Designing Programmable Toy's Interfaces for Small Children

Projetando Interfaces de Brinquedos Programáveis para Crianças Pequenas

Cesar Pereira Viana, Applied Computing Master Program, Itajaí Valley University
cesarviana@edu.univali.br

André Raabe, Applied Computing Master Program, Itajaí Valley University
raabe@univali.br

Cassiano Pereira Viana, Applied Computing Master Program, Itajaí Valley University
cassiano.viana@edu.univali.br

Abstract
Programmable toys seek to help children in their first contacts with algorithms and facilitate the development of Computational Thinking. This article examines programmable toy interfaces and describes the design process for three programming interfaces for RoPE (Educational Programmable Robot). The first interface targets children aged from 3 to 6 years old and have colorful physical buttons. The second interface is a smartphone app that allows programming by fitting blocks and keeps the sync with the physical buttons of the toy. Finally, the third is a tangible block interface, designed to allow natural interaction and collaboration. A quantitative evaluation of the application showed that the use of physical buttons is more efficient than dragging blocks on a smartphone screen. We evaluated qualitatively the tangible interface through video and content analysis. The analysis shows that children often read the tangible blocks, which encouraged the debugging of an algorithm and collaboration between students.

Keywords: Programmable Toys, Computational Thinking, Programming Interfaces

Resumo
Brinquedos programáveis buscam auxiliar crianças nos primeiros contatos com algoritmos e facilitar o desenvolvimento do Pensamento Computacional. Este artigo analisa interfaces de brinquedos programáveis e descreve o processo de design de três interfaces de programação para o RoPE (Robô Programável Educacional). A primeira interface tem botões físicos coloridos, e foi projetada para crianças de 3 a 6 anos. A segunda interface é um aplicativo que permite programar encaixando blocos e mantém a sincronia com os botões físicos do brinquedo. Por fim, a terceira é uma interface de blocos tangível projetada para permitir interação natural e colaboração. A avaliação do aplicativo foi quantitativa, e indicou que o uso dos botões físicos é mais eficiente do que arrastar blocos em uma tela de smartphone. A interface tangível foi avaliada qualitativamente por meio de vídeo e análise de conteúdo. A análise indicou que as crianças frequentemente leram os blocos tangíveis, o que favoreceu a depuração de um algoritmo e a colaboração entre os alunos.

Palavras-chave: *Brinquedos programáveis, Pensamento Computacional, Interfaces de Programação*

1. Introduction

The origin of programmable toys can be attributed to the LOGO language. This language allows children to program a physical or virtual turtle to move and draw on paper or on the screen. Initially, its commands are typed on a keyboard. This style of interaction was hard for young children, as many are illiterate and typing characters is prone to syntax errors. This type of error can be boring for children and dry the attention from the construction of the algorithm to the interface interaction.

In 2006 Wing proposed the concept of Computational Thinking (CT). She says CT is fundamental to the analytical ability of all children, such as writing, reading and arithmetic. Since then and more since 2013, the CT topic has been gaining attention from the scientific community (TANG; CHOU; TSAI, 2020) and government policies (HSU; IRIE; CHING, 2019), concerned with preparing people to live and act in the digital world.

CT is not just about programming (BBC LEARNING, 2015; BRACKMANN, 2017), but programming allows us to carry out CT processes. When writing algorithms, we need to select relevant information, decompose a sequence of steps and identify repeated patterns. There is an increasing number of materials and tools that allow children's programming, such as tangible interfaces (REIS; GONÇALVES, 2016) unplugged computer activities, smartphone apps and programmable toys (PT). Our analysis focuses on PTs.

PTs can help children to take their first steps into the world of programming. With the aid of pedagogical mats and other accessories, the activities with PT can be fun and engage children in contextualized learning situations to develop problem-solving strategies and computational thinking. There are many existing toys on the market but most of them are designed for children over 5 years old. Furthermore, few studies focus on the programming interfaces of these toys.

Due to the lack of toys for the public under 5 years available in Brazil, our research group decided to design a new programmable toy focusing on children from 3 to 6 years. We started by investigating how to create a simple programming interface for children to interact. After five years of research and development, we created a programmable toy called RoPE (an acronym for Educational Programmable Robot, in Portuguese).

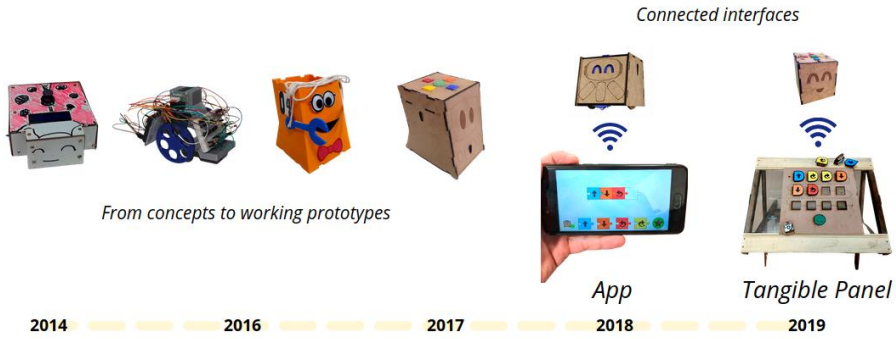

Figure 1: Concepts, prototypes and programming interfaces.

