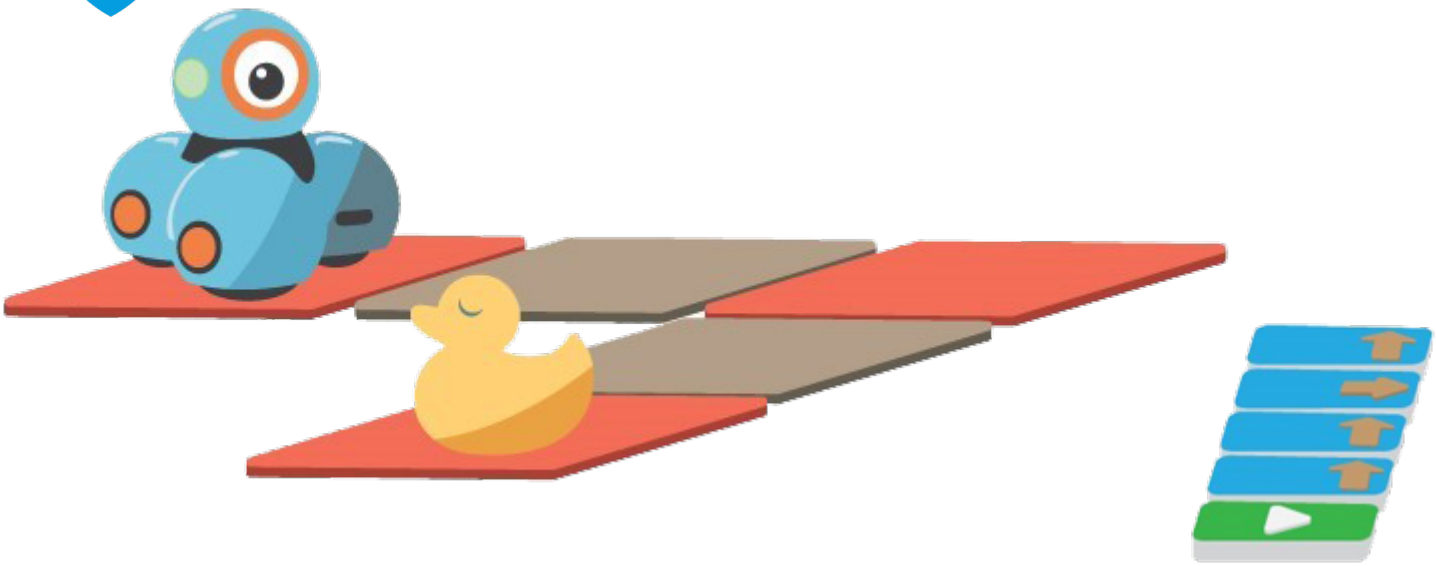


Using Robots to Make Computational Thinking Accessible to Children With Visual Impairments

Filipa Gravato de Matos de Sousa Rocha

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Hugo Miguel Aleixo Albuquerque Nicolau
Drª Ana Cristina de Oliveira Tomé Lopes Pires

Examination Committee

Chairperson: Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur
Supervisor: Prof. Hugo Miguel Aleixo Albuquerque Nicolau
Member of the Committee: Prof. João Pedro Vieira Guerreiro

September 2020

Acknowledgments

Before anyone else I would like to thank my parents for the opportunities they gave me throughout my life and the support they always showed me in my studies. I would also like to thank my sister for reminding me to be myself and strive to be better. Also to my grandparents and aunt who believed in me all these years.

I would also like to acknowledge my dissertation supervisors Prof. Hugo Nicolau and Dr. Ana Pires for their insight, support and for introducing me to this investigation area that makes sense to me.

To all the professors and investigators at LASIGE who helped me in this journey. A special thanks to Prof. Tiago Guerreiro, Hugo Simão, Dr. António Barros and Lúcia Abreu. And to all the teachers and educators who participated in the studies. In particular, Dr. Cristina Mota, who introduced me the theme and was a big support through it all.

I want to thank Henrique Santos for his unconditional support. Thank you for being there in the good, the bad and the impossible times.

Last but not least, my friends who were there in the good and the bad moments. Especially, Mariana, Liliana, Diogo and Tiago you made my days in Taguspark the best days.

To each and every one of you – Thank you.

Abstract

Computational Thinking is becoming a fundamental literacy skill, such as reading and writing, and expected to be used worldwide by the middle of the century. Visually impaired students can find many barriers in learning computer science. These barriers start with inaccessible tools, where the outputs are usually visually-demanding. Previous approaches already use tangibles to program, however the output usually consists of audio stories or music. To bridge this gap, we propose a fully tangible system to engage children in learning computational thinking.

We conducted interviews with computer science teachers to understand with which methods and concepts should children initiate their learning. We sought to understand the qualities and flaws present in current programming environments by conducting a focus group with educators. And a second formative study following a Wizard-of-Oz methodology with children playing with a first prototype. Finally, we conducted a remote user study with families at their homes.

We contribute with a set of qualities that programming environments should have to be inclusive to children with different visual abilities, and evidence that inclusive tangible robot-based programming is worth pursuing.

Keywords

Computational Thinking, Computer Science, Programming, Collaboration, Visually Impaired, Blind, Accessibility

Resumo

Tal como aprender a ler e a escrever, o pensamento computacional está a tornar-se uma ferramenta importante na educação das crianças. As crianças com deficiências visuais ainda encontram muitas barreiras nas ferramentas que permitem a aprendizagem destas competências. Grande parte das ferramentas têm uma apresentação visualmente exigente. Abordagens anteriores já consideram a utilização de blocos tangíveis para programar, porém o resultado consiste em histórias áudio ou música. Para preencher esta lacuna, propomos um sistema totalmente tangível que cative as crianças para aprenderem pensamento computacional.

Conduzimos entrevistas com professores para compreender os métodos e conceitos pelo os quais as crianças devem começar a aprendizagem. Com um grupo de referência de educadores identificámos as qualidades e falhas presentes nos ambientes de programação atuais. Num segundo estudo com crianças com deficiências visuais para testar um primeiro protótipo, seguimos uma metodologia *Wizard-of-Oz*. Por fim, conduzimos um estudo remoto com famílias com crianças com deficiências visuais nassuas casas.

Com este trabalho contribuímos com um conjunto de qualidades que os ambientes de programação devem ter para serem inclusivos a crianças com diferentes habilidades visuais, e evidências que ferramentas tangíveis com robôs são um caminho a perseguir.

Palavras Chave

Pensamento computacional, Programação, Colaboração, Cegos, Acessibilidade

Contents

List of Figures

List of Tables

Acronyms

APL	Audio Programming Language
CT	Computational Thinking
GDD	Global Development Delay
IDE	Integrated Development Environment
IT	Information Technology
PB	Pseudospatial Blocks
PCS	Picture Communication Symbols
SNEs	Special Needs Educators
VI	Visually Impaired
VUI	Visual User Interface

1

Introduction

Contents

Computational Thinking (CT) is becoming a fundamental literacy skill, such as reading and writing, and expected to be used worldwide by the middle of the century [1, 2]. CT "is the thought process involved in formulating a problem and expressing it in a way that a computer - human or machine - can effectively carry out" [2]. It borrows concepts from computer science [3], such as sequences, operators, and iteration as well as practices like being incremental and iterative, testing, debugging, abstracting, and reusing. The value of CT goes beyond computing contexts and promises to impact the social, emotional, and cognitive development of children [4, 5], while fostering personal and career growth [2].

There is not a direct and defined strategy to teach young children these computational concepts and practices, however most research scenarios include block-based programming languages with graphical outputs [6], for example Scratch [7]. The Scratch platform introduces many programming activities that can be personalized and modified to ones interests, keeping children engaged and motivated to build their own projects. Since the platform is block-based, the children can tinker with the different digital bricks and put together sequences of instructions, as opposed of having to learn how to read the syntax from traditional programming languages. While building their own ideas, they are actively participating in the development process and with it grows an interest to learn how to make new functionalities with new computational concepts.

Growing up, children progress from a stage of reaction to external stimulus to actively engage with the environment surrounding them and learn by interactive play [8]. Physical artifacts such as robotic kits and toys can immerse children in a playful learning environment becoming a powerful tool that promotes the development of computational thinking skills, for example problem-solving skills and thinking at different levels of abstraction [9].

Since robots can have a multi-modal interface, users can interact with the robot with no need for screen-readers, a keyboard or a mouse [8]. Navigating in the real world with the robot is a good starting point to prepare for future programming tasks [10], and contribute to the development of spatial cognition [11].

Young people in general have an interest in robotic kits and toys, so promoting these objects with Visually Impaired (VI) children with similar interests can help building inclusive learning experiences [12].

1.1 Problem

VI students can find many barriers in learning computer science. These barriers start with inaccessible tools, where the outputs are usually visual (e.g. Scratch, ScratchJr); and as any other child, a high level of abstraction is needed to understand what they are coding [13]. Graphical programming environments create accessibility barriers to blind children by heavily relying on visual elements and output [14]. Block-based languages are not accessible to screen readers and the syntax is hard to understand via audio

output.

Existing solutions are visually demanding and inaccessible to VI children [14], placing them at risk of being excluded from learning CT. Coding kits often require children to create a set of instructions that will help a character (e.g., digital avatar or robotic device) overcome a series of spatial challenges by following a given path, avoiding obstacles, and collecting rewards. We refer to these activities as spatial programming activities as they contribute to the development of spatial cognition, a critical skill for VI children [11]. Ironically, previous research efforts to make CT accessible to VI children are largely limited to sequential audio-based actions, preventing VI children from engaging in spatial programming activities as their sighted peers.

Previous approaches, such as Storyblocks [13,15], and Project Torino [16], use tangibles to program. However, the output of the activities consists of audio stories or music. Blocks4all uses a touchscreen device compatible with VoiceOver[1] to move the robot spatially. Their studies showed that VI children struggled with the activities, and additional work was required to reduce the demand associated with these challenges. We sought to bridge this gap of a fully tangible solution accessible to VI children.

1.2 Approach

In this dissertation, we propose the use of tangible blocks that build computer programs and a robotic device to render an accessible multimodal output, while engaging children in learning computational thinking. In our work, we started by conducting interviews with computer science teachers to understand with which methods and concepts should children initiate their learning. Following this, we sought to understand the qualities and flaws present in current programming environments for children regarding accessibility. First, we conducted a focus group with IT and Special Needs Educators (SNEs), where we presented four environments with at least one tangible component. The educators interacted with each and gave feedback. For the second study, we adapted blocks with additional tactile cues and audio feedback, and the Dash robot with augmented physicality, feedback, and feed-forward mechanisms. We used a Wizard of Oz methodology, where the children tinkered with the blocks, and an investigator would use the Wonder Blockly[2] app to program the robot accordingly.

Following the formative user studies, we considered the results and created a fully tangible system. The proposed solution, takes into consideration the benefits of block-based languages, and is composed of tangible blocks with embossed pictograms to represent actions, magnets and saliences to facilitate coupling and TopCode markers. An app in an Android mobile device recognizes the markers with the camera and translates to actions for the robot to execute. The robot moves on top of a tactile map and gives feedback on its actions, allowing the user to follow it by touch. We took into account that most

[1] https://en.wikipedia.org/wiki/VoiceOver
[2] https://www.makewonder.com/apps/blockly/

children find a robot engaging.

1.3 Goals

Our work aims to allow VI children to learn CT and computer programming in an accessible way. We expect to meet the following goals:

- Understand the main difficulties children have when interacting with technology and programming tools, by interviewing computer science teachers and special need educators;
- Understand the necessary qualities of a spatial programming environment with SNEs and VI students;
- Design tangible blocks that allow VI children to overcome the existing barriers while programming;
- Determine a robotic output with audio feedback that will allow VI children to understand instruction abstraction and perform spatial tasks;
- Validate our solution by performing user tests with VI children

1.4 Paper Submissions

Regarding the work carried out under this dissertation, we submitted a paper in collaboration with a team from the LASIGE[3] for the annual Interaction Design and Children 2020 conference: **Exploring Accessible Programming with Educators and Visually Impaired Children** [17] [Submitted, Accepted]

We also a submission underway to the ACM CHI Conference on Human Factors in Computing System 2021: **Enabling Spatial Programming Activities for Children with Visual Impairments and their Families** [Abstract Submitted]

1.5 Thesis Outline

This document is structured as follows. Chapter 2 sets the base to understand the computational concepts children can learn, the tools and robots available to do so and their accessibility. An examination of fundamental concepts of CT. The chapter ends with a comparison of different commercially available environments, tools, and robots. In chapter 3, we present a literature review of the works related to the learning of CT through the use of programming environments and languages. Chapter 4 reports on the formative studies conducted with SNEs and VI children to understand the qualities and flaws present in

[3] https://www.lasige.di.fc.ul.pt/

current programming environments for children regarding accessibility. First, we present the procedure and results of the focus group with Information Technology (IT) and SNEs, where four different environments were presented and the educators gave feedback. Secondly, the workshop with the VI children where a Wizard of Oz methodology was applied. The final solution is presented in Chapter 5.We present the design and implementation of the different components of BATS - a Block-based Accessible Tangible programming System. Chapter 6 reports on the final user study. A toolkit composed of BATS and a guidebook to instruct the participating families in conducting the proposed spatial activities. The document ends with a conclusion on all the developed work, limitations and possible future work in chapter 7.

2

Learning Computational Thinking

Contents

To have a better understanding of the computational concepts children can learn, the tools and robots available to do so and their accessibility, we examined fundamental concepts of computational thinking. In this chapter, we also present a comparison of different commercially available environments, tools, and robots.

2.1 Computational Thinking

There is still some misunderstanding distinguishing CT and computer science when instructing children [18]. The first is a discipline that uses computerized devices to create solutions and automate processes. CT is defined by the Computer Science Teachers Association and International Society of Technology in Education [19] as a ”problem-solving process that includes formulating problems in a way that enables us to use a computer and other tools to help solve them; logically organizing and analyzing data; representing data through abstractions such as models and simulations; automating solutions through algorithmic thinking (a series of ordered steps); identifying, analyzing, and implementing possible solutions to achieve the most efficient and effective combinations of steps and resources; and generalizing and transferring this problem-solving process to a wide variety of problems.”.

There are four essential methods in adopting CT to start solving a complex problem: decomposing a problem or system into smaller parts, recognizing patterns and similarities among different problems, abstraction from irrelevant details and focusing on the crucial information, and develop a straightforward solution by steps or develop an algorithm. Finally, these simple steps can build a computer program and solve the problem in the most appropriate way.

To adopt the previous methods it is crucial to consider the existing computational concepts that are highly useful in programming contexts. Sequences are a series of individual instructions or steps that specify the behavior or action to be produced. Loops are a mechanism to repeat a sequence of instructions multiple times. Events are the component that makes one action cause another to happen. Parallelism is the ability to have different instructions executing at the same time. Conditionals are components that allow distinct outcomes based on a defined condition. Arithmetic operators provide support to perform numeric and string manipulations. Data is the information to be stored, received, and sent.

While adopting computational methods and concepts, the developers may follow CT practices even if they are not aware. When facing a problem, designing a solution is not a straightforward sequential process. It is an adaptive process in which the solution can change in response to the obstacles found in every step of the problem. After achieving the solution, it may not work as imagined, and there is a need to test and debug to find the issues. When a solution is not easy to build, many developers resort to work from other people modifying to adapt to their problem and find a solution. Abstracting and modulating

is how ones characterize this process of building a large solution by assembling collections of smaller parts.

