programming language and environment. ACM Trans. Comput. Educ. (TOCE) 10(4), 16 (2010)
7. Milne, L.R., Ladner, R.E.: Blocks4All: overcoming accessibility barriers to blocks programming for children with visual impairments. In: Conference on Human Factors in Computing Systems - Proceedings, April, vol. 2018. Association for Computing Machinery, April 2018. https://doi.org/10.1145/3173574.3173643
8. Rao, A., Bihani, A., Nair, M.: Milo: a visual programming environment for data science education. In: 2018 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), pp. 211–215. IEEE (2018)
9. Thieme, A., Morrison, C., Villar, N., Grayson, M., Lindley, S.: Enabling collaboration in learning computer programming inclusive of children with vision impairments. In: Proceedings of the 2017 Conference on Designing Interactive Systems - DIS 2017, pp. 739–752 (2017). https://doi.org/10.1145/3064663.3064689, http://dl.acm.org/citation.cfm?doid=3064663.3064689