Figure 1 shows some artefacts produced between this period, including concepts, prototypes and connected interfaces. This paper emphasizes RoPE's design process and critical definitions

for design programming interfaces dedicated to small children. The second section shows results from an industrial mapping study about programmable toys interfaces. The next three sections present interfaces created for RoPE, discussing the design process, lessons and the rationale behind each one.

2. Programmable Toys Interfaces

The origin of programmable toys can be attributed to the LOGO language. This language allows children to program a physical or a virtual turtle to move and draw on paper or screen. Initially it's commands are typed using a keyboard, which is difficult for small children. Many are illiterate, and typing characters is prone to syntax errors. This type of error can be boring to children and dry the attention from algorithm construction to interface interaction.

Today we see many other types of programmable toy interfaces. The categorization of Hamilton et al. (2020) organizes 30 toys in 6 categories: (i) Board Games and Books: toys without electronic components; (ii) Non-Robotic Electronics: electronic, but without a physical robot; (iii) Screen-Based Robots: physical robots programmed by graphic interfaces; (iv) Button-Operated Robots: robots programmed by buttons on the toy body; (v) Robots with Tangible Interface: robots programmed by physical blocks; (vi) Blended: toys having characteristics of multiple categories.

Yu and Roque (2019) also analyze 30 toys and organize them in physical, virtual and hybrid kits. The physical kits have all components tangible, typically comprising a physical robot and a set of coding blocks. Virtual kits are mobile or desktop applications and don't have a physical robot. Hybrid can be of two types: tangibles programming virtual characters, like Osmo, and screen-based applications programming physical robots, like Dash and Dot.

Although these classifications cover a wide range of interfaces, little research focuses on the design of the programming interfaces of physical programmable toys. In other words, the existent research also covers apps and robots not focused on programming. To fill this gap and find toys not covered by previous research, we accomplished an Industrial Mapping study. We followed this strategy to find industrial products and include toys launched by small companies not related to academic studies or paper publications.

Although it is not a systematic review of academic publications, we followed a systematic approach to reduce research bias, selecting search sources, using a search string and applying inclusion and exclusion criteria. The four primary are Amazon and Kickstarter websites, YouTube videos, and a bookmark built by the first author. When a search field was available, the following research string filtered the results:

((coding OR programmable) AND toys)

The first inclusion criteria states that the toy must be programmable, in other words, at least allow input sequences to be executed later by the toy. The second criteria was the presence of some physical/tangible component either in the toy's interface or in the toy itself. The last criteria is the toy or it's interface should have some electronic component, excluding products like books and board games.

The application of the inclusion criteria resulted in a table with 56 toys. The table is available online at http://lite.acad.univali.br/progtoys and contains information such as manufacturer, country, year of launch and age of the target audience. Screenshots focusing the toy's interfaces were taken from videos and imported into an online drawing software and were spread out. Two researchers discussed similarities and differences between the interaction metaphors for each interface and grouped the pictures, following the Kawakita Jiro Method (SCUPIN, 1997). The method results in an affinity diagram, in which similar images are placed close to each other and distant from those that have less similarity. Our resulting diagram has two major categories: virtual and tangible interfaces. Figure 2 shows a simplified version of the diagram with fewer images for easy visualization.