It is nearly impossible to learn new methodologies and not change perspectives about our surroundings and tackle the problems presented. Computational thinkers start questioning how to express their creativity using technology and developing solutions for problems they find in their world.

CT empowers the development of creativity, planning, and problem-solving skills. We can already observe the influence it has on other subjects, such as how machine learning has transformed statistics [20]. We can conclude that CT is not just a discipline but a thinking methodology. Students who learn it can start to observe relationships between the academic concepts they learn and their life events.

2.2 Block-based Tools for Learning

Computer programming can be a way of teaching computational thinking, and the majority of research scenarios include block-based languages [8].

Blocks programming languages [21] can allow beginners to construct programs without struggling with the syntax, making text-based programming more comfortable to learn in the future. These graphical programming languages have proven to be more engaging and understandable for new learners. There are learning barriers that block programming languages that attempt to minimize: selection, use, and coordination.

When constructing a program with a blocks programming language, users can rely on recognition of the categories and shape of the block instead of recalling vocabulary. Block-based code reduces the cognitive load a new programmer has to learn by simplifying the syntax grouping computational patterns into blocks. Since blocks can only be assembled in a certain way or with other specific types of blocks, the code is less likely to be syntactically incorrect. These characteristics and designs simplify the discovery and exploration of building programs for new learners that can concentrate on the meaning of the code instead of the writing notation.

There are online programming environments that make use of block-based languages to allow children to learn computer programming. To compare these environments, we have used the information available online and our experience with such environments to construct Table 2.1. In Table 2.1, we can observe that most of the environments have a visual output, except for Tynker and Tickle, that can program a drone or robot. All of them make use of block-based languages that have proven to be successful with children. The computational concepts present in the list of blocks of the different environments are diverse. However, we can recognize that all have tasks to use variables, loops, and conditionals.

We can also observe that all of them have a free version for children to explore. We can also find tangible tools to teach computational programming to children. These tools are not directed to VI

Table 2.1: Comparison of programming environments publicly available

Features Environments	Age	Concepts	Language	Output	Price
Kodu	+8	variables, branching,loops, number and string manipulation, subroutines, polymorphism	Icon-based	Object behaviour in a 3D game world	Free
Tynker	+4	variables, loops,conditionals, numbers and strings, functions	Icon-based, Block-based, Text-based	Character, Drone or Robot behaviour	10 Free Projects
Scratch	+8	variables, loops, conditionals	Block-based	Character behaviour	Free
Tickle	+4	variables,loops, conditionals	Block-based	Character, Drone or Robot behaviour	4 Free Demos

Table 2.2: Comparison of tangible tools existing in the market

Features Tools	Age	Embossing	Concepts	Output	BlockRecognition	Price *
Osmo Coding Awbie	+5	NO	sequence, loops, debugging	Character in game	Osmo Camera	187.17
Cubetto	+3	NO (yet distinguishable non-visually)	algorithms, queue, debugging, recursions and functions	Robot on Map	Blocks placed in board	+201.45
KIBO	+4	NO	algorithms, loops, conditional	Robot	Barcode Scanner	+178.21
Algobrix	+4	YES	functions, parameters, loops, conditional, algorithms	Robot	Connectors and Bluetooth	+222.99
Matatalab	+4	NO	sequence, loops, functions	Robot, Audio, Drawing	Camera	+152.15

*Prices converted

children, although they can have features that become more accessible.

To make a comparison of the tools publicly available, we have used the information available online and our experience with the tool to construct the Table 2.2. When examining Table 2.2, we conclude that none of the presented tools possesses features that could make it an accessible, tangible programming tool. Most of the tools do not have blocks with embossing nor characteristics that make them distinguishable non-visually. This way, children can not recognize the block and the scope of the program with low effort, except for Algobrix and Cubetto. None of the tools have blocks and tasks to allow children to learn all the necessary computer programming concepts. Except for Osmo Coding Awbie, the most popular output of the programming action is the behavior of a robot. For the system to recognize the blocks, all the tools have different techniques that are not comparable.

2.3 Robots for Learning

There is research regarding children robot interaction in different areas, such as children suffering from anxiety [22]. Although robots encourage interest in science and technology, there is minimal research regarding the interaction between VI people and robots.

Researchers are exploring how blind people perceive robots and how these devices fit in their expectations and fears regarding the increase of dependency in them [23]. The researchers could infer from the answers that the participants preferred to feel in control. If they were to be in contact with a robot, they preferred that it responded to commands and questions. While interacting with a robot, it was helpful to the users that it made a sound, allowing them to identify what was its position and state.

Castro-Gonzalez et al. [24] proposed a physical haptic interaction study that focuses on the communication between anthropomorphic robots and children with visual-auditory impairments in a game mode. They proposed a game interaction where the interface is the movement of a robotic arm because some users may not perceive sound or visualize lights.

Researchers in Australia presented a project [8] that uses a SONY Aibo platform targeting blind toddlers and focusing on how robots can assist in developing and learning pre-orientation. The robot is programmable, has sensors, buttons, and speakers. The play sessions between toddlers and the robot demonstrated that the children enjoyed interacting with it and adopted different postures before and after the session.

The existing robots and robotic kits in the market have different features that can make them accessible to VI children. To compare some of these robots, we used the information available on the official robot websites, reviews of users with experience, and hands-on experience to build Table 2.3. All the considered robots move in real space.

Table 2.3: Comparison of robots and robotic kits existing in the market

Features / Robot	Assembly	Age	AudioFeedback	CompanionApp	Price	Microphone	Speakers	VisualElements	Other Capacities
Fisher-Price Code-a-Pillar	YES	+3	NO	NO	40.30*	NO	YES	YES	Visual Feedback
Modular Robotic Cubelets	YES	+4	YES	YES	125.34*	YES	YES	YES	
Code&Go Robot Mouse	NO	+4	NO	NO	53.73*	NO	YES	YES	
WowWee Elmoji/Coji	NO	+4	YES	NO	26.86*	NO	YES	YES	
Botley	NO	+5	YES	NO	71.64*	NO	YES	NO	Sensors
Photon	NO	+5	NO	YES	219.90	YES	YES	YES	Sensors
Ozobot Bit	NO	+6	NO	YES	52.84*	NO	NO	YES	Sensors
Hasbro FurReal Proto MaX	YES	+6	NO	YES	23.28*	NO	YES	YES	Tactile sensors
Dash	NO	+6	YES	YES	134.32*	YES	YES	YES	Sensors
Lego Boost	YES	+7	from App	YES	143.28*	on App	on App	YES	color/proximity sensor
ArcBotics Sparki	NO	+7	NO	YES	133.44*	NO	Buzzer	YES	Arduino compatible
Anki Cozmo	NO	+8	YES	YES	161.20*	NO	YES	YES	VGA Camera, Cubes
Marty the Robot	YES	+8	NO	YES	170*	NO	Buzzer	NO	Expandable
Sphero 2.0	NO	+8	NO	YES	116.41*	NO	NO	YES	Sensors
UBTECHJimu Robot	YES	+8	YES	YES	88.66*	NO	YES	YES	Color sensor, Expandable
Tenergy Odev Tomo	YES	+8	NO	YES	62.68*	NO	NO	YES	Sensors
Ozobot Evo	NO	+9	YES	YES	88.66*	NO	YES	YES	Optical and Proximitysensors
Makeblock mbot	YES	+14	NO	YES	71.64*	NO	YES	YES	Sensors, Remote Control

*Prices converted

After analyzing the table 2.3, we conclude which robots are adequate to be used by a VI child taking into account their features. The younger the recommended age for the toolkit, the simpler it should be so that children can easily explore the interface. One of the most significant features for a robot to be accessible to VI people is possessing speakers to communicate audio feedback. This way, the users can have a perception of the position of the robot and its behavior. When the robots carry visual elements, such as light, children with low-vision have an advantage while following and finding it. The microphone can be an advantage for children to communicate with the robot and personalize the interaction. The companion applications grant another method of interaction between the user and the robot by allowing behavioral programming. However, the interaction with the applications relies on compatibility with mainstream accessibility tools in mobile devices. A robotic kit needing assembling is not as easily explored and ready to work as assembled ones.

Metatla et al. [25], used a co-design approach to design and evaluate a robot-based educational game that could be inclusive of both VI and sighted children in the context of mainstream education. They ran a focus group and co-design workshops to evaluate the existing barriers and opportunities in inclusive play, and design an inclusive educational game that could address some of the barriers outlined. They used off-the-shelf Ozobots[1], that are designed to engage sighted children, but they have also been shown to provide functional accessibility to VI adults. Children interacted with the game through tangibles and audio feedback. They were able to read the map and apply navigation instructions. From the sessions resulted a set of design guidelines that emphasise multisensory feedback, hands-on creation, and narration as a means for modulating pace and stimulation; all of which were highlighted as potential barriers to inclusive interactions.

With the previous comparisons in mind and considering the price, the top choice of a robot to use as a tool by VI children is Dash, as shown in Table 2.3.

[1] www.ozobots.com

3

Related Work

Contents

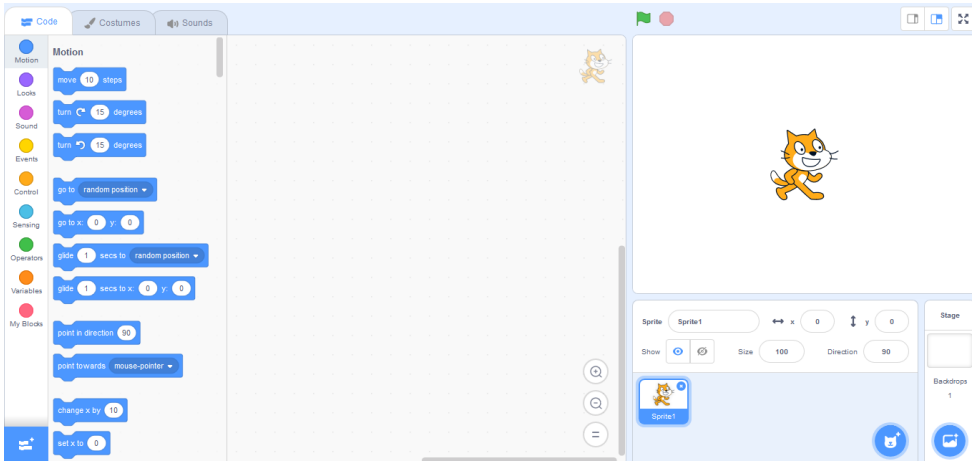


Figure 3.1: Scratch Programming Environment

In this chapter we conduct a literature review of the works related to the learning of computational thinking through the use of programming environments and languages.

3.1 Programming for Children

Scratch [7] is a visual programming environment (3.1) to learn computer programming with the goal of self-directed learning and tinkering. Children code through snapping together blocks to control sprites on screen. The platform is visually intuitive to navigate with key components always visible and with commands divided into color-coded categories. Scratch is always live with no need for a compilation step or an edit/run mode distinction. It allows the user to tinker, encouraging the learning process. This strategy eliminates syntax errors since the blocks only fit together in specific ways or with certain block types. All these features allow users to learn and evolve in programming thinking quickly.

ScratchJr [26] is the redesign of the Scratch platform for the developmental and learning needs of children from kindergarten to second grade. Previous research has shown that children as young as four years old can understand basic computer programming concepts and build and program robotics projects. ScratchJr was developed, focusing on making the interface easier to understand, allowing children to explore and develop while fostering their creativity. Similarly to Scratch, the users have an organized screen with an area for placing the blocks, a palette to explore the blocks organized by categories, and the story page to observe the output. Users can snap together blocks identified with icons to form instruction and have a visual output, thus eliminating the need to read and maintain the user engaged. They can see the connection between the instruction block and the action. To evaluate the learning with ScratchJr, the authors conducted two different pilot tests with children and observed their interaction with the tool. Overall, the children were able to follow the curriculum and learn how to

work with the tool. Illiteracy was an identified barrier for the kindergartens who could not identify blocks by the category name. As children tend to be imaginative and play freely, they would explore the tool alone during the pilot tests.

Strawbies [27] is a real-time tangible programming game, block-based as the previous languages. It was designed for children ages 5-10 and based on an iPad app that can identify the tangible blocks that construct a sequence of instructions. The goal was to make an inviting, playful and open-ended, simple but fluid system appropriate for different ages and skill levels. Tangible technology is an excellent way to introduce computational thinking since it evokes familiar teaching and learning. The tiles have a design to make them easy to assemble, and once connected, prevent the set from falling apart if dragged around. The tiles split into categories: verbs, adverbs, units of measure, an ”Always” and a ”When” tile for looping behavior and if/then, logic. The game was evaluated in six play sessions where the authors could observe the users. Results revealed that the system was engaging, as children wanted to play more and new children joined the open sessions. It was also possible to infer it was easy to understand as most children were able to take Awbie, the character of the game, to its goal, the strawberries.

TurTan [28] is also a tangible programming system. It is operational in a tabletop interface designed to overcome the main difficulties in learning computer programming. The authors identified that the main difficulties were related to syntax and the used tools than the programming concepts themselves. This resulted in using tangibles to promote the complete exploration and expressiveness of the users. The system supports seven types of instructions: move without painting, move while painting, rotate, scale, change color, repeat, and start. The interaction to program the character of the game, a turtle, begins with placing one of the tangibles, and the table recognizes it, displaying to the user visual feedback on the tabletop. For the tabletop to recognize the correct sequence, the system uses the distance between tangibles and their angle. However, users could also explicitly make a gesture to establish a connection. The authors have not made any thorough testing, but they believe the system is very easily understood.

E-Block [29] is another block-based tangible programming tool that was created for children from 5 to 9 years old. The goals in the design and implementation of this system were for the children to program with no space limitation, the system being easy to learn, and having real-time feedback for the debugging process. The system has four types of programming blocks that guide a character through a maze: start, end, direction, and sensor. When connected in sequence, the blocks connect between them and the computer to verify if they are placed correctly and give the user the proper feedback on the screen and the block. After conducting a preliminary study with six children, the authors had positive feedback since they liked the system and finished the proposed task.

T-Maze [30] is one more system that allows children to create their maze and game with tangible blocks while learning computational thinking. The system is composed of sensors, wooden blocks, and a camera connected to a computer. The goal was to develop a tool that children could play with

ease while improving logical thinking abilities, the consciousness of optimization, and raising modern science and technology. The tool uses easy-to-grab wooden blocks as they are familiar to children, to construct instructions. The camera is then able to recognize the set of instructions through the *TopCodes* encoding. There are four types of blocks: start and end, direction, loop, and sensor. The system can give the user feedback if the semantics are not correct, allowing for a more straightforward debugging process. With a preliminary user study, it was possible to infer that the children found engaging in finding blocks and thinking about what to do next. With practice and the help of the real-time feedback given by the system, all the children were able to finish the maze tasks, but still had difficulty in understanding more complex programming concepts.

The Cubetto [31] robot was designed for children. The tool does not use a screen, and it consists of a robot, a programming tray, and four categories of blocks: forward, right, left, and function. This way, the system promotes learning programming concepts such as algorithms, sequencing, and debugging with a code-by-code methodology and a robot to perform spatial activities. The activities can progress into more complex programming as the skills of the children progress.

Anzoategui et al. [31] performed a study with 21 children between the ages of 4 and 5. They measured time and the number of errors during the programming interaction between the children and the tool. The users had to perform spatial activities, such as move Cubetto from point A to B, avoiding obstacles on the map. The authors observed that the children used systemic thinking and developed the code line according to all the steps they identified, thus building a longer sequence and not using the function block to incorporate repetition. The children took 2-6 minutes to complete the tasks with 0-5 errors. The conclusions taken from this study show that younger children need a more fundamental approach in code construction to progress to more complex computer programming.