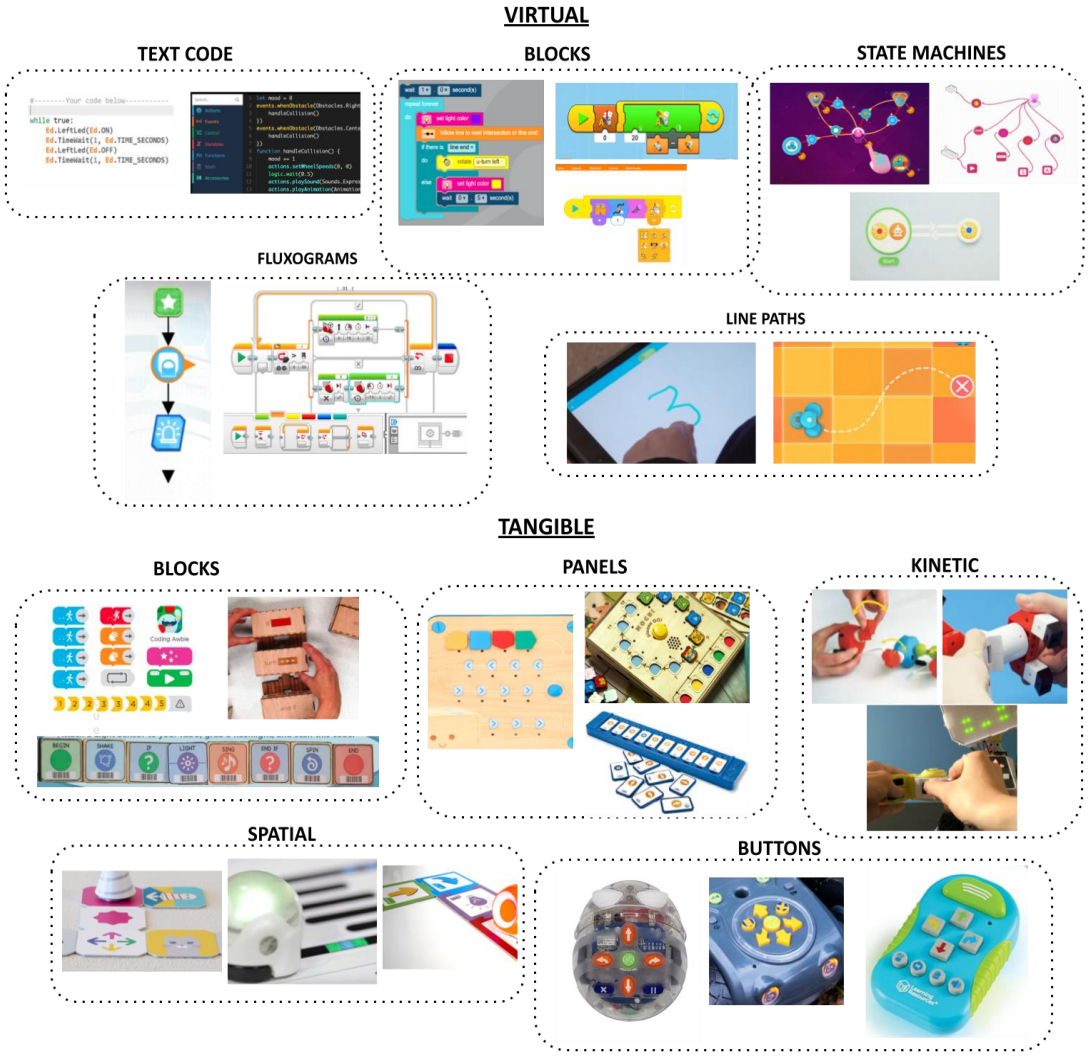

Figure 2: Programmable toys interfaces classification.

Virtual interfaces allow the user to interact with elements produced by computers. The elements appear on a display, and the user interacts using keyboard, mouse ou touch screen. Tangible interfaces, on the other hand, allow the user to program using only real-word objects. Both categories have subcategories, shown in Table 1.

Virtual	
Text code	Interfaces using typed text, like general programming languages. Robots like Thymio, Edson and Anki Cozmo have this type of interface.
Blocks	Puzzle pieces that are connected to represent sequences and other programming structures. Examples are Scratch and Scratch Jr.
State machines	Geometric shapes connected by lines, where the lines represent transitions between groups of robot actions. Example robots are Dot and Dash and Robo Wunderkind.
Line paths	The robot moves reproducing the shape defined by a line drawn on mobile or desktop application.
Fluxograms	Geometric shapes connected by lines. Each shape represents an action or decision.
Tangible	
Blocks	Like virtual blocks, but made of wood or plastic. They are interconnected and do not depend on a board or panel. Example: KIBO.
Panels	Physical blocks are connected in panel slots. Examples: Cubetto, Mojobot and Mochi.
Kinetic	Toy records moves performed by the user on parts of the toy's body. The main example is Topobo.
Spatial	The toy moves and recognizes instructions disposed over the moving space. The user can change the program while the robot moves.
Buttons	Physical buttons attached to the toy's body or in a remote control. Examples: Bee-Bot, Big Track and Go Robot Mouse.

Table 1: Categorization of Programmable Toy's Interfaces.

The type of interface most commonly used is virtual blocks, present in 53% of the toys. Tools like Scratch and ScratchJr follow this paradigm, that avoids syntax errors because only blocks with complementary shapes are conectable. The user also isn't forced to recall the syntax and commands (ROQUE, 2007). Other sources of virtual blocks popularity can be frameworks like OpenBlocks (ROQUE, 2007) and Blockly (FRASER, 2015) that simplify the implementation of block programming interfaces and enable developers to focus on creating their own programming systems rather than implementing blocks functionalities.

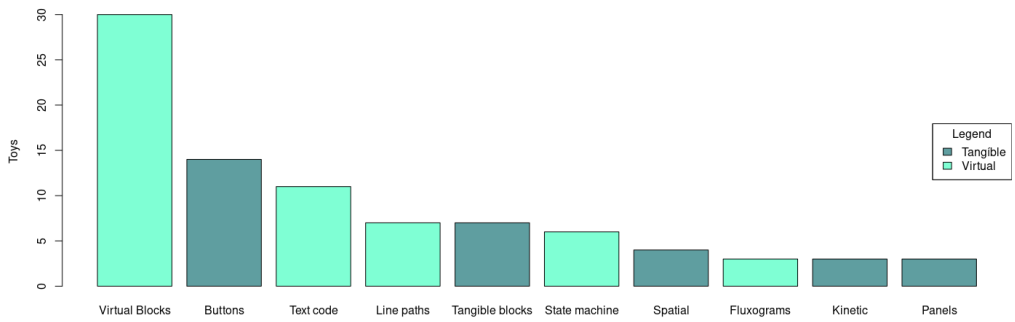

Figure 3: Toys per interface type

Another finding from the review was the multi-modality of programming interfaces: 33% of the toys have two or three interfaces and 17% have over three types of interface. Tinkerbots, for

example, has virtual blocks, physical buttons and also kinect memory. The multiple-interfaces approach has two immediate consequences. First, the children can choose the interface she feels more comfortable. Children with difficult motor skills, for example, can program using large blocks instead of small buttons.

A second consequence is the possibility of progressively unveiling computational concepts. Buttons, for example, can be sufficient for children to understand the idea of sequencing, but fluxograms and state machines can better represent parallelism. Toys like Thymio, Root and Cue offer levels of complexity in their virtual blocks interfaces, scaffolding children in their learning trajectories.

3. RoPE's Interfaces Design

3.1. Buttons

The RoPE's design process started by analyzing interfaces of PTs like Thymio, RoboDoc, and Bee-Bot (Figure 4). Despite the similarities of buttons interface and output movements, the target audience of them are different. While Thymio targets children 6+ years old (RIEDO et al., 2012), RoboDoc targets 5+ years old, and Bee-Bot starts at 3 years old.

(a) (b) (c)

Figure 4: Thymio, RoboDoc and Bee-Bot.

Based on the target audience, we choose Bee-Bot to advance the understanding of children-toy interaction. Bee-Bot programming interface has seven buttons. Four buttons are used to program movements: go forward, go backward, turn right and turn left. The other three are used to start execution, clear memory and pause the execution.

Intending to verify the interface's intuitiveness, we run a workshop for 25 children (3-6) of a kindergarten to play with Bee-Bot. The researchers presented the toy to teachers and children and observed children explore it without adult interference. Most of the children required teacher's explanation to understand the buttons functions. At the end of the section, we showed drawings of many alternatives for toy appearance such as car, ladybug, rabbit, train, robot, cat, dog, and others. The character "robot" was chosen by most of the children.

We observed the following aspects when children played with the toy: (i) children didn't understand the need to press the clear button to erase the commands recorded in the toy's memory each time they wanted to play; (ii) it was hard for them to understand the rotation movement; (iii) the button's sound and the robot's steps (way of moving) were very useful; (iv) the slight pause

between each step helped the children to count the steps; and (v) it is essential to use a sound to warn the program has ended.

(a) (b)

Figure 5: The RoPE robot and children programming it over a mat.

An interview with the teacher brought many important suggestions for the interface design for small children such as (i) to increase the size of the buttons, since kids are still developing fine motor skills; (ii) to use different colors for each direction since kids are still learning laterality, and the association with color helps the teacher to communicate directions such as "turn right with the yellow button"; and (iii) to remove the "clear" and "pause" buttons since these actions were not adequate for the mind of small kids. Figure 5a shows the resulting interface, present since 2017 in kindergarten schools of Balneário Camboriú city and available by almost 1000 children. The toy's interface proved suitable for children of 3 years old (Figure 5b) and older. The pedagogical impact of the RoPE initiative is the subject of many ongoing research projects.

3.2. Smartphone App

One limitation of toys like RoPE and Bee-Bot is the impossibility of visualizing a sequence of programmed instructions. We think the lack of visualization could make debugging less viable. Debugging is between the CT practices (BRENNAN; RESNICK, 2012), motivating the construction of a second interface allowing children to see and edit instructions stored in the toy's memory.

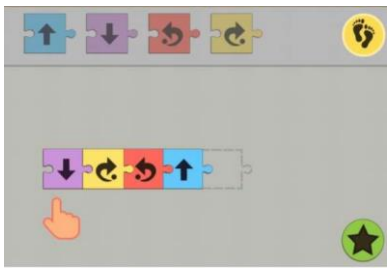

Figure 6: The app interface: blocks snapped together show the program sequence.

We developed a smartphone app in order to meet these requirements. According to Nacher et al., (2015), young children can interact with touchscreen interfaces, so we created a new programming interface based on this paradigm. We implemented a Bluetooth connection in the toy to enable the communication of programs created on the app as sequences of puzzle pieces

(Figure 6). The app interface highlights the active command, helping children to identify an undesired result. A “two foots” button turns-on the debug-mode. In this mode a hand points to the current executing command before its execution. The toy only moves when children tap the current block. This allows the child to slowly follow each command execution and to do the mapping between symbol and movement.

The toy’s buttons interface remains enabled when the app is connected, so the children can choose to input commands by app, buttons or both. The app allows inserting, moving and deleting commands, and the physical buttons allow only insertion. Each button pressure makes the inserted command to appear on the screen. This synchronization and parallel interaction turns the app and buttons on a hybrid interface. A bidirectional protocol synchronizes the information between the smartphone and toy. Each action in either interface goes to both app and robot, so they are always consistent.

Horn, Crowser and Bers (2012) advocates for the advantages of hybrid interfaces. Beyond allowing multiple forms of interaction, the hybrid interfaces can provide scaffolding to students. They can start with simple programming interfaces, but progress toward more authentic and powerful programming environments. Catlin et al. (2018) say the same about robots:

> Robots allow teachers to create environments which reflect Bruner’s Spiral Curriculum [...]. Young children start with [a robot] where all they do is put symbols in the right order, but as their experience and interest grows they can end up coding in professional programming languages. Several robots provide rich educational environments by offering different ways for students to program them (CATLIN et al., 2018).