KIBO [32] is also a robotic kit that was developed considering three learning goals for the kit: foundational engineering concepts of sturdy building and construction, foundational programming skills of sequencing, repeat loops, and conditional branching, and open-ended creativity and artistic design. KIBO involves hardware parts to assemble and tangible blocks to program the robot. The tangibles are wooden blocks, for the educational familiarity children already have with these, that have drawings and text to identify their function. After constructing the code sequence, the children can scan the barcode on the block with the robot and then watch it follow the instructions. By observing and interviewing the children testing the kit, it was possible to infer that it suited the proposed learning goals. Children can learn programming concepts at a very early age.

Object-oriented programming is easily accessible to beginners since it allows modeling of real objects as software objects. Research has shown that programming can have a positive impact on the educational achievements and social-emotional interaction. Tangible Interfaces are a popular method that connects the interaction with the physical world with computer programming. With the previous

ideas in mind, TanProStory [33] is a tangible programming tool for a storytelling environment to transmit object-oriented programming concepts to children. The tool consists of different cubes that connect between them and the computer so that the children can create their characters building the story and observing the result on the computer screen. In a preliminary user study, the authors introduced the tool to children, observed, and interviewed them. The feedback from the testers was positive, and the children seemed to enjoy completing the tasks. The feedback from the tool was essential to children who had difficulties in distinguishing some blocks or forgot or got confused with the right way to build the sequence.

3.2 Non-Visual Programming

There are solutions directed to learning computational programming for VI users, which generally resort to audio output, tangibles, and assistive technologies such as screen readers.

Blocks4All [14] is a prototype environment developed to overcome the current accessible barriers for VI children in digital block-based programming environments (3.2). The application was developed for a touchscreen Apple iPad since these have a built-in screen reader and zoom capabilities, and are already popular with VI people. The goal of Blocks4All is to be used independently and be engaging to VI people or sighted people while supporting the features of block-based environments that make them suitable for young children. The researchers identified the following barriers in existing programming environments: Accessing Outputs and Elements, Moving Blocks, Conveying Program Structure, and Type Information.

To surpass the existing barriers, Milne et al. [14] used a Dash Robot as an accessible and tangible output. Blocks are accessible by VoiceOver to announce their location and type of block. An audio-based drag and drop system, or a selection of a location and block, moves the blocks. The code can have a spatial representation where the repeated loops and the conditionals have a start and end blocks. Nested statements are placed vertically, or an audio representation with start and end blocks for the encoding statements nested blocks are announced. To convey type information of the existing blocks, the user can select from an embedded drop-down menu in each block, or audio-cue typed blocks where the matching blocks in the workspace play the same audio cues as the selected block. The authors conducted a formative study with five VI children, where they observed the time the children took to program the robot to move spatially and asked for feedback on the interface. Dash was popular among the participants, and the interface was modified according to the feedback given. The spatial representation to convey program structure was preferred. The audio-cue typed blocks did not work as well without using VoiceOver. With practice, children could use the two existing methods to move blocks.

Bonk [34] is an accessible programming toolkit that enables the creation of interactive audio games based on the JavaScript programming language. Teaching programming through media has been shown

Figure 3.2: Blocks4All

to increase engagement with programming tasks. Although the VI still face inaccessible barriers, such as Integrated Development Environment (IDE), diagrams with non-visual alternatives, and difficulty navigating and debugging code. The code is JavaScript, with many convenient functions that reduce programming overload in listening to the code. The toolkit has a text-to-speech feature adjustable to the needs of the users that produce words or sound effects. The initial version of Bonk was tested in a week-long workshop with 10 VI high school students using assistive technologies to access their computing devices. Students developed different games throughout the workshop, and although everyone progressed at a different pace, all of them were able to present at least one working program. Overall, the authors found the workshop successful because even though they found some difficulties while writing code, sharing and versioning documents, and syntax errors, the students gave positive feedback and were able to create their own audio game.

Pseudospatial Blocks (PB) [35] is another block-based language, based on arrow key navigation. The arrangement is pseudo spatial as the navigation is in a two-dimensional space with operations to move right, left, up, or down. In the case of PB, the toolbox located is to the left of the working space where the blocks are stacked vertically, so regardless of the position of the block, a left movement will lead to the same location in the toolbox. In graphical environments, users can place and connect the blocks by observing their different shapes. However, in PB, the code is constructed by first selecting an insertion point to the new block, and the system will filter so that only the eligible candidates can be selected. The system produces audio as output.

Audio Programming Language (APL) [36] is based on audio interfaces to enhance problem-solving and thinking skills in teenage blind learners with computer programming. APL has two main layers: Audio Interface and Programming Logic. The first includes a circular command list and query; the second consists of four states that define the logic: run program, loop or condition, delete the last command, save command and verify the next step. The circular command list is dynamic and introduces the available commands updating as the context of the code evolves. The program can request answers to a set of questions to complete the command line at the audio interface layer. The programmer can use five commands to build the code: input, output, cycle, condition, and variable. The output of the code is always audio-based. The authors implemented a qualitative case study methodology where they observed and recorded the interactions of the users with the program. The main goal was to familiarize the users with elements and concepts used to solve problems with the APL program while learning computer programming basics. In the beginning, users had difficulties understanding the more complex concepts, such as loops and conditionals, but they could apply them and build their projects with practice. The authors believe that blind learners tend to rely heavily on experience before building abstract thinking. That is why APL is a starting step in developing higher-level cognition.

Torino [16] is a physical programming language to teach the basic programming concepts and computational thinking to children between 7 and 11 years old, regardless of their level of vision. The primary purpose is to explore collaborative learning scenarios between children with mixed visual abilities and observe how technology can participate. The system comprises instruction beads that can be physically connected and manipulated to generate sound as output. There are three types of instruction beads: play, pause, and loop, each translating to a line of code in the program. Each bead has buttons to control the repetition of the sound to be played, introducing the concept of variable, different size and shape to be better identified, and a custom circuit that allows them to connect and transmit data to its neighbors and the hub to form the sequence. In addition to the physical, spatial, and auditory configuration of the system, the authors designed a set of activities to assist the children in exploring programming concepts as commands, sequences, variables, abstraction, threading and iteration, problem-solving skills, and domain-specific vocabulary. The authors conducted evaluation sessions with ten children who had varying degrees of visual ability. The children collaborated with one of their peers in the first two sessions and one of their parents in the last session. It was possible to observe that the exploration of the beads was critical for VI children to understand their functionalities. When children physically followed the beads as the program executed, it allowed them to understand better the abstract mapping between the bead and its audio representation. In the end, the authors were able to infer that the roles of touch, audio feedback, and visual representation are essential in the design of inclusive tools for VI children.

StoryBlocks [13,15] is another tangible programming toolkit. The blocks represent story components, and the code produces an audio story or game as an output. Users assemble physical blocks that rep-

resent characters and actions to create an interactive story. Blocks are made from a low-cost material, identifiable by its shape and color of the drawn characters. The shape of each block indicates how it can be connected to other blocks and is marked with a distinctive dot pattern to be recognized by the program running on the mobile device. A mobile device scans and executes the code. The program has a text-to-speech feature adjustable to characters, sound effects, and music. The authors conducted a formative study with 16 participants, ranging in age from 11 to 65, students, and teachers. In general, all the participants were able to produce their own stories with the toolkit. During the evaluation, the authors identified some issues in the interaction between the users and StoryBlocks. VI students were sometimes not able to identify individual blocks based on the tactile alone. The participant groups could not fully understand the more complex programming concepts, such as conditional branching, and while constructing the program, they did not follow the correct syntax. Overall, the authors believe the combination of tangible programming blocks and audio output can be a compelling way to teach basic computational concepts.

Kane et al. [37] describe an introductory environment implemented in a workshop where students from different areas were working towards a common goal, determine the fictional impact of Comet ISON. Students in the computer science track had the task to monitor comet sightings using the Twitter social media and write programs to produce visualizations of the predicted impact zone. Students worked with an Apple MacBook Pro to make use of the VoiceOver screen-reader, it is terminal, and the irb interpreter prompt to code in Ruby and use the Twitter API. All the workshop activities were in a step-by-step tutorial, and when the students wanted to visualize the aggregated data, they had two ways. The students could send the data to a web service that would plot the coordinates on a USA map and explore using VoiceOver, or generate a credit-card-sized 3D tactile graphic based on the map where higher bars indicated more geotagged tweets in that area. The authors found the workshop successful because even though not all students finished the activities, they spent a week developing their programming skills. They also found issues with novice programmers, as they could not keep track of the entire scope of the program and had trouble with screen readers.

Ludi et al. [12] performed pre-college workshops targeting VI students between grades 7 and 12 to engage them in robotics programming and address the existing gap in introducing students to a future in the computing area. The goal was to present computer hardware concepts, general computing skills, and problem-solving activities in an accessible environment. The requirements for the workshops to be successful are for the activities to be accessible regardless of the level of visual impairment. The software tools have to be compatible with screen readers and magnifiers, the tutorials to be available in digital and braille form, and age-appropriate challenging fun activities. The students performed different planned activities, starting with building the robot kit and then programming it, writing code with the help of screen readers and magnifying tools, and careful placement of comments in the code. While

observing the workshops, researchers noticed the students were engaged in the proposed activities to interact with the robot and collaborated to achieve their goals. Most students also demonstrated an interest in pursuing computer programming in school courses.

Ludi et al. [38] also proposed JBrick as a part of a pre-college workshop for the VI since the LEGO software to control and program LEGO Mindstorms is not accessible because it is not screen reader compatible. The user will be able to hear the source code and errors from the compiler, and the system will recognize the NXT brick, compile the code, download it to the brick while indicating the process status with audio. The interface was designed to merely having a streamlined menu with fewer and larger icons and adjustable to accommodate needs of the user, such as modifying font size and color, keywords, and background. The keyboard is the only input device to navigate the platform, and the output is the robot behaviors and sound.

A group of informatics teachers also decided to introduce robotics and computational thinking to VI pupils [10]. They planned separate lessons for every secondary school grade in a 4-week activity where they had one lesson per week. To the younger classes, they introduced Bee-Bot, a programmable toy in the shape of a bee with seven color-coded buttons at the top embossed with different shape symbols. The students had a gridded map to perform the spatial programming tasks proposed by the teachers and move the bee-bot from a starting point to an endpoint avoiding obstacles. For the older classes, the activities prepared involved a LEGO WeDo kit. The kit needed assembly and programming using a text-based language. This activity proved to be difficult for VI students, that had problems assembling the bricks. Children with color vision disorders used different color schemes on their monitors – that hindered their ability to identify the bricks.

There is progress with a block-based tangible programming tool with a robot for spatial output. Card-Bot 2.0 [39] is a low-cost solution that uses geometric cards with QR Codes to program the robot. The user has to layout the cards building the program, and then use a mobile phone to read the codes in the correct order. P-Cube [40] is another solution that makes use of a mat with RFID readers where the children can place the tangible blocks to construct the desired sequence. The blocks have RFID tags to be identifiable and have embossed icons representing their action. The mat connects to a computer sending the instructions to the robot, although users do not have to interact with it. Younger children can program the robot by drawing lines for it to follow.

3.3 Discussion

The described works regarding programming languages and toolkits for children to learn programming present useful strategies.

The following table (3.1) displays a summary of the presented papers and articles in our related work.

We highlight the relevant features of programming environments for them to be accessible to young and VI children. The programming environment can be entirely virtual, fully physical, or mixed. The language can be block-based, written, or an array of other possibilities; written characters or icons can identify the actions in block-based languages. The input can be tangible, by audio or with peripheral instruments (e.g., mouse and keyboard). The output can be entirely virtual, sound, or a tangible robot.

Analyzing the table (3.1), we can see three works with the characteristics we acknowledge compelling to teach VI children basic computational concepts. In the case of Cubetto [31] and KIBO [32], the tangible blocks are not non-visually distinguishable, creating an accessibility barrier. Although P-Cube [40] meets all the criteria, the work has not been seriously tested.

Block-based languages are very popular as there are findings relating to the interest and engagement of children while using these to code. Block-based languages have advantages for users, such as avoiding syntax errors, interacting with a simple interface, and rapid code creation progress. When these languages have icons instead of text, illiterate children have the opportunity to start developing their computational thinking.

The existing tangible toolkits are attractive to young children since they allow them to tinker with blocks exploring the different outcomes, and discovering new solutions. When tangible toolkits are block-based, they share the advantages of both strategies. If the system provides feedback to the user, it can help the children understand what they are coding, debug errors, and distinguish similar concepts and blocks.

Hands-on experience is also essential for blind learners and young children to facilitate what they are learning and build abstract thinking. It is one reason a 3D or tangible output can help children understand the meaning of the computer program they have built. An example of an output that has proven to be very popular with children of all ages is a programmable robot. Robots can move in space, produce audio and light, and allow the children to follow their behavior.

There are different accessible tools for VI to access the content on a computer, such as VoiceOver. However, these technologies are not always compatible with the program or the computer.

Even though there is little research with an entirely tangible system with blocks and a robot to execute spatial tasks, we acknowledge that it is a compelling way to teach VI children basic computational concepts.

Table 3.1: Summary of presented works and their features. In green, the relevant accessibility features to young and VI children.

		Environment (**V**irtual, **P**hysical, **M**ixed)	**Language** (**B**locks, Written, **O**ther)	**Blocks** (**W**ritten, **I**cons)	**Input** (**T**angible, **A**udio, **P**eripheral)	**Output** (**V**irtual, Robot, **S**ound, **3D**)	**Spatial Activities**
Programming for Children	Scratch	V	B	W	P	V	NO
	ScratchJR	V	B	I	P	V	NO
	Strawbies	M	B	I	T	V	NO
	TurTan	V	O	-	T	V	NO
	E-Block	M	B	W,I	T	V	NO
	T-Maze	M	B	W,I	T	V	NO
	Cubetto	P	B	I	T	R	YES
	KIBO	P	B	W,I	T	R	YES
	TanProStory	M	B	W,I	T	V	NO
Non-visual Programming	Blocks4All	M	B	I	P	R	YES
	Bonk	V	W	-	P	S	NO
	PB	V	B	W	A	S	NO
	APL	V	O	-	A	S	NO
	Torino	P	O	-	T	S	NO
	StoryBlocks	P	B	I	T	S	NO
	Kane et al.	M	W	-	P	3D	NO
	Ludi et al.	M	W	-	P	R	YES
	Ludi et al.	M	W	-	P	R	YES
	Kabátová et al.	P	O	-	T	R	YES
	P-Cube	P	B	I	T	R	YES
	CardBot 2.0	M	B	I	T	R	YES
	Our Approach	P	B	I	T	R	YES

4

Exploring Accessible Programming

Contents

Before designing our prototype, we sought to understand the qualities and flaws present in current programming environments for children regarding accessibility. First, we conducted a focus group with IT instructors and SNEs, where we presented four environments with at least one tangible component. The educators interacted with each one and gave feedback. Considering the results from the first study, we conducted a second one with VI children. We adapted blocks with additional tactile cues and audio feedback, and the Dash robot with augmented physicality, feedback, and feed-forward mechanisms. We used a Wizard of Oz methodology, where the children tinkered with the blocks, and an investigator would use the Wonder Blockly[1] app to program the robot accordingly.

4.1 Study 1: Exploring Current Programming Tools

Schools have adopted certain approaches to promote CT with spatial activities in their curricula, aimed at different ages, with situated tasks and accessories (e.g., maps). None of them is fully accessible to VI children. In this study, we selected different examples of CT environments that had at least one tangible component (robot or blocks), leaving aside the fully-virtual setups that have shown to be inaccessible, too complicated, and restrictive for spatial tasks [14].