The design of ScratchJr (FLANNERY et al., 2013) also shows the scaffolding between programming interfaces. While Scratch blocks have text and could be difficult for illiterate children, the ScratchJr blocks are icon-based. The RoPE’s app aims to scaffold the understanding of what is an algorithm, showing the sequence of commands otherwise hidden in toys memory.

3.2.1. App-Buttons Assessment

We conducted an experiment to assess how children interact with the app-buttons interface. We are interested in (a) comparing the efficiency and efficacy of the smartphone interaction versus button interaction on problem-solving tasks, and (b) observing debugging attempts.

The teacher illustrated the functioning of both interfaces to a group of 24 children from 4 to 5 years old. The children freely explored and played with the toy for around 30 minutes and then are splited into two groups (A and B). Group A used the smartphone interface and Group B used the buttons interface. Every child had to individually solve the problem over a mat created for the experiment (Figure 7). The solution consisted of programming 8 instructions with 3 direction turns. In case of errors, the teacher repositioned the toy in the last correct position.

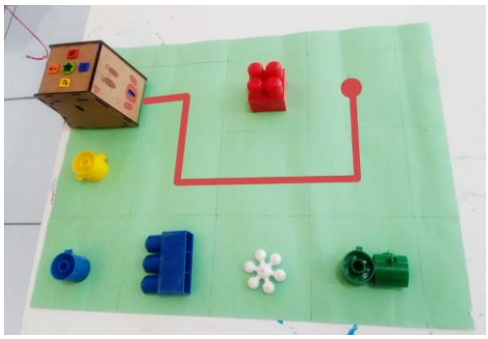

Figure 7: The problem to be solved.

The teacher assisted the children during problem solving. An observer took notes from errors and input commands, and marked the start and end instants of each activity. The teacher helped all children to accomplish the task, so all children completed it successfully. Despite that, the teacher don't touched the interface, and the children inputted all the commands.

During the free exploration phase, only two children had problems understanding how to program RoPE using the hybrid interface. Many enjoyed pressing the buttons to see the corresponding puzzle piece appear on the app screen. The exploration phase also showed the lack of feedback from the app's start button, as the children repeatedly pressed it making the toy start and stop immediately. The solution was to disable the button temporally and change its color.

The average time children took to solve the problem with the app interface was 222.5 seconds. The average resolution time with the button interface was less than a half: 103.5 seconds. Children spent more time inserting the instructions on the app because they constantly had to look at the robot position on the mat. Children also had difficulty dragging the pieces from the bottom of the screen and fitting them with the other pieces in the center of the screen. When programming using the buttons, this didn't happen. Each child made a few mistakes while solving the problem, but the interface had no influence on the average number of mistakes. Button's interface average was 3.2, and the hybrid interface was 3.4 (Table 2).

Average	Buttons	App
Time (seconds)	103.5	222.5
Mistakes (occurrences)	3.2	3.4

Table 2: Comparison between button and app interfaces.

The children solved the problem in parts, programming one movement at a time and executing it. The metaphor of fitting parts to the smartphone screen encouraged creating larger programs. Still, none of them had over three commands.

None of the children tried to change (move or delete) the pieces on the app and always constructed new programs from scratch. The children always looked at the robot's movements and did not compare it to the instructions on the screen, so the highlighting of the commands on the app was not perceived. The hybrid interface failed to help the children to debug their solutions, presented less efficiency and the same efficacy when compared to buttons interface, at least in this first experiment.

The major result was the time needed to solve the problem: the app required twice the time of the button's interface. A set of theories can explain this result.

One of them is the need for spatial mapping. Piaget and Inhelder (1948) discusses the fact children struggle to put yourself from someone else's perspective. In fact, the buttons over the toy's "head" are always pointing to the correct moving direction. In other words, the buttons and the toys are equally rotated. The app screen, on the other hand, could be rotated differently, in a way that the screen's forward arrow could be not pointing to the toy's forward.

Another cause could be the tangibility of buttons interface. The results match the existing literature about tangible interfaces for young children. Comparing tangible versus graphical interfaces, Sapoudinis, Demetriadis and Stamelos (2015) show that younger children in particular needed less time in robot programming using a tangible system. In a similar study Zuckerman and Gal-Oz (2013) found that most participants preferred the tangible system, because of physical interaction, rich feedback and high levels of realism.

3.3. Tangible Blocks

The results from the app-buttons assessment and the existent literature pointed out the advantages of tangibility for small children. We then decided to prototype and evaluate a tangible block interface. The idea, like in the app, was to allow children to see and debug their algorithms, but also promote collaboration.

The interest in fostering collaboration is based on the belief that children learn by interacting with each other. Childhood is a critical period for learning social skills, which has a significant influence on school success and life (LANDY, 2009). In this sense, tangible interfaces are seen as promoters of active collaboration (PLOWMAN; LUCKIN, 2004). Unlike graphical interfaces, where children need to alternate in controlling the programming, tangible interfaces allow simultaneous interaction. Thus, children collaborate during the programming activity, using their hands to position commands (BURLESON et al., 2018).