4.1.1 Participants

We recruited 4 SNEs and 2 IT instructors from two primary and secondary inclusive schools from our city, Agrupamento de Escolas das Olaias and Escola Básica Bairro do Armador. Such schools accommodate children with different abilities and are the reference schools for VI children in Lisbon.

4.1.2 Procedure

We started the focus group by presenting four different environments (4.1). The first one was the DOC robot by Clementoni[2], which is fully tangible. The robot has buttons on its head to walk forwards, backward, turn right, turn left, ”OK” to send the commands, and ”X” to cancel the built programming sequence. It comes with two colorful themed maps and activity cards.

The Dash robot controlled by the Wonder Blockly app was the second to be presented, a mixed environment where the input is virtual, and the output is tangible. Blockly allows the children to control the robot using a colorful block-based language. The robot moves in all directions, rotates its head, has colorful lights, an obstacle, and sound sensors.

Thirdly, we presented another mixed environment, Osmo Coding Awbie. The children use tangible blocks to control the actions of the virtual character in the game. Awbie can move forwards, backward,

[1] https://www.makewonder.com/apps/blockly/
[2] https://www.clementoni.com/en/61323-doc-interactive-talking-robot/

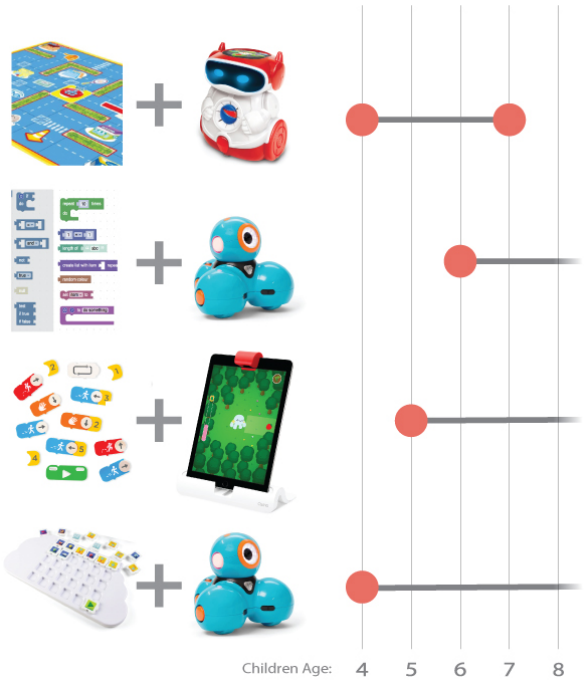


Figure 4.1: Environments used in Study 1. From top to bottom: DOC and map; Dash and Blockly App; Osmo Coding Awbie App and blocks; and Dash, Puzzlets blocks and tray. Image by Hugo Simão

left, right, pick up objects, and jump over obstacles. The blocks are colorful with icons drawn on them representing the action of the character, numbers representing the number of times to execute that action, or a loop to repeat a sequence.

Fourth, we showed a fully tangible environment with Dash and Puzzlets[3], a setup where the user composes sequences of blocks on a tray. The blocks have icons representing the actions the robot can perform and a play block. Only when the play block is in the tray is the programming sequence sent to Dash. With tangible blocks that fit in the tray, children can program the actions the robot can execute. The robot can move back and forth, turn left and right, have different speed levels, rotate the head, emit lights, and sound.

For each of the environments, we made a demonstration of the features and highlighted the most relevant. We commanded the virtual character or robot to perform one action, a sequence, and walk in a square. For the character to move in a square, we made a repeating sequence - a loop - except with Puzzlets and DOC that do not make it possible.

The educators were encouraged to interact with the environments and brainstorm to identify qualities and opportunities in each setup for in-class activities with VI children. They wrote the ideas on post-its that and grouped them in a whiteboard. At the end of the session, we asked for general opinions and debriefed the participants. The workshop session was audio recorded for posterior analysis.

[3]https://www.makewonder.com/blog/announcing-puzzlets-for-dash/

4.1.3 Analysis

We gathered all the information from the focus group, the post-its, and the recorded audio to analyze and categorize data using affinity diagrams [41]. Two researchers iterated on the relationships and categorization of the data, which was then presented, discussed, and refined with the entire team.

4.1.4 Findings

Educators observed limitations and opportunities in the CT environments presented.

4.1.4.A Benefits of Tangible User Interfaces

TUIs! (**TUIs!**) were one of the most commented during the session. Educators were very favorable of this type of system that combines a hands-on approach with digital feedback. **TUIs!** can also be beneficial in engaging children since it is more natural and familiar to play with objects. Educators also highlighted the advantage of children developing their fine motor skills. None of the educators considered screen-based technology as a solution for blind children.

Robots

Robots are tangible tools that are also very attractive to children. Educators highlighted the importance of using a robot in this environment and the inclusion of physical and socio-emotional affordances.

Different features in robots generate different interactions, and for these reasons, the educators preferred the robot Dash. The educators observed aesthetic qualities in the robot, the ergonomic spheres that compose it, the soft plastic, and the bright colors. On the contrary, they observed limitations such as the presented programming methods not being accessible to VI children and the robot not giving any audio feedback on performed actions or its position. The educators admitted the possibility of augmenting Dash to surpass these limitations.

Considering the DOC robot, educators mentioned that the physical features were not suitable for VI children to distinguish its elements, such as which side moves forward. On the other hand, educators observed qualities such as feed-forward and feedback, when the robot announces the current goal and reaches it.

The most highlighted relevant physical features of the robot were the color, shape, pleasant to touch, pretty, and personalizable with accessories. Accessories could be made of low-cost and children-friendly material as plasticine, legos, and adhesive tape.

Blocks

Blocks are an essential tool in representing abstract actions, enabling children to reduce memory and cognitive load while learning new computational concepts. The educators were attracted to the Osmo Awbie blocks due to the magnets that easily force the blocks' correct attachment. The intuitive design

in which the arrow of the direction block smoothly and precisely changes direction. Educators also observed that the blocks should be small and with easily-recognizable icons. They suggested that the icons be embossed or in 3D and with minimalist design, such as Picture Communication Symbols (PCS), primarily used with deaf children.

Regarding the blocks in Puzzlets, the educators mentioned that they do not have embossings for VI children to distinguish the represented action. On the other hand, the fact that they are small and light may help develop their fine motor skills. Educators showed enthusiasm in the use of a tray or board to place the blocks. They mentioned the children used the cubarithm[4], a braille maths teaching aid for the VI. The tray defines a personal workspace allowing the children to work with both hands simultaneously, and the organization of the blocks facilitates the debugging process. A series of possible opportunities were visited, such as incorporated sensory feedback, tactile clues or braille inscriptions, and audio feedback.

Physical Map

In addition to the robot and programming blocks, the educators highlighted the importance of a map. It allows children to explore the space and bounds of the workspace. The map could be of a real place familiar to the children, for example, their school or neighborhood. The robot could be integrated with the map to give audio feedback to announce its location and orientation. The map could also have tactile cues or braille inscriptions.

4.1.4.B Voice User Interfaces

Only one educator thought about the possibility that the children could use a Visual User Interface (VUI) system. These interfaces are especially relevant for young children as they do not require full visual and manual engagement. However, solely using VUI to learn computational thinking concepts would be highly cognitive demanding as all the information would be received in just one sensory modality. Children would need to say what they want the robot to do and then listen to the robot to announce its actions.

4.2 Study 2: Exploring Potential Approaches

The solutions presented in Study 1 were all inaccessible. However, they all showed to have qualities that could leverage in an accessible programming environment. Considering the first study results, we conducted a second one following a Wizard of Oz methodology with VI children. We adapted a solution to include a set of qualities identified in the previous study and engage VI children in a programming

[4]https://shop.rnib.org.uk/cubarithm-board.html

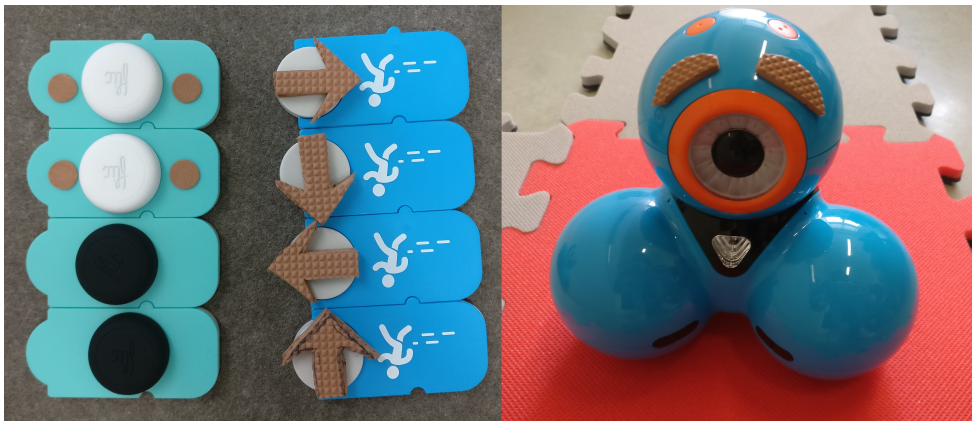


Figure 4.2: Augmented Osmo Coding Blocks and Dash Robot

workshop with spatial activities. The children manipulated adapted Osmo[5] blocks to control the Dash robot moving in a tactile map. The blocks and the robot were both augmented with additional tactile cues and audio feedback. The findings reported here emerged from our observations later validated in a follow-up focus group with their educators, where we showed the video-recordings of the workshop.

4.2.1 Participants

Seven VI children from the same school of Study 1 agreed to participate in this second workshop. The children were between five and eleven years old with different comorbidities.

The first group was composed of 3 children. A female five-year-old in first grade, with low-vision and Global Development Delay (GDD), a female six-year-old in second grade with low-vision and GDD, and a male seven-year-old with low-vision.

The second group was composed of 2 children. A blind male nine-year-old in third grade; and a male nine-year-old in third grade with low-vision, GDD, Attention Deficit Hyperactivity Disorder (ADHD), poor laterality, and compromised hand-eye coordination.

The third group was composed of 2 children. A partially blind male eleven-year-old in fourth grade, and a male ten-year-old in fourth grade with low-vision.

To fine-tune our analysis, we later conducted a focus group with the six SNEs from the previous study, a new SNE, and a IT instructor.

4.2.2 Bespoke Programming System

Informed by the results of Study 1, we set out to adopt a solution that would feature tangible and audio-rich blocks, a recognizable tactile map, and a robot with augmented physicality, feedback, and feedfor-

[5]https://www.playosmo.com/en/coding/

ward mechanisms.

The robot used was Dash due to its current usage in school settings and because the educators praised it. This robot already integrated rich tactile cues, but to make it more obvious, we added felt pads cut as eyebrows and placed them over the eye to augment its front/back asymmetries. We added audio feedback and feedforward for the actions of the robot programmatically.

Following the preferences of the educators, we used augmented Osmos Awbie blocks as the programming tools. The original play block had enough tactile information for the children to press it. On the other hand, the other blocks needed strong tactile cues. We added a Flic Smart Button[6] with audio feedback to the action blocks; action blocks instruct the robot to dance or speak. When pressing the Flic, a secondary device announced the corresponding action. To distinguish between two different action blocks, we added different round tactile stickers and used buttons of two different colors. The direction blocks were augmented with a felt pad arrow placed to rotate as the original block allowed.

For the spatial activities, we used a map. It consisted of 6 EVA foam tiles of 33 x 33 cm with two colors: red and grey. Each tile represented 1 unit, and colors were interleaved to enable children with low vision to distinguish them. The user can perceive the union of the tiles by touch and count how many units composed the map.

We used a Wizard-of-Oz methodology, where the children tinkered with the blocks, and a researcher was translating the sequence made by the children to the Blockly app that controlled the robot. Children could explore the blocks freely, and when they were ready to send the instructions to the robot, they had to attach the blocks to the play button and press it.

4.2.3 Procedure

Before starting the workshop, the seven children were divided into three groups by their educators according to age, grade, and cognitive abilities. Each group session took approximately thirty minutes and the activities had the same structure. All the sessions were video recorded.

The activities started with a discussion about robots, with one investigator asking the children questions to motivate them. She asked if the children knew what a robot was, what it does, how it worked if they ever touched any, and if they are autonomous or needed a human to command it.

Next, the investigator handed the Dash robot turned-off to the children in an unstructured exploratory activity, where the children were free to explore its physical aspects. Afterward, she taught them how to turn the robot on, and they explored its mobile actions. We introduce the blocks in the same manner. The children were free to explore the physical characteristics of the blocks and the casual relationship between each block and the actions performed by the robot.

The following activities were goal-directed, were the children had to complete the challenges the

[6] https://flic.io/

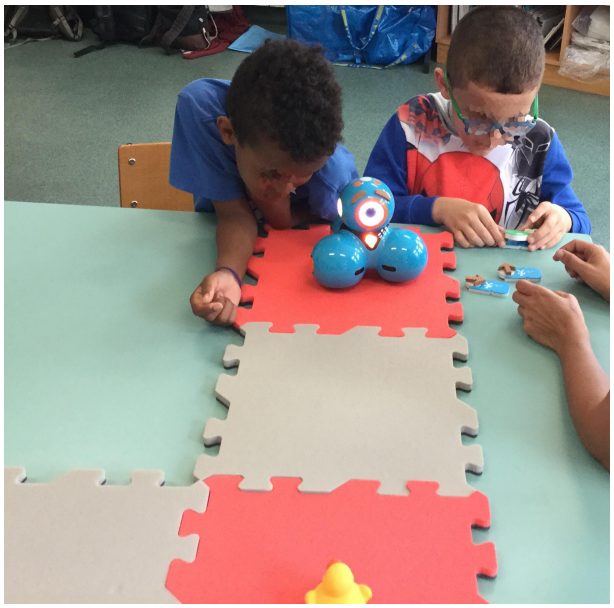


Figure 4.3: Study 2: Children engaging with the system. Photograph by Tiago Guerreiro

investigator proposed. They were spatial activities where the robot had to move in a map placed on the table. The goal of the activities was the robot meeting a rubber duck.

In the first activity, the robot is in the first foam tile and the duck in the third. The goal was to program the robot to move three times forwards to reach the duck. In the second activity, we reposition the robot in the first tile, and the duck was placed one tile to the right of the third one, meaning the robot had to make a right turn at the end of walking forwards to meet the duck. With the third group, with the older children, they performed so well in the first two activities that we added a new challenge. The rubber duck is in the second tile to the right of the third one. The children had to add a new block to walk forwards at the end of the instruction set.

At the end of the sessions, we debriefed the children about the executed programming activities, the blocks, and the robot. Finally, we asked if they were interested in playing with the robot in the future and why.

4.2.4 Analysis

During the workshop, we video-recorded and photographed each session with the children for posterior analysis. The videos were coded using a reflexive thematic analysis [42] following a hybrid coding approach. First, we designed a codebook from our theoretical background and knowledge and used it to code the videos. With each iteration, we enriched the codebook with new observed codes.

To validate or refute our findings, we conducted a follow-up focus group with the SNEs and IT instructors, where we presented short video clips from the session with the children. Even though the video

clips covered all the workshop phases and activities, we decided to cut the portions with no activity or no new observed behaviors. The educators were invited to interpret and discuss the behaviors demonstrated in the videos. Our goal was to enrich our analysis with the perspective of educators who work daily with the children.

4.2.5 Findings

Our findings indicate that, from a perspective of physical access, VI children can execute similar programming spatial activities as their sighted peers.

4.2.5.A Exploring the Robot

The children were happy and enthusiastic to play with the robot, and their bodily expressions demonstrated interest and pleasure. The younger participants were unfamiliar with some robotic concepts and approached it with an exploratory strategy. Not expecting too much from Dash, they moved it and tested the mechanics and physical limitations. Older children already knew robotic concepts and made a more conscious and concrete approach. They sought to understand the morphology of the robot, how they could use Dash, and what to expect from it by pressing on a specific part of the robot or using the blocks. Educators sustain that the unfamiliarity with robotic nomenclature is age-related. At the same time, younger participants are less likely to have been exposed to robots, and hence, less likely to talk about it and understand what it is.

We started with unstructured exploratory activities to promote physical exploration. Children immediately examined Dash by touching its head, eyes, body, eyebrows, and moving it forwards and backward on the table. Educators were amazed by the motivation and interaction the children demonstrated. They justified such enthusiasm, in part, with the agency of control in switching ON the robot by themselves and using the blocks to program its actions. Being able to anticipate actions performed by Dash gave children a sense of control and safety.

The integrated feedforward was useful for the children to follow the actions executed. Educators suggested further enrichment of this feature. If the robot could detect the map and announce its position and orientation, or when the robot accomplished or failed the task emitting a sound. However, educators also stressed that the use of audio cues should be carefully designed to avoid cognitive overload in children and jeopardize learning.

Anthropomorphization is the tendency to attribute human characteristics, emotions, or intentions to non-human entities. We often observed, especially with younger participants, children talking to the robot, kissing or petting it, bringing it closer to them, and turning the robot towards them to be face-to-face.

4.2.5.B Exploring the Blocks

The play block provided a clear anchor for the children to construct the sequence. Initially, the children tried to stack the blocks as Legos or side by side. After an explanation, the children understood that the blocks should attach above the play block. The magnets and the protrusions eased the task of attaching blocks.

The blocks had tactile integrated representation. We found that with a robust haptic representation in the blocks, the feedforward information may be irrelevant. The educators reinforced the affordances designed should include symbols that are already recognized by the children. They also insisted on integrating braille inscriptions and augmented 3D or 2D to represent the actions the blocks represent.

4.2.5.C Emotional and Embodied Experience

Overall, the children showed enthusiasm and motivation during the workshop with the robots and the blocks. Introducing new or unexpected elements increased the surprise. Their body posture, inclined to the table to be next to the robot, face directed to the location of the robot, and rapidly grabbing and moving it to start the activity, denoted interest, and satisfaction. In a few instances, children showed boredom or neutrality when others were accomplishing the task. The boredom was mitigated throughout the workshop by having short activities, so the child could collaborate or wait just a few minutes.

Children were curious about the effects of the blocks in the robot and showed great excitement when the robot collided, reached the duck, talked, and danced by laughing or applauding. Educators were amazed by their interactions with the system. The fact that they were learning to program the robot with pleasure was promising for future learning endeavors: “It was positive. Even in the initial expressions, they were already interested [..] I think they were very happy to be there” – SNE2.

The structured and goal-directed spatial activities are cognitively more demanding, so to mediate this factor, we applied a self-paced session structure and step by step instructions. This format helped children understand the activities and, consequently, facilitated a sensation of empowerment as they could perform the task and command the robot.

Children used their bodies to explore the table where the map was, move around it and the target, and indicate other aspects. Such embodied conceptualization transfers from the bodily to the abstract plane may facilitate learning. The dimensions of the map and path should be known to the child or explorable to ease the physical embodiment. Educators remarked that the distance of each robot movement should be explicit for children to develop a mental model on the blocks and the robot moves. In our setup, the robot moved in the distance of one unit of the map, so children did not have to think about distances but only in steps, translating in a less cognitive load.

4.2.5.D Collaboration

The activities were not designed to foster collaboration. However, collaboration emerged naturally within the groups. We structured the activities with sequential turn-taking so that each participant would have the same opportunities. We observed collaborative actions, such as supporting learning, instruct partners, and reinforcement. We reinforced learning through dialogue or embodied behaviors as *a priori* corrections or help in the debugging process. These corrections reflect that they have learned how to command the robot, and they were able to indicate to their partners the correct solution. One educator remarked, "the second always performed better than the first. This is inevitable. They learn from the mistakes" – SNE2.

4.2.5.E Learning CT

Programming knowledge in children has been identified with two key indicators [26, 43]. One is the ability to match a programming command with its outcome. All children, except one, reached this key indicator; they understood the function of the blocks and the output they generated in the robot. They also understood that the robot executes the sequence if they press the play block and not before.

Only older children achieved the second key indicator and one of the youngest. It relies on the ability to create a program that uses the correct commands in the correct order. The difficulty presented itself in understanding that the robot executes the complete sequence when pressing the play block. Children tended to press the play button several times. This is also coupled to the lack of understanding that they cannot change blocks during execution.

The development of abstract thinking is related to the way children associate and interpret concepts [44]. Younger children rely more on concrete thinking, physical symbols and representations [45,46]. We observed that young children were more focused on the robot rather than on its action. Older children, on the contrary, were more concerned with the action of the tangibles. For example, debugging was often observed in older children, while younger children were more prone to trial and error.

Perspective-taking is the foundation for many higher cognitive skills, especially for social skills and theory of mind, and develops systematically. Once the robot turns, the reference frame of the child and the robot are not the same anymore. Children needed to take the perspective of the robot to solve the challenge. We used this type of setup to improve spatial cognition in children, mental rotation, and navigation skills. It is important to relate these spatial concepts with their position, body, and environment. Understanding spatial concepts are crucial in VI children to navigate, orientate, and build more abstract reasoning.

4.3 Discussion

This study enabled us to identify key aspects to design an accessible programming environment. Educators were motivated about using a robot and tangible blocks, as they found it engaging for activities. They remarked that the robot should have feedback and feedforward information, and the blocks added sensory representations, such as auditory or tactile cues. To conduct programming activities related to spatial perception, it seems opportune to use a tactile-rich map to foster spatial perception and orientation. The usage of boards was also praised, having in consideration that it could restrict collaboration.

Our approach focused on programming a robot in predominantly spatial and non-visually activities for the second study. One clear difference from previous works is our focus on tangibles that are recognized to reduce cognitive demand.

Our studies highlighted that it is achievable and affordable to promote inclusive tangible robot-based programming environments through spatial activities. Together with IT instructors and SNEs, we explored the required sensory integrated representations they should feature to be accessible. The augmented physicality of robots, blocks, and maps showed promising to provide a layer of inclusion to VI children.

Collaboration emerged during the activities with children. Educators proactively saw interest in these interactions. When discussing how to promote collaboration, different elements of the setup were dissected. Small boards and trays were seen as valuable in many ways but detrimental for collaboration unless there could be different boards for each child while contributing to a shared solution. Programming by voice was seen as a reasonable means that could accommodate further collaboration. Tangible blocks were confirmed as enablers of collaboration, allowing children to share, explain, explore, and divide. The same applied to the robot that could be handled, listened, and followed.

Regarding the map for the activity, opinions were divided between being on the floor, a broader space, or a table. Adopting more space could be relevant, e.g., to promote spatial concepts but jeopardize collaboration. One of the ideas discussed to mitigate this challenge was to enlarge the programming area as well, e.g., using the entire vertical whiteboard as the programming area where children could contribute to a solution.

We also discussed collaboration within the scope of a mixed-ability classroom. Educators found interest in fostering collaboration by creating activities that attribute different roles to different actors, which could be valuable for the learning task and promote closeness and awareness.

5

BATS - a Block-based Accessible Tangible programming System

Contents

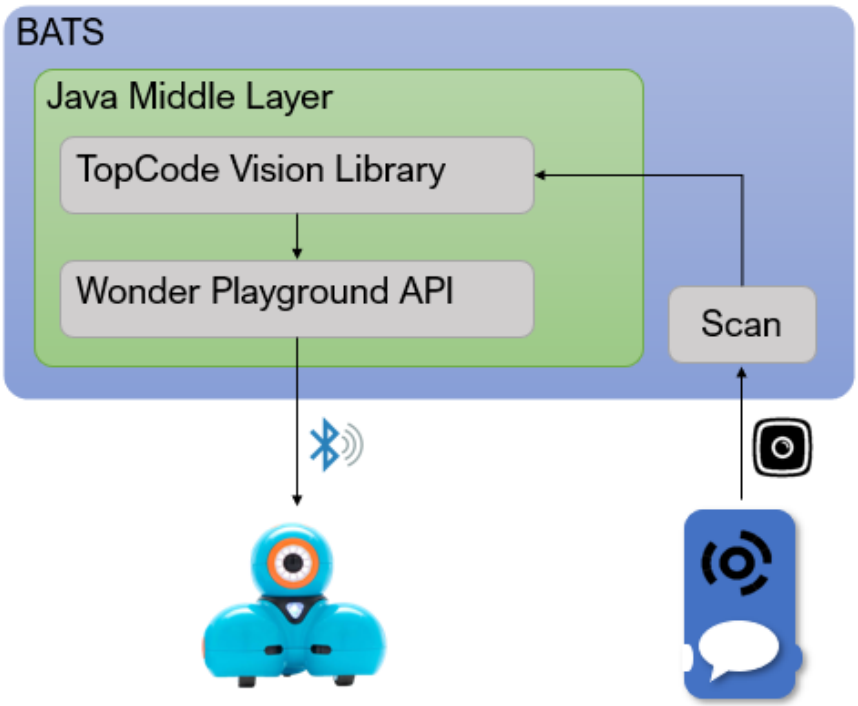


Figure 5.1: BATS Architecture

Considering the advantages of the existing approaches to teach computational concepts, our research with SNEs, and VI children, we designed BATS - a Block-based Accessible Tangible programming System. In this chapter, we present the design and implementation of the different components of BATS.

5.1 Architecture

BATS is composed of a set of tangible blocks, a Dash robot, an Android app, a tripod, and Legos.

Children manipulate the tangible blocks to construct a program with the desired actions for the robot to execute. The blocks are placed in the workspace underneath the mobile device hold by the tripod. The app can recognize the TopCode marker in the blocks and translate them before sending the instructions to Dash. In the following section, we analyze the architecture in further detail.

5.2 Programming Blocks

The set of tangible blocks (5.2) programs the sequences for the robot to execute. The blocks have a small-size to help children develop their fine motor skills, as mentioned by the SNEs in our first study.

Figure 5.2: From left to right: the play block on top, the red direction blocks and the yellow dancing blocks; on the bottom, the stop repetition block, the two and three loops blocks and the blue talking blocks

The educators were also attracted to the characteristics of Osmo Coding Awbie blocks. The bright colors, magnets, saliences, and round edges were the ones we found essential to include in our design. The magnets and saliences on the sides help children to correctly couple the blocks. The different bright colors representing each action allow them to be better distinguishable to children with low vision. In the carried out studies, SNEs also stressed the significance of embossed or 3D icons to represent the actions. We chose to emboss simple symbols on top of the blocks.

The children build the sequence from left to right, as how they learn to read and write. The constructed sequence in anchored in the play block, and the robot only executes the action when the user presses the button. The play block is different from the others since it has two pieces and works as a spring push button. The upper piece of the block covers the TopCode marker on the bottom piece. The children have to press the upper piece for the marker to be uncovered and identifiable by the system. With this mechanism, the children know that they have to perform a defined action to execute the sequence.

The direction blocks are red with a droplet-shape arrow for the user to determine the direction in which the robot will move. We choose to have just one block for all directions to promote the development of fine motor skills of the children. The dancing block is yellow with an embossed music note to instruct the robot to dance. The talking block is blue with an embossed speech-bubble for the robot to say ”Hi!”. There are green blocks that determine a looping sequence, two blocks with two or three dots representing the number of times for the sequence to be repeated, placed in the beginning, and a block with an embossed hand to place at the end of the desired sequence to repeat.

The blocks can be made with low-cost material, for example, MDF[1] or k-line[2]. For both the base of the block and icons, the previous two materials can be used. The TopCode should be printed in a paper, cut, and glued in the top center of the base. The magnets should be placed on the sides of the blocks so that the opposite poles on two consecutive blocks face each other.

5.3 Dash Robot and Spatial Output

The robot executing the output is Dash. This robot can move in all directions, sing, move its head, and beam lights, giving the user multimodal feedback to follow its behavior. Audio cues are injected in the code for Dash to announce the actions it is performing. For example, when turning left, Dash announces: ”I will walk to the left.”.

To enhance the physicality of Dash and help with the identification of the front of the robot, we added felt pads making eyebrows above the eye and a bowtie in the body.

5.4 Workspace

As shown in 5.3, the workspace represents the field of view of the camera. The Legos are in a rectangle giving the children tactile cues to help identify where they should construct the sequence of blocks. There is also a recognition area where the user can place any block for the system to announce which action it represents. For example, if the blue block is in the recognition area, the system announces ”Block for speaking.”.

5.5 Block Interpreter

The tripod holds the Android mobile device with the back camera parallel to the workspace. The mobile device app uses the back camera to scan the workspace and identify the tangible blocks. To recognize their identification number, general position, and orientation, we use the TopCode vision library, transforming them into digital markers. First, we need to determine if the TopCode associated with the play block is present. Otherwise, Dash will not execute the sequence. If the marker is present, it plays a feedback sound to inform the users. The remaining markers are then sorted from left to right, and each marker represents an action for Dash to execute. The Wonder Playground API for Android[3] translates the code to actions then sent for Dash to execute. If the TopCodes associated with the repetition

[1] https://pt.wikipedia.org/wiki/Medium$_{Density}Fiberboard$
[2] A polystyrene foam board padded with cardboard
[3] https://github.com/playi/playground

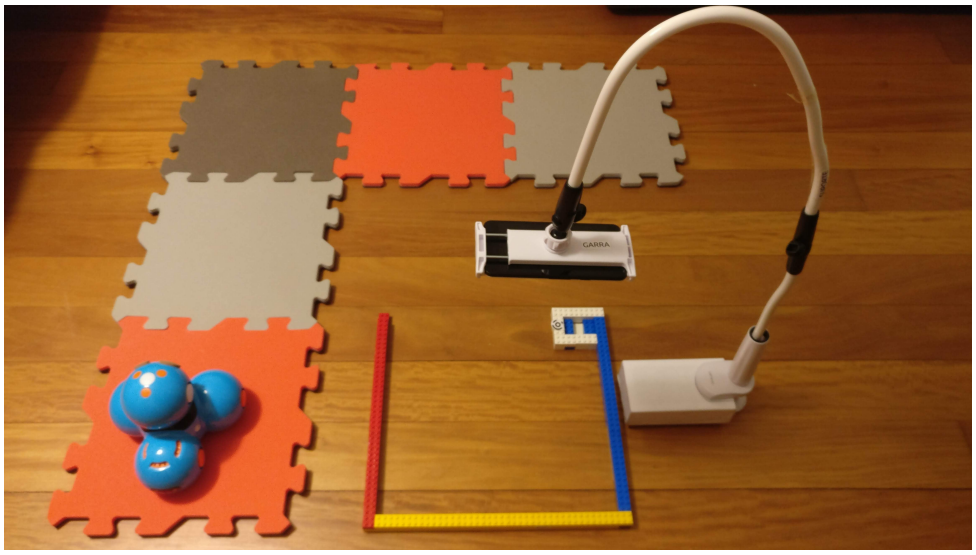


Figure 5.3: The system setup. The Dash robot on top of the map with the Lego workspace to the right and the mobile device on the grip.

blocks are present, the app identifies how many times the following sequence will execute and where the sequence starts and ends.