Historically, the initial idea of tangible interfaces was to bring the richness of human interaction with the physical world to the human-computer interaction field (HORN; BERS, 2019). Firstly this richness was explored through physical computing kits like Arduino, in which the user manipulates physical components and programs using virtual blocks or textual interfaces. Later, researchers have explored how computers can be programmed through physical objects, which the user manipulates to build runnable programs (BURLESON et al., 2018; HORN et al., 2009; MCNERNEY, 2004; ZUCKERMAN; GAL-OZ, 2013) .

An example of this type of interaction is the tangible board that we made (Figure 6a). It is a functional prototype without electronic components, composed of a small table having a glass top, in which a wood square was placed. The wood square has twelve rectangular slots, allowing the user to insert wood blocks, and a green circular star representing the start command. Four types of wood blocks represent the RoPE's actions: go forward, go backward, turn left and turn

right. When the children turn the star, the smartphone below the table reads special marks in the fitted blocks and sends the runnable program to the robot.

(a) (b)

Figure 8: Tangible programming interface and the proposed challenge.

This instantiation revealed some disadvantages of this type of interaction. Unlike in the previous app graphic interface, the tangible blocks cannot be automatically synchronized when the user clicks on a toy button. Therefore, the buttons from RoPE's "head" were replaced by a drawing of buttons, to allow the user to map between the wood block and RoPE's directions. The prototype also doesn't highlight the blocks while the robot is executing the corresponding instructions, like did by the app and other tangible interfaces.

3.3.1. Tangible Blocks Assessment

While the app assessment was quantitative, the tangible assessment qualitative. 10 children, all 5 years old, participated in the study. Following the experimental design proposed by Sapoudinis et al., (2019), the children worked in pairs, due to the interest of observing collaboration between children. Each pair solved a problem (Figure 8b) using the tangible interface and the buttons interface. In order to prevent learning effects from threatening the internal validity of the experiment, the order of interaction with the interfaces was changed between pairs.

During problem solving, we recorded video and audio from children programming the robot. The audio was transcripted and submitted to content analysis. The content analysis is a technique for making replicable and valid inferences from some content (text, audio, video) to the context of their use (SHELLEY; KRIPPENDORFF, 1984). The analysis approach was exploratory, and consisted of the analysis of word frequencies and subsequent categorization.

Frequency analysis is counting the number of occurrences of a word in a dialog. One way to represent word frequencies is through word clouds. In word clouds, the size of each word is proportional to the number of times it appears in the text, which helps in the perception of the most relevant topics during communication. The word cloud in Figure 9 was generated from the children's speech when using the tangible interface.

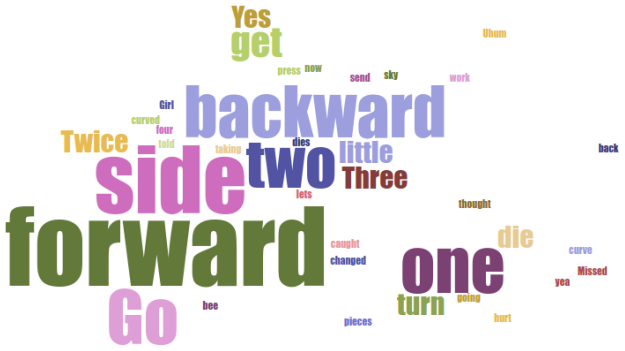

Figure 9: Word cloud from record audio.

Two remarkable aspects of the word cloud were that:

- The words “forward” and “backward” are bigger, while “left” and “right” are invisible. We can infer that the children can better distinguish front and back than left and right.
- The words related to numbers also stand out: “one”, “twice” and “three”. This can evidence the toy’s encouragement to the playful use of numbers during problem solving. It is interesting to note that the lower the number represented by the word, more frequently it is. “One” and “Two” are more present than “Three”, and “Four” also appears, but it is almost imperceptible. Possibly these numbers are a consequence of the chosen challenge, as it requires one-step and two-step movements.

After frequency analysis, we explored the transcription of children utterances following an inductive approach (ELO; KYNGÄS, 2008). Two researches read the transcripts independently, and each one categorized the utterances. Finally, the categories were analyzed and divergent categories were eliminated.

The main category found in the use of the tangible interface was "code reading". Code reading appeared in the researcher's guidelines for children to look at the pieces displayed on the board, and also in the children's opinions about the pieces placed by their peers. An example is the following dialog:

Researcher: Can you read what will happen?

Child A: No...

Child B: Backward, backward, turn, and forward.

While Children A was not able to read the code, her peer helped reading the sequence. We consider this is evidence of collaboration promoted by the tangible interface. Collaboration took place at events such as children simultaneously placing blocks on the board, pointing to blocks, and giving opinions on whether the peers’ programming was right or wrong. The dialog below is an example:

Researcher: Now, how many times does the robot need to go forward?

Children B: Two!

Researcher: Ok, then what's next?

Children B: Then two side-turns

Children A: No, just one side-turn!

It is noticeable that one child corrected the other child's response. This occurred in a playful and constructive way, in which both wanted to achieve the same goal.