The user can manipulate the blocks in the workspace freely, and only when pressing the play block and with an uncovered TopCode will the system read the full sequence and send the instructions for execution.

In the recognition area, there is a TopCode marker. If the system recognizes a block marker next to it, it will announce which action the block represents.

The mobile device and the robot communicate via Bluetooth.

5.6 Map

The robot moves on top of a map. It consists of foam tiles of 33 x 33 cm with two different colors. Each tile represents 1 unit, and the two colors are interleaved to enable children with low vision to distinguish them. The user can perceive the tiles union by touch.

Objects can be placed on the map to represent obstacles or goals for Dash during the spatial activities.

6

User Study

Contents

Previous attempts to make block-based programming accessible to VI children have mostly focused on audio-based challenges. We sought to bridge this gap by developing a tangible accessible system with a robot to execute spatial tasks. We developed a toolkit for VI children to use together with their families at their homes.

The goal of this study was to explore BATS usage in a naturalistic setting and identify its benefits and limitations in the context of VI children.

6.1 Research Questions

These studies aimed to address three main research questions: (1) Does BATS support computational thinking learning? (2) Is BATS engaging and what aspects are effective in engaging children and parents? (3) Can the system be used in collaboration between two users?

6.2 Session Design

We carried out a within-subjects design study, where each participant tested the system usability by performing the tasks on the provided guidebook. For each activity, the participant had to manipulate the tangible pieces so that the Dash robot achieved the proposed goal. There is no time or a limited number of trials. The main objective is to observe how the system could be explored, and if it is engaging to children and parents.

Interviews were held after the participants tested the system via video call. The questions focused on the personal experience and engagement of the participants with the components of the system.

6.3 Participants

The participants are VI children between the ages of 7 and 14, and members of their family. The families were recruited through contacts in reference schools for VI children. Each study was held at the home of the participants.

Four families agreed to participate: (C1) an 8-year-old girl and parents; (C2) an 11-year-old with a sighted 8-year-old sibling and parents; (C3) a 7-year-old girl and one parent; (C4) a 13-year-old girl with a sighted 17-year-old sibling and one parent; and (C5) a 7-year-old boy and parents.

6.4 Toolkit

We created a remote testing toolkit to be autonomously used by the participating families. The toolkit included the BATS prototype with the tangible blocks, the Dash robot, an Android mobile device with the app installed, and a foam map for the robot to move. The included guidebook instructed the participants on how to perform the activities autonomously and prepare the setup. In some activities, the participants had to use the provided animal toys as obstacles or goals for Dash. The participants had to record the session, and for that, we included a second smartphone and a tripod.

6.5 Remote Session Procedure

We started by contacting parents and schedule the delivery and collection of our kit to be used autonomously by the family. A researcher personally delivered the box with the toolkit. The included guidebook had the intention of guiding the parents throughout the interaction with the system. The document introduced the theme and the purpose of the study to the participants. It also contained the instruction for the participants to record the interaction, to set up the experiment correctly at home and the activities. The guide includes activities embedded in storytelling to engage children in programming the robot to move, with difficulty increasing progressively. After these goal-directed activities, children and parents are encouraged to perform exploratory activities with the setup.

The first activity is the system set up and start recording. The map is composed of a three-unit straight line and two to the right. The first position of the robot should always be the one closer to the child. For the following activities, the participant has to manipulate the tangible pieces so that the robot can achieve the purposed goal. There is no time or a limited number of trials. The second activity is composed of simple tasks of one block sequence, connecting each action block to the play and observing the robot. The first activity also instructs on how to use the recognition area to hear the system announce the action the block represents. The third activity has three tasks incorporated inside a story and the answers to each. The goal of the first task is for the robot to meet and greet the giraffe. In one iteration, the participants use one direction block and one action block to talk. In the second task, the robot has to meet the giraffe again, this time moving once to the front and then turning right. The goal of the third task is for the robot to meet all its friends at a party. The friends are in the last unit of the map, the sequence should be two-direction blocks to move forwards, one to turn right, another to move forwards, and one action block to dance. The fourth activity presents an obstacle for Dash. The map has nine units laid out in a 3x3 square. The penguin is the obstacle in the center of the square, and the goal is to instruct Dash to meet his friends. In this activity, we did not give answers in the guidebook. The fifth activity makes use of the loop block. In the first task, the penguin is two blocks to the right, and Dash has to meet it. Then they both meet the friends in the opposite corner of the map.

Figure 6.1: C2 family interacting with the system at home

After children and parents interacted with the system we conducted a semi-structured interview to both children and parents to gather their opinions about the system, the interaction with the system, programming knowledge, and collaboration.

6.6 Data Analysis

For the collected data, we followed a deductive approach using a Thematic Analysis of the recorded videos. We were able to observe if the system is engaging and what barriers it could present with the first approach. The analysis identifies repeated codes and themes between the videos and answers to the interviews.

6.7 Findings

We designed activities to guide the family to interact and learn how to command the robot. Such tasks would afford children to apply five CT competencies. Activities started with the system setup (A1), Learn the system and the blocks (A2), Learn sequences (A3), using the system alone (A4), learn looping blocks (A5), and an exploratory activity fabricated by the children (A6).

6.7.0.A Computational Thinking Learning

We analyzed the videos of the interaction with BATS. We searched if children could apply five CT competencies using our toolkit and focus on Debugging, Data collection, Algorithms and Procedures, Problem Decomposition, and Pattern Recognition [47]. When coding the videos, we used these competencies as codes.

CT Competencies

Debugging is when the user can identify and address the problems that inhibit progress toward task completion. On one occasion, C1 did not rotate the arrow enough for the robot to turn right, and her father explained what she had to do. Usually, parents take over and become logistics supporters testing the system, without the children interacting. Generally, it is followed by a second try to achieve the goal.

Data Collection is when the child gathers relevant information to solve a problem, such as counting how many cells, asking about the position of the robot, or observing it. The children ask their partners for help about the position of the robot or to finish their line of thought. The parents, mostly in the role of collaborators, help them think, explain, and confirm what is happening. For instance, C2 asks his mother if the robot is facing forward to build the sequence.

Children can learn algorithms and procedures by following, identifying, using, and creating an ordered set of instructions. The children would manipulate the blocks, join them to construct the sequence while thinking aloud. They would listen to the contributions the parents give, as explanations, suggestions, and corrections. Sometimes, children might directly ask for help or use the block recognition area. For example, C1 wants the robot to make a right turn but does not turn the arrow correctly, and the robot goes forward. The parent then explains what she has to do, and she follows the correction resulting in the robot meeting its goal.

Problem decomposition employs breaking down data, processes, or problems into smaller and more manageable components to solve. It was often observed by the child thinking out loud the steps and solution towards the goal. The children usually think aloud with the help of a collaborator, in most cases, the parents. With C2, the sibling is usually the collaborator that helps to decompose the problem. The children manipulate the blocks, rotate the arrows, and re-arrange the sequence while considering the suggestions and corrections of their peers. Problem decomposition often occurs before the families start building their algorithms. For example, the mother of C5 states the goal; he thinks about the sequence, and the mother confirms that he is right.

Children use pattern recognition by observing patterns, trends, and regularities in data. In the interviews, the children and the parents recognized that they could understand the blocks commanded the actions of the robot. The children knew they had to press the play button to send the instructions to Dash. Pattern recognition was mostly applied while using the loop block. The parents explained to the child what the pattern was and how it could convert in a repetition.

CT Concepts

We also observed if children could apply CT concepts during the interaction, such as building sequences and repetition.

C1 did not understand at first that she could have two blocks connected to the play and her father had to explain and exemplify. After some more examples together, the child started to manipulate the blocks alone and building her sequences. C2 and his sibling built the sequences together from the beginning. Sometimes she was the one placing the block; other times, she would give him the block to place. They discussed the actions and then built the sequence, while the parents only intervened with suggestions. The siblings were almost autonomous while playing.

C5 started building sequences with the help of his mother. They were working in collaboration, and the parents exemplified and corrected him. He was already exploring the system alone, building sequences, and sending them for Dash to execute in later activities. The mother of C1 said in the interview: *"She might not have had the notion that she was creating a sequence. However, she was aware of what blocks she needed and the direction she had to give them in order for the robot to do what she needed and wanted."*.

C2 and his sibling soon understood the repetition and took turns doing it. They both understood how to use the loop blocks, although C2 did not find an advantage of using the loop blocks instead of all blocks. The mother and sibling helped the child think and identify the blocks throughout the activities.

The C5 family had some trouble understanding the loops at first. They examined the block and reread the guidelines for several minutes. They used the blocks correctly, with the parents explaining to the child what was happening and exemplifying. In the interview, the mother of C5 said *"The repetitions were amusing because... in case that he had to go to his friends ... we had to put front, right, front, right, we had to think how to use the piece of the repetition."*.

6.7.0.B Engagement

During the interviews, most families said that the time of exploration of the toolkit was longer than recorded.

Blocks

We observed that children enjoyed the use of tangible blocks. From the first activities, whether the parents give the blocks or the children search for themselves, we often see them manipulating blocks. Children identified blocks by their color or the embossed pictograms and sometimes used the recognition zone. Some children did not recognize the symbol itself, but they memorized which shape identified which action. A blind child commented that the size of the embossing was not big enough. Two children mentioned that they did not understand how to use the loop block, and another child said he preferred to use all the blocks instead. Families agreed that braille inscriptions in the blocks could be interesting

for the children to identify the action, and even help them learn braille. Three families with low-vision children added that the coloradd [1] symbols could help them better distinguish the colors. Regarding the play block, the participants commented that it was challenging to keep it in place when pressing it.

Robot

The families found the robot attractive, for some participants, their favorite element in the toolkit. The parents mentioned the sound and the lights as relevant characteristics for the children to perceive the behavior of the robot. The mother of C4 mentioned in the interview, *"She could see the big light in the eye"*.

We could observe the children talking and touching the robot very often during their interaction. When the robot reached the objective, children often clapped or laughed enthusiastically. Children and parents also enjoyed it when the robot fell off the map or crashed into the toy animals. Anthropomorphization is to attribute human form or personality to things not human, as children often did by naming the robot or talking to it. It shows how affectionate the children became with Dash. For example, C5 said he was already missing the robot during the interview.

Most parents believe that the children could use the system autonomously after the first interaction with their help. Parents also play an essential role in engaging children with the system. They would positively empower the children, bring their attention back to the activity, or changing the task to challenge them. We could observe the father empowering the child, *"you are very good at it"- Father of C1*.

6.7.0.C Collaboration

We observed the families performing the activities together, and different types of collaboration between them.

During the interviews, both parents and children mentioned they enjoyed spending that time together, which is not always easy. In the interview, the mother of C1 said : *"... It turned out to be a moment of union at the end of the day and it's nice, we as parents like that"*.

C1 and her father executed the activity together. The father started with the role of teacher, explaining how the system works and what she had to do for the robot to achieve its goals. The child quickly learned, and they started to collaborate, constructing the sequence to reach the proposed goals. To keep the engagement of C1 with the system, the father added some extra challenges to the activities.

C2 performed the activities with his family. He and his sibling were the ones who interacted more with the system and each other. The parents made suggestions, exemplified, and offered help when the children needed it. They became competitive, but she quickly realized that she was there to help her brother. They were almost all the time collaborating, one choosing the blocks and the other placing them in the area. Sequential turn-taking was less frequent, mainly because the parents reminded the sibling

[1] http://www.coloradd.net/

that the activity focused on C2. In the videos, we can observe the sibling directing C2: *Put together those two blocks".*

C5 also performed activities with his parents. The father took on the role of teacher, reading the instructions, and the mother was closer to the action exploring the system with the child. The mother offered help frequently, contributing by giving blocks, helping him step by step, exemplifying, making suggestions, correcting, stating goals, or confirming if his sequence was correct.

Moving between leading and following in conversations gives space for children to express their thoughts and ideas. It is much easier for a child to contribute to conversations if their ideas are valued.

For two participants, the interactions with the system were completely different from the other children. The first child did not engage with the system, only with the robot. She was always focused on the lights and sound of the robot and did not care for the blocks. The other child was not guided by the activities in the guidebook and her interaction was fully exploratory with no goals. The mother was helping her during the activity and seemed to not understand the purpose. These two facts had several implications; the child was somehow lost in what to do with the blocks and the interaction was not directed towards a goal that could have been helpful to apply CT concepts. In the interview, the child accurately named all the blocks and the actions they represented. She understood that the blocks changed the behavior of the robot. The mother remarked that activities were for younger children and chose not read them. However, the activities were crucial to apply CT concepts and direct the activity toward a specific goal and the solution to it.

6.8 Discussion

To answer our first research question, we tested if BATS promotes the use of CT competencies and concepts. The children tinkered with the blocks to build sequences for the robot to execute. The parents had a crucial role in the interaction. Their roles interchanged between teacher, logistics supporter, scaffolder, and spectator. We could observe that all parents followed the guidebook to teach the children to use the system and consequently apply CT concepts. Most parents quickly understood the purpose of the activity and encouraged the children to focus on the goal, mainly with positive reinforcement.

We observed that children built their first sequences with their sighted partner, but they were almost autonomous in the last activities. Except for three children, they comprehended how to use the loop blocks correctly and explored this concept in the last activity. Even though they were not aware, we could observe the children apply five CT competencies while interacting with the system: Debugging, Data collection, Algorithms and Procedures, Problem Decomposition, and Pattern Recognition.

We found that this setup, including blocks, a map, and a robot to perform spatial activities, was engaging. The use of tangible blocks to command a robot was successfully implemented and used.

Besides the fact that the use of blocks provides an accessible way to program, it can have other benefits as developing their fine motor skills. Tangibles also prompt more collaboration than virtual elements and are also more engaging for young children. Manipulating tangibles affords the use of concrete and embodied experiences that serve to deepen abstract concepts like programming.

The use of a robot is engaging for children, as it serves as motivation to perform programming activities. Our robot gave feedforward and feedback on its performing actions, giving children the autonomy to understand the behavior of the robot. By including tactile cues in the robot to identify its front, children could know its orientation. The tactile map and toys also helped children explore the robot location, target location, and possible paths.

We tested if BATS was accessible for VI children. We found that it was adequate and that children and parents enjoyed using the toolkit. We also examined the roles of the parents in the interaction and how our setup can support collaborative activities. All of the children had at least one partner during their interaction, a parent, or a sibling. The sighted partner collaborated by giving the blocks, contributing to their train of thought, or exemplifying how to do. In the case of the two siblings, they became competitive while collaborating.

In sum, we believe that BATS is an engaging way for VI children to learn CT. We could not test it with more families, but overall, the qualitative feedback of this experience was very positive.

7

Conclusion

Contents

Everyday technology evolves and children have the opportunity to be part of this evolution. However this is not a direct approach for VI children, since they find many barriers in learning computer programming. Most programming scenarios for children include block-based programming languages with graphical output, or input.

In this chapter, we reflect on our study, our contributions and achieved goals to overtake the problem. We also present the identified limitations and possible improvements for future work in this field.

7.1 Achievements

We presented the design and implementation of a tool to help VI children learn computational thinking. We reported on two studies addressing the flaws and benefits present in current programming environments in terms of accessibility.

We designed the tool taking into consideration the previous work in the field and the results from studies. Opposing other environments in the market, our tool is block-based and fully tangible, with multimodal feedback for VI children.

Then we reported on a remote user study to test the tool with five families. The results show the system is engaging for children and they were able to apply CT competencies. However, users helped identify improvements in the tool.

7.2 Limitations

Our study included five families, which represents a small sample of VI children interacting with the system. Of the five children, only one was completely blind. A higher number of participants would result in a more significant analysis of quantitative data.

During the user study, we were able to identify complications with our prototype. For example, when the robot disconnected from the mobile device, there was no alert. If the mobile device was not parallel to the blocks, sometimes the system could not scan and identify the TopCode.

Due to the COVID-19 pandemic, we had to perform a remote user study. The families recorded themselves, which means that sometimes the field-of-view is not the best to analyze the interaction between the user and the system, or the recording does not show the entire interaction.

7.3 Future Work

The last user study participants gave feedback on the tool and what improvements to make to all the components.

Regarding the blocks, the families suggested braille inscriptions or coloradd symbols embossed to identify the represented actions. Even though the blocks already have magnets and saliences to help with the coupling, this could be made more evident or stronger.

The robot could have more tactile clues to identify its orientation. The robot could also have more audio feedback for the user to identify its position and when it reached or failed the goal.

Finally, the activities presented should guide the children in the learning evolution.

Bibliography

[1] M. Prensky, “Programming is the new literacy,” *Edutopia magazine*, 2008.

[2] J. M. Wing, “Computational Thinking,” *Commun. ACM*, vol. 49, no. 3, pp. 33–35, mar 2006. [Online]. Available: https://doi.org/10.1145/1118178.1118215

[3] K. Brennan and M. Resnick, “New frameworks for studying and assessing the development of computational thinking,” in *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada*, vol. 1, 2012, p. 25.

[4] M. U. Bers, *Coding as a playground: Programming and computational thinking in the early childhood classroom*. Routledge, 2017.

[5] D. H. Clements and D. F. Gullo, “Effects of computer programming on young children’s cognition.” *Journal of Educational Psychology*, vol. 76, no. 6, pp. 1051–1058, 1984.

[6] M. Mut-Puigserver, M. Magdalena Payeras-Capellá, J. Castellá-Roca, and L. Huguet-Rotger, “mCITYPASS: Privacy-preserving secure access to federated touristic services with mobile devices,” *Studies in Computational Intelligence*, vol. 727, pp. 135–160, 2018.

[7] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond, “The Scratch Programming Language and Environment,” *ACM Transactions on Computing Education (TOCE)*, vol. 10, p. 16, 2010.

[8] B. J. Bartlett, V. Estivill-Castro, S. Seymon, and A. Tourky, “Robots for Pre-orientation and Interaction of Toddlers and Preschoolers who are Blind,” no. May 2014, 2003. [Online]. Available: http://www98.griffith.edu.au/dspace/handle/10072/1796

[9] B. Caci, G. Chiazzese, and A. D’Amico, “Robotic and virtual World Programming labs to Stimulate Reasoning and visual-spatial Abilities,” *Procedia - Social and Behavioral Sciences*, vol. 93, pp. 1493–1497, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.sbspro.2013.10.070

[10] M. Kabátová, [U+FFFD] Jašková, P. Lecký, and V. Laššáková, “Robotic activities for visually impaired secondary school children,” *3rd International Workshop, Teaching Robotics, Teaching with Robotics*, pp. 22–31, 2012.

[11] S. Ungar, M. Blades, and C. Spencer, *The Construction of Cognitive Maps by Children with Visual Impairments*. Dordrecht: Springer Netherlands, 1996, pp. 247–273. [Online]. Available: https://doi.org/10.1007/978-0-585-33485-1{_}11

[12] S. Ludi and T. Reichlmayr, “The use of robotics to promote computing to pre-college students with visual impairments,” *ACM Transactions on Computing Education*, vol. 11, no. 3, 2011.

[13] V. Koushik, D. Guinness, and S. K. Kane, “StoryBlocks: A tangible programming game to create accessible audio stories,” *Conference on Human Factors in Computing Systems - Proceedings*, 2019.

[14] L. R. Milne and R. E. Ladner, “Blocks4All: Overcoming Accessibility Barriers to Blocks Programming for Children with Visual Impairments,” in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, ser. CHI ’18. New York, NY, USA: ACM, 2018, pp. 69:1———69:10. [Online]. Available: http://doi.acm.org/10.1145/3173574.3173643

[15] V. Koushik and S. K. Kane, “Tangibles + Programming + Audio Stories = Fun,” in *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS ’17. New York, NY, USA: ACM, 2017, pp. 341–342. [Online]. Available: http://doi.acm.org/10.1145/3132525.3134769

[16] A. Thieme, C. Morrison, N. Villar, M. Grayson, and S. Lindley, “Enabling Collaboration in Learning Computer Programing Inclusive of Children with Vision Impairments,” in *Proceedings of the 2017 Conference on Designing Interactive Systems*, ser. DIS ’17. New York, NY, USA: ACM, 2017, pp. 739–752. [Online]. Available: http://doi.acm.org/10.1145/3064663.3064689

[17] A. C. Pires, F. Rocha, A. J. De Barros Neto, H. Simão, H. Nicolau, and T. Guerreiro, “Exploring accessible programming with educators and visually impaired children,” in *Proceedings of the Interaction Design and Children Conference, IDC 2020*, 2020, pp. 148–160.

[18] M. Israel, Q. M. Wherfel, J. Pearson, S. Shehab, and T. Tapia, “Empowering K–12 Students With Disabilities to Learn Computational Thinking and Computer Programming,” *TEACHING Exceptional Children*, vol. 48, no. 1, pp. 45–53, 2015.

[19] CSTA, “K-12 Computer Science Framework,” p. 297, 2016. [Online]. Available: https://dl.acm.org/citation.cfm?id=3079760

[20] J. Théry-Schultz, “Les enjeux de la clarification des règles,” *Concurrences*, vol. 2018, no. 3, pp. 22–24, 2018.

[21] D. Bau, J. Gray, C. Kelleher, J. Sheldon, and F. Turbak, “Learnable Programming: Blocks and Beyond,” *Commun. ACM*, vol. 60, no. 6, pp. 72–80, 2017. [Online]. Available: http://doi.acm.org/10.1145/3015455

[22] L. Arnold, “Emobie™: A robot companion for children with anxiety,” *ACM/IEEE International Conference on Human-Robot Interaction*, vol. 2016-April, pp. 413–414, 2016.

[23] M. Bonani, R. Oliveira, F. Correia, A. Rodrigues, T. Guerreiro, and A. Paiva, “What My Eyes Can’T See, A Robot Can Show Me: Exploring the Collaboration Between Blind People and Robots,” in *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS ’18. New York, NY, USA: ACM, 2018, pp. 15–27. [Online]. Available: http://doi.acm.org/10.1145/3234695.3239330

[24] Á. Castro-González, X. Z. Tan, E. Carter, and A. Steinfeld, “Social Haptic Interaction Between Robots and Children with Disabilities,” in *Companion of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI ’18. New York, NY, USA: ACM, 2018, pp. 81–82. [Online]. Available: http://doi.acm.org/10.1145/3173386.3176984

[25] O. Metatla, S. Bardot, C. Cullen, M. Serrano, and C. Jouffrais, “Robots for Inclusive Play: Co-designing an Educational Game With Visually Impaired and sighted Children,” vol. 3, pp. 1–13, 2020.

[26] L. P. Flannery, E. R. Kazakoff, P. Bontá, B. Silverman, M. U. Bers, and M. Resnick, “Designing ScratchJr: Support for early childhood learning through computer programming,” *ACM International Conference Proceeding Series*, pp. 1–10, 2013.

[27] F. Hu, A. Zekelman, M. Horn, and F. Judd, “Strawbies: Explorations in tangible programming,” *Proceedings of IDC 2015: The 14th International Conference on Interaction Design and Children*, pp. 410–413, 2015.

[28] D. Gallardo, C. F. Julia, and S. Jordà, “TurTan: A tangible programming language for creative exploration,” in *2008 IEEE International Workshop on Horizontal Interactive Human Computer System, TABLETOP 2008*, 2008, pp. 89–92.

[29] D. Wang, Y. Zhang, T. Gu, L. He, and H. Wang, “E-Block: A Tangible Programming Tool for Children,” in *Adjunct Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST Adjunct Proceedings ’12. New York, NY, USA: ACM, 2012, pp. 71–72. [Online]. Available: http://doi.acm.org/10.1145/2380296.2380327

[30] D. Wang, C. Zhang, and H. Wang, "T-Maze: A Tangible Programming Tool for Children," in *Proceedings of the 10th International Conference on Interaction Design and Children*, ser. IDC '11. New York, NY, USA: ACM, 2011, pp. 127–135. [Online]. Available: http://doi.acm.org/10.1145/1999030.1999045

[31] L. G. C. Anzoategui, M. I. A. R. Pereira, and M. Del Carmen Solis Jarrin, "Cubetto for preschoolers: Computer programming code to code," *2017 International Symposium on Computers in Education, SIIE 2017*, vol. 2018-Janua, pp. 1–5, 2017.

[32] A. Sullivan, M. Elkin, and M. U. Bers, "KIBO robot demo," pp. 418–421, 2015.

[33] Y. Qi, D. Wang, L. Zhang, and Y. Shi, "TanProStory: A Tangible Programming System for Children's Storytelling," in *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '15. New York, NY, USA: ACM, 2015, pp. 1001–1006. [Online]. Available: http://doi.acm.org/10.1145/2702613.2732806

[34] S. K. Kane, V. Koushik, and A. Muehlbradt, "Bonk : Accessible Programming for Accessible Audio Games Learning Environments for Programming," pp. 132–142, 2018.

[35] V. Koushik and C. Lewis, "An Accessible Blocks Language: Work in Progress," in *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS '16. New York, NY, USA: ACM, 2016, pp. 317–318. [Online]. Available: http://doi.acm.org/10.1145/2982142.2982150

[36] J. Sánchez and F. Aguayo, "Blind Learners Programming Through Audio," in *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '05. New York, NY, USA: ACM, 2005, pp. 1769–1772. [Online]. Available: http://doi.acm.org/10.1145/1056808.1057018

[37] S. K. Kane and J. P. Bigham, "Tracking @Stemxcomet: Teaching Programming to Blind Students via 3D Printing, Crisis Management, and Twitter," in *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '14. New York, NY, USA: ACM, 2014, pp. 247–252. [Online]. Available: http://doi.acm.org/10.1145/2538862.2538975

[38] S. Ludi, M. Abadi, Y. Fujiki, P. Sankaran, and S. Herzberg, "JBrick: Accessible Lego Mindstorm Programming Tool for Users Who Are Visually Impaired," in *Proceedings of the 12th International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS '10. New York, NY, USA: ACM, 2010, pp. 271–272. [Online]. Available: http://doi.acm.org/10.1145/1878803.1878866

[39] A. A. R. S. S. Burlamaqui, "CardBot - Assistive Technology for Visually Impaired in Educational Robotics : Experiments and Results."

[40] S. Kakehashi, T. Motoyoshi, K. Koyanagi, T. Ohshima, and H. Kawakami, "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.

[41] A. Lucero, "Using affinity diagrams to evaluate interactive prototypes," in *IFIP Conference on Human-Computer Interaction*. Springer, 2015, pp. 231–248.

[42] V. Braun and V. Clarke, "Reflecting on reflexive thematic analysis," *Qualitative Research in Sport, Exercise and Health*, vol. 11, no. 4, pp. 589–597, 2019. [Online]. Available: https://doi.org/10.1080/2159676X.2019.1628806

[43] A. Strawhacker and M. U. Bers, "What they learn when they learn coding: investigating cognitive domains and computer programming knowledge in young children," *Educational Technology Research and Development*, vol. 67, no. 3, pp. 541–575, Jun 2019. [Online]. Available: https://doi.org/10.1007/s11423-018-9622-x

[44] M. Gattis, "Space as a basis for abstract thought," *Spatial Schemas and Abstract Thought*, 2001.

[45] D. H. Feldman, "Piaget's stages: the unfinished symphony of cognitive development," *New Ideas in Psychology*, vol. 22, no. 3, pp. 175 – 231, 2004, stage Theory. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0732118X04000327

[46] J. Piaget, *The origins of intelligence in the Child*. London: Routledge and Kegan Paul, 1953.

[47] V. J. Shute, C. Sun, and J. Asbell-Clarke, "Demystifying computational thinking," *Educational Research Review*, vol. 22, pp. 142–158, 2017. [Online]. Available: https://doi.org/10.1016/j.edurev.2017.09.003

A

Appendix

The guidebook sent with the toolkit for the families to follow the activities. It instructed the participants on how to perform the activities autonomously and prepare the setup.

Sessão Remota com as Famílias

Antes de mais, gostaria de agradecer por aceitarem participar neste estudo. O principal objetivo deste trabalho **não é avaliar** o seu desempenho ou o desempenho do/a seu/sua filho/a, mas verificar se o sistema promove o ensino de pensamento computacional de forma inclusiva, nomeadamente a crianças normo-visuais e crianças com deficiências visuais.

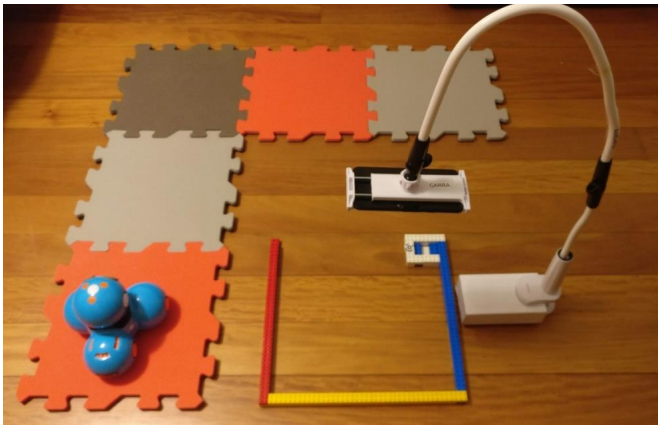
O sistema é composto por peças físicas, um telemóvel Android, legos, um mapa de esponja e um robô - *Dash*. Todas as atividades devem ser realizadas em conjunto com o/a seu/sua filho/a.

Atividade 1 - Montagem

Material:

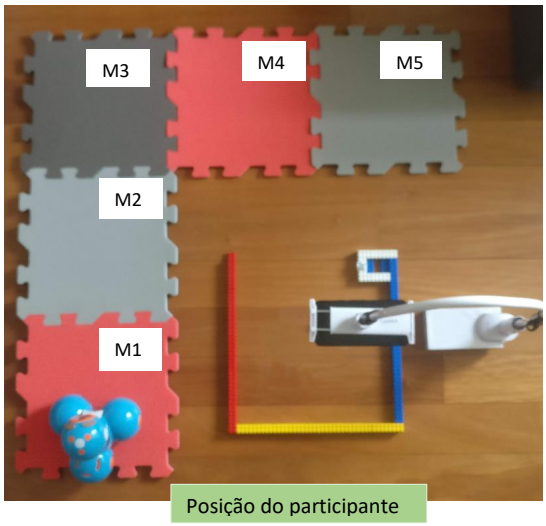
- 1 Telemóvel Nexus Branco
- 1 Suporte com garra
- 3 conjuntos de peças de legos
- 1 Robô *Dash*
- Peças tangíveis
- 9 peças de mapa de esponja
- 1 elefante de brincar
- 1 girafa de brincar
- 1 panda de brincar
- 1 pinguim de esponja
- 1 Telemóvel Nexus Preto para gravação
- 1 Tripé para gravação

Vamos começar por montar o nosso sistema. É recomendável realizar a montagem no chão onde existe mais liberdade de movimento. Todo o material tem uma etiqueta de identificação.