Finally, the tangible interface allowed debugging, as in the following dialog:

Researcher: What did you tell me you had to do?

Children: Two forward, one sideways and one forward

Researcher: And what is here?

Children: Forward

Researcher: But there is only one... How many have you told me before? Two, isn't it?

As the algorithm created was wrong, the error became an opportunity for the researcher to use the physical blocks to concretely show the discrepancy between the algorithm that would solve the problem and the algorithm currently in the board. This allowed the children to update the algorithm gradually until they reached the correct solution.

4. Final remarks

The aim of this paper was to show the design and evaluation of three programming interfaces of a programmable toy: buttons, smartphone app, and a tangible interface. As an overview of the interfaces frequently used by programmable toys, we first showed the results of an industrial mapping study. The results show that there are more toys using virtual block-based programming interfaces. In addition, it is also clear that toys seek to offer interface alternatives, possibly to meet a wider age range.

The quantitative evaluation of the app compared to buttons interface in terms of efficacy and effectiveness. The button interface was more efficient than the application screen. Children also preferred to interact directly with the smiling, funny robot. The smartphone, on the other hand, created fear in the teacher that the children would drop it and break it. The teacher said the children also had no contact with touchscreen devices at home, which may have influenced the results.

The tangible interface had a qualitative assessment, which focused on the children's statements. Predominant words found were "forward" and "backward", while "side" replaced the use of "left" and "right", showing less distinction for the rotation movement. The same pattern occurred in the researcher's statements, and this may have influenced the results. Words related to numbers also stood out in the word cloud, which may show that the toy encourages mathematical reasoning.

Both assessments—app and tangible interface—had a few participants. Still, it was possible to detect usability problems, and desired events like fun, collaboration and positive feedback from teachers. In addition, we believe that running evaluations in an evolutionary spiral of evaluation-test-evaluation is more productive than testing the same interface with many children, as this can consume time and children available for a first contact with the interfaces.

The description of design for the three interfaces highlighted the importance of doing empirical research to improve programming interfaces for small children. Our observations showed there is much to learn from children about the way they interact with programming devices. The buttons, app screen and tangible are three types of interface, but there are many input methods available. The interaction using speech currently allows children to communicate with personal assistants, like Alexa (BIELE et al., 2019). Hand gestures are an important example of non-verbal communication (VENDRAMINI; HEEMANN, 2020), and can contribute to research with children as they often speak vaguely. Future research should explore these input methods applied to the design of programmable toy interfaces.

References

BBC LEARNING. **What is computational thinking? - Introduction to computational thinking - KS3 Computer Science Revision**. Disponível em: <https://www.bbc.co.uk/bitesize/guides/zp92mp3/revision/1>. Acesso em: 29 maio. 2020.

BIELE, Cezary et al. **How Might Voice Assistants Raise Our Children?** (W. Karwowski, T. Ahram, Eds.)Intelligent Human Systems Integration 2019. **Anais**...: Advances in Intelligent Systems and Computing.Cham: Springer International Publishing, 2019

BRACKMANN, Christian Puhlmann. **Desenvolvimento do Pensamento Computacional Através de Atividades Desplugadas na Educação Básica**. Tese— Porto Alegre, BR-RS: Universidade Federal do Rio Grande do Sul, 2017.

BRENNAN, Karen; RESNICK, Mitchel. New frameworks for studying and assessing the development of computational thinking. p. 25, 2012.

BURLESON, Winslow S. et al. Active Learning Environments with Robotic Tangibles: Children's Physical and Virtual Spatial Programming Experiences. **IEEE Transactions on Learning Technologies**, v. 11, n. 1, p. 96–106, 1 jan. 2018.

CATLIN, Dave et al. **EduRobot's Taxonomy and Papert's Paradigm**. Vilnius, Lithuania: [s.n.].

ELO, Satu; KYNGÄS, Helvi. The qualitative content analysis process. **Journal of Advanced Nursing**, v. 62, n. 1, p. 107–115, 2008.

FLANNERY, Louise P. et al. **Designing ScratchJr: support for early childhood learning through computer programming**. Proceedings of the 12th International Conference on Interaction Design and Children - IDC '13. **Anais**... In: THE 12TH INTERNATIONAL CONFERENCE. New York, New York: ACM Press, 2013Disponível em: <http://dl.acm.org/citation.cfm?doid=2485760.2485785>. Acesso em: 26 mar. 2020

FRASER, N. **Ten things we've learned from Blockly**. 2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond). **Anais**... In: 2015 IEEE BLOCKS AND BEYOND WORKSHOP (BLOCKS AND BEYOND). out. 2015

HAMILTON, Megan et al. An Emerging Technology Report on Computational Toys in Early Childhood. **Technology, Knowledge and Learning**, v. 25, n. 1, p. 213–224, mar. 2020.

HORN, Michael; BERS, Marina. Tangible Computing. In: FINCHER, S. A.; ROBINS, A. V. (Eds.). . **The Cambridge Handbook of Computing Education Research**. 1. ed. [s.l.] Cambridge University Press, 2019. p. 663–678.

HORN, Michael S. et al. **Comparing the use of tangible and graphical programming languages for informal science education**. Proceedings of the 27th international conference on Human factors in computing systems - CHI 09. **Anais**... In: THE SIGCHI CONFERENCE. Boston, MA, USA: ACM Press, 2009Disponível em: <http://dl.acm.org/citation.cfm?doid=1518701.1518851>. Acesso em: 1 jun. 2019

HORN, Michael S.; CROUSER, R. Jordan; BERS, Marina U. Tangible interaction and learning: the case for a hybrid approach. **Personal and Ubiquitous Computing**, v. 16, n. 4, p. 379–389, abr. 2012.

HSU, Yu-Chang; IRIE, Natalie Roote; CHING, Yu-Hui. Computational Thinking Educational Policy Initiatives (CTEPI) Across the Globe. **TechTrends**, v. 63, n. 3, p. 260–270, maio 2019.

LANDY, Sarah. **Pathways to competence: Encouraging healthy social and emotional development in young children, 2nd ed**. Baltimore, MD, US: Paul H Brookes Publishing, 2009. p. xxvi, 640

MCNERNEY, TimothyS. From turtles to Tangible Programming Bricks: explorations in physical language design. **Personal and Ubiquitous Computing**, v. 8, n. 5, set. 2004.

NACHER, Vicente et al. Multi-touch gestures for pre-kindergarten children. **International Journal of Human-Computer Studies**, v. 73, p. 37–51, jan. 2015.

PIAGET, Jean; INHELDER, Bärbel. **A representação do espaço na criança**. Porto Alegre: Artes Médicas, 1948.

PLOWMAN, L.; LUCKIN, R. Interactivity, interfaces, and smart toys. **Computer**, v. 37, n. 2, p. 98–100, fev. 2004.

REIS, Alessandro Vieira dos; GONÇALVES, Berenice dos Santos. Interfaces Tangíveis: Conceituação e Avaliação. **Estudos em Design**, v. 24, n. 2, p. 92–111, 2016.

RIEDO, Fanny et al. A Two Years Informal Learning Experience Using the Thymio Robot. In: RÜCKERT, U.; JOAQUIN, S.; FELIX, W. (Eds.). . **Advances in Autonomous Mini Robots**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p. 37–48.

ROQUE, Ricarose Vallarta. **OpenBlocks: An Extendable Framework for Graphical Block Programming Systems**. Dissertation—Massachusetts: Massachusetts Institute of Technology, 2007.

SAPOUNIDIS, Theodosios et al. Tangible and graphical programming with

experienced children: A mixed methods analysis. **International Journal of Child-Computer Interaction**, v. 19, p. 67–78, 2019.

SAPOUNIDIS, Theodosios; DEMETRIADIS, Stavros; STAMELOS, Ioannis. Evaluating children performance with graphical and tangible robot programming tools. **Personal and Ubiquitous Computing**, v. 19, n. 1, p. 225–237, 2015.

SHELLEY, Mack; KRIPPENDORFF, Klaus. Content Analysis: An Introduction to its Methodology. **Journal of the American Statistical Association**, v. 79, n. 385, p. 240, mar. 1984.

TANG, Kai-Yu; CHOU, Te-Lien; TSAI, Chin-Chung. A Content Analysis of Computational Thinking Research: An International Publication Trends and Research Typology. **The Asia-Pacific Education Researcher**, v. 29, n. 1, p. 9–19, fev. 2020.

VENDRAMINI, Louise Clarissa; HEEMANN, Adriano. O papel dos gestos na interação designer-usuário. **Estudos em Design**, v. 25, n. 1, p. 18, 2020.

YU, Junnan; ROQUE, Ricarose. A review of computational toys and kits for young children. **International Journal of Child-Computer Interaction**, v. 21, p. 17–36, set. 2019.

ZUCKERMAN, Oren; GAL-OZ, Ayelet. To TUI or not to TUI: Evaluating performance and preference in tangible vs. graphical user interfaces. **International Journal of Human-Computer Studies**, v. 71, n. 7–8, p. 803–820, jul. 2013.

About the authors

André Luis Alice Raabe
Ph.D. in Informatics in Education from UFRGS (2005) has completed postdoctoral studies at Stanford University (2016). He holds a master's degree in Computer Science from PUCRS (2000) and a degree in Computer Science from PUCRS (1996). He is a research professor at UNIVALI (Itajaí Valley University - Brazil) linked to the Masters of Applied Computing and the graduate school of Education. His research focuses on Interaction Design and Children, Computer Science Education, and Maker Education.
ORCID: https://orcid.org/0000-0002-8093-2507.

Cesar Pereira Viana
Student from Applied Computing Master Program and Bachelor in Computer Science (2018) at Itajaí Valley University. Research on interface design to improve children's understanding of algorithms.
ORCID: https://orcid.org/0000-0001-6902-1932.

Cassiano Pereira Viana
Master's student in Applied Computing at the University of Vale do Itajaí and Bachelor of Computer Science. Works on the design of tools adapted for the visually impaired people learn algorithms. Is also building a platform to assist evaluation of Computational Thinking Tests.
ORCID: https://orcid.org/0000-0003-2439-5572.