Em caso de dúvidas ou problemas, podem contactar em qualquer altura:
Filipa Rocha - 917007207

Para as primeiras atividades, montamos o mapa com 5 peças como está na imagem. O mapa é de esponja e ajuda-nos a criar as atividades e seguir o movimento do robô *Dash*.

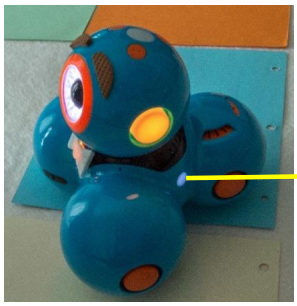


O robô fica na primeira peça do mapa (M1), que vamos considerar ser a casa dele. O robô tem movimento, som, luz e para ajudar na identificação da orientação do robô, adicionámos umas sobrancelhas na cabeça e um laço na parte da frente. O corpo, onde está o laço, dita a direção do robô.

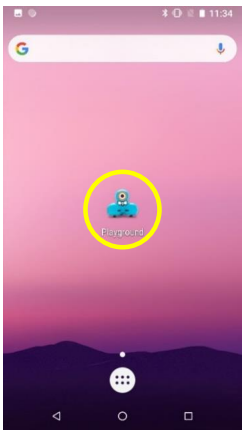
Ao lado do mapa, vamos montar a nossa área de trabalho. Temos 3 filas de peças de lego, a vermelha fica à esquerda, a azul à direita e a amarela em baixo juntando as outras duas.

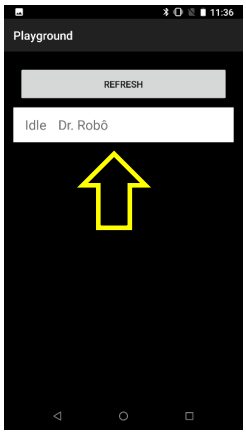
Colocamos agora o suporte, com uma garra para o telemóvel. A base deve ficar junto aos legos azuis e centrada, com a garra diretamente por cima da área.

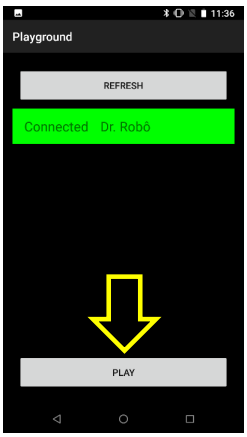
Antes de colocar o telemóvel na garra é preciso ligar o robô *Dash*.



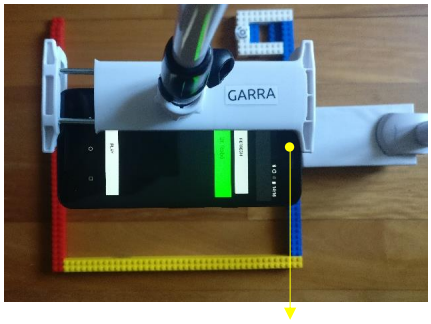
Vamos também iniciar a aplicação "Playground" que está no ecrã principal do telemóvel(branco), ligar o robô *Dash* e escolher a opção "Dr. Robô" na aplicação seguida de "PLAY".

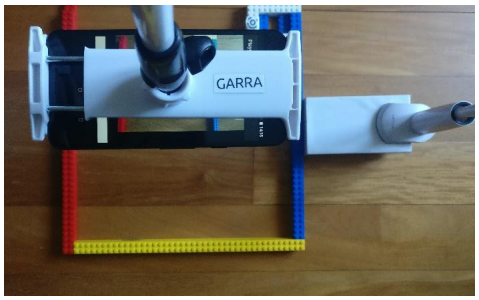






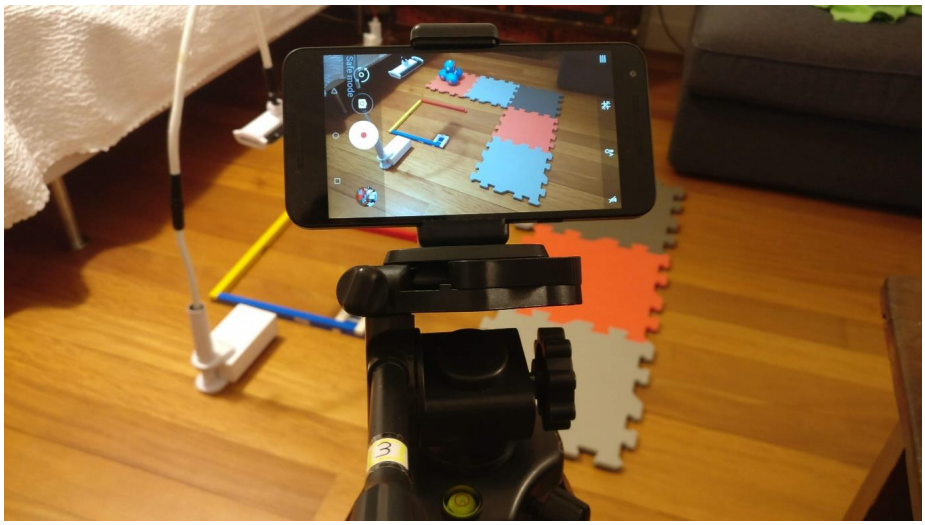
Agora podemos montar o telemóvel (branco) na garra, com o ecrã virado para cima e a câmara para baixo.





Câmara frontal

Já estamos quase prontos! Só falta montar o tripé com o telemóvel para gravação (preto). Vamos posicionar o tripé de maneira a conseguir gravar toda a área. No tripé, começamos por puxar para baixo a roldana central (identificada com o número **1**) que vai abrir os pés e rodamos para trancar a posição (setas indicativas na etiqueta). De seguida, abrimos as **3** abas identificadas com o número **2**, descemos as 3 pernas completamente e voltamos a fechar as abas. Antes de colocar o telemóvel no suporte de gravação, vamos ligá-lo e abrir a câmara que se enccontra logo no ecrã inicial. Colocamos o telemóvel no suporte de gravação e ajustar a alavanca identificada com o número **3** até conseguirmos um campo de visão parecido com o seguinte:



Na câmara do telemóvel passamos para o modo de vídeo e podemos começar a gravar! Estamos prontos para começar as atividades! Não se esqueçam de ficar de frente para a câmara.

Atividade 2 – O *Dash* vai dançar, cantar e andar

No conjunto de peças, existe a peça play e é a esta peça que vamos juntar às peças de ação.

1. Na área de trabalho, a área delimitada pelas peças de lego, coloque a peça amarela com o símbolo ♫.

2. Coloque a peça na zona de legos brancos e azuis. O sistema anuncia que é a peça para dançar! Retire a peça da zona.

3. Encaixe a peça play do lado esquerdo da peça amarela.

4. Pressione a peça play na parte mais alta da peça, onde se encontra o símbolo ▶ até ouvir um sinal sonoro.

5. Na área de trabalho coloque a peça azul com o símbolo 💬 .

6. Coloque a peça na zona de legos brancos e azuis. O sistema anuncia que é a peça para falar! Retire a peça da zona.

7. Afaste a peça play da peça amarela e coloque a peça amarela fora da área.

8. Encaixe a peça play do lado esquerdo da peça azul.

9. Pressione a peça play!

10. Na área de trabalho coloque a peça vermelha com a seta.

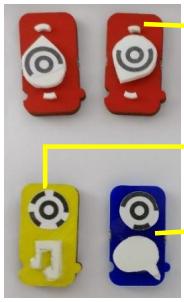
11. Coloque a peça na zona de legos brancos e azuis. O sistema anuncia que é a peça para andar! Retire a peça da zona.

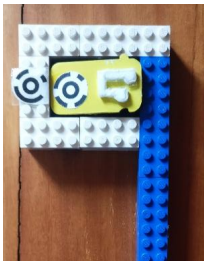
12. Afaste a peça play da peça azul e coloque a peça azul fora da área.

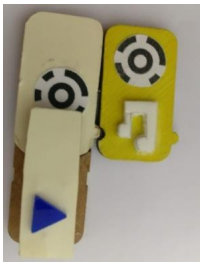
13. Encaixe a peça play do lado esquerdo da peça vermelha.

14. Rode a seta para a direção em que quer que o robô se movimente, em frente.

15. Pressione a peça play!

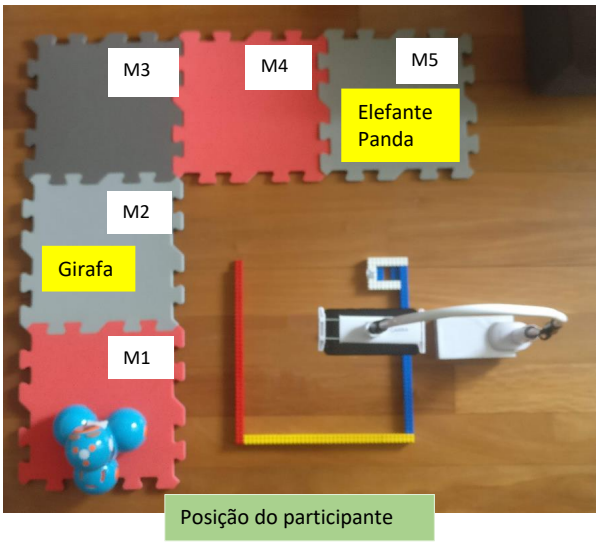






Atividade 3 – *Dash* e os amigos

Agora o objetivo é construir sequências com 2 ou 3 peças. Voltamos a colocar o robô em casa (M1), a girafa na segunda peça do mapa (M2) e o elefante na última peça (M5).



(História) Era uma vez, um robô chamado *Dash* e os seus amigos girafa, elefante e panda. Viviam todos na aldeia dos quadrados. O panda era um pouco preguiçoso e estava sempre em cima do elefante - mas o elefante não se importava! O *Dash* gostava muito de brincar com os seus amigos.

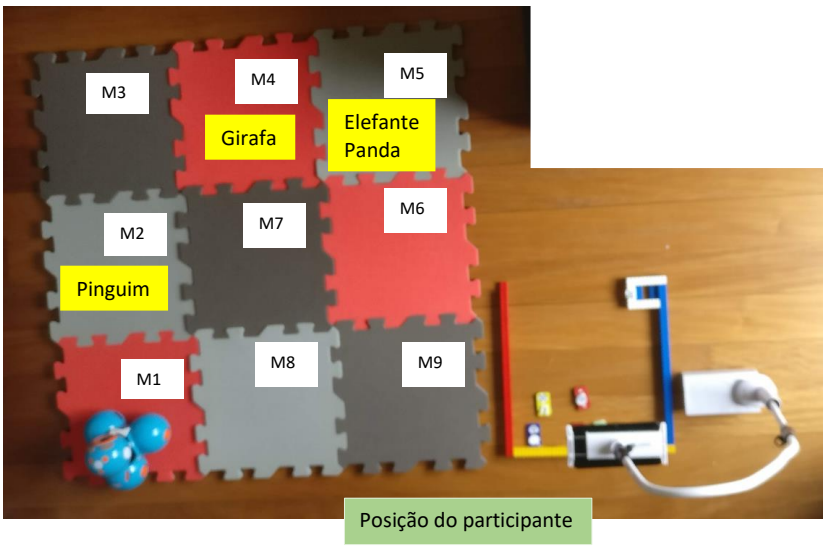
Hoje o *Dash* está em casa, mas gostava de ir ter com a sua amiga girafa. Será que consegues ajudar o *Dash* a chegar ao pé da sua amiga e cumprimentá-la?

(Colocamos a girafa em M4) Ups! A girafa não reconheceu o *Dash* e fugiu! Será que o *Dash* consegue ir ter com ela outra vez?

(Colocar *Dash* em M1 e girafa em M4) É o fim do dia e há uma festa em casa do elefante. O *Dash* precisa de ir a casa antes de ir para a festa. Consegues ajudar o robô a chegar à festa dos amigos e dançar?

Atividade 4 – O pinguim malvado

Antes de continuar as novas atividades vamos montar mais peças no nosso mapa para construir um quadrado (M6-M9). Vamos colocar o robô em casa (M1), a girafa na quarta peça(M4), o elefante na quinta peça (M5) e o pinguim na segunda peça (M2).



No dia seguinte, o *Dash* queria ir ter com os amigos, mas estava um pinguim malvado no caminho, que não o deixava passar! Era um pinguim invejoso e dizia "Se eu não brinco, ninguém brinca!". Por isso, punha-se no caminho do robô para o impedir de ir ter com os amigos girafa, elefante e panda.

Consegues fazer com que o robô vá ter com os seus amigos elefante e panda sem passar pelo pinguim?

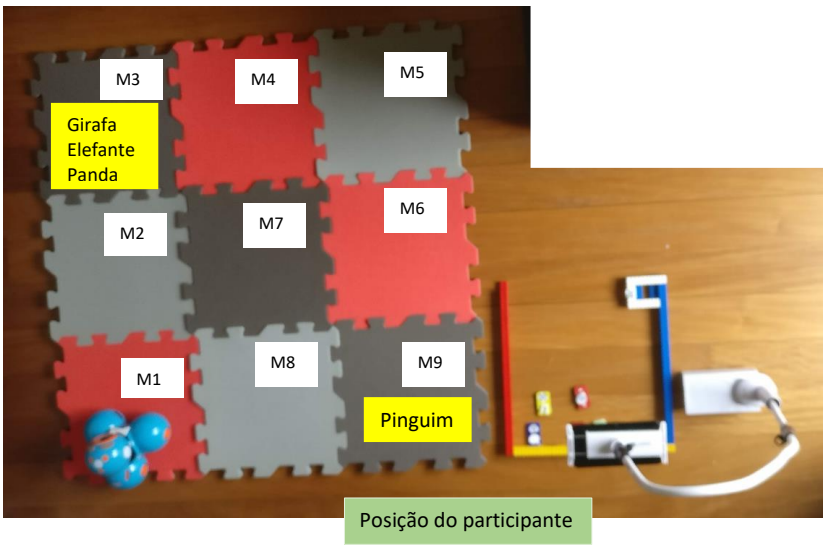
O *Dash* quer ir chamar a amiga girafa para se juntar à brincadeira. Consegues ajudar o *Dash* a ir até à amiga e dançar?

Depois de muita brincadeira o *Dash* quer voltar para casa, vamos ajudá-lo a evitar o pinguim?

(Propostas de solução à parte)

Atividade 5 – O pinguim malvado outra vez, mas…

Vamos colocar o robô em casa (M1), os animais na terceira peça (M3) e o pinguim na última peça (M9). Existe um conjunto de peças verdes que permitem a repetição de sequências criadas. A parte superior da primeira peça determina quantas vezes a sequência vai ser repetida (representado pela quantidade de pontos em relevo) e a peça com a mão o fim da sequência.



Um dia, o robô reparou que o pinguim não estava no seu caminho e pensou que seria uma boa oportunidade para falar com ele. Então foi ter com ele:

O pinguim adorou e ficaram amigos! O que ele queria era um pouco de atenção!

Vamos levar o pinguim e o robô para a festa com os outros animais? Conseguimos usar as peças verdes que repetem instruções?

(Propostas de solução à parte)

Atividade 6 – Dar asas à criatividade!

PARABÉNS! Terminou todas as atividades planeadas. Agora que já domina todos os conceitos básicos dê asas à sua criatividade, pode criar novas aventuras para o *Dash* e os amigos! O céu é o limite do que poderá fazer.

Em caso de dúvidas ou problemas, podem contactar em qualquer altura:
Filipa Rocha - 917007207