Computational Thinking, Visual Impairment and Melodic

Rui Costa[1], Cristiana Araújo[2] , Alvaro Costa Neto[3], Angélica Cunha[4], Pedro Rangel Henriques[5]

[1, 2, 4, 5]ALGORITMI Research Centre/LASI, DI-University of Minho, Portugal
[3]Instituto Federal de Educação, Ciência e Tecnologia de São Paulo
Barretos, Brasil

[1]rui.diogo.costa@hotmail.com, [2]decristianaaraujo@hotmail.com, [3]alvaro@ifsp.edu.br,
[4]angelicasc22@hotmail.com, [5]pedrorangelhenriques@gmail.com

Resumen: La creciente presencia de dispositivos informáticos ha transformado las habilidades de razonamiento lógico de deseables a indispensables. Aunque siempre ha sido crucial poseer la capacidad de pensar lógicamente y descomponer problemas en partes manejables, el uso constante y el rápido desarrollo de dispositivos digitales han incrementado enormemente su valor. El Pensamiento Computacional (PC) ha surgido como un conjunto de habilidades mentales para hacer frente a los nuevos desafíos planteados por las computadoras y sus aplicaciones. Se considera ahora un nuevo dominio de conocimiento que debe ser desarrollado en las escuelas, junto con disciplinas clásicas como las Matemáticas y los Idiomas. Dada esta creciente importancia, resulta lógico considerar injusta y desalentadora la falta de apoyo inclusivo en la formación en PC, especialmente para las personas con necesidades especiales. *Melodic* fue desarrollado con el propósito de combinar música, sonidos y lógica en una plataforma de juego virtual y física, con el objetivo de capacitar a personas con y sin discapacidad visual en el ámbito del Pensamiento Computacional. Esta plataforma se compone de diversas piezas de madera que permiten a los jugadores programar una secuencia melódica utilizando operaciones similares a las instrucciones de programación de computadoras. Además, se complementa con una aplicación móvil que enriquece la experiencia de juego, desafiando a los participantes a construir diferentes melodías. La realización de cada desafío requiere la aplicación de diversas habilidades de PC, lo que permite a los jugadores entrenar su capacidad de razonamiento lógico y resolución de problemas. En este artículo se presenta el diseño, desarrollo, evolución y validación de Melodic, así como una introducción a *PathIt*, un proyecto complementario que extiende los esfuerzos inclusivos de Melodic hacia nuevos paradigmas. Ambas plataformas educativas de juegos tienen como objetivo democratizar y respaldar la formación en PC de personas con y sin discapacidad visual a través de características físicas y auditivas, con la esperanza de preparar mejor a personas de todas las edades para la presencia impactante e irreversible de los dispositivos informáticos.

Palabras clave: Pensamiento Computacional, Educación de Estudiantes con Discapacidad Visual, Aprendizaje Inclusivo, Educación Especial, Evaluación del Pensamiento Computacional

Abstract: The ever growing presence of computational devices shifted logical reasoning abilities from desirable to required. While it is factual that the abilities to think logically and break down problems into manageable pieces have always been important, the constant use and rapid development of digital devices have increased their values tenfold. Computational Thinking (CT) emerged as a set of mental skills to cope with the novel challenges that computers and their uses have raised. It is being considered a new domain of knowledge to be constructed in schools, along with Mathematics, Languages and other classic disciplines. Given this growing importance, it is only logical to consider unfair and disheartening the absence of inclusive support in CT training, especially to those with special needs. *Melodic* was developed to combine music, sounds and logic in a mix of virtual and physical game platform, aimed at training visual and visually impaired people in Computational Thinking. It is composed of several wooden pieces that allow players to program a melodic sequence using operations similar to computer programming instructions. A mobile app completes the gaming experience, challenging trainees with the construction of different melodies. Several CT skills must be applied in order to complete each challenge and in doing so, players train their mental abilities in logical reasoning and problem solving. This paper presents Melodic's design, development, evolution and validation, as well as an introduction to *PathIt*, a sibling project that carries on Melodic's inclusive efforts to new paradigms. Both educational gaming platforms aim to democratize and support CT training of visual and visually impaired people through physical and audio features, in the hopes to better prepare people of all ages to the impactful and irreversible presence of computational devices.

Keywords: Computational Thinking, Visually Impaired Students Education, Inclusive Learning, Special Education, Assessment Computational Thinking.

1. Introduction

Learning and growing as a capable human being requires overcoming several challenges through the development of important skills. Many factors influence this process, including access to teaching and learning resources, experimentation, social inclusion *etc.* While not as cemented in cultural, historical and social knowledge as more classic disciplines such as Mathematics and Literature, Computational Thinking (CT) has been gathering attention in recent years by its important role both in Information Technology development and in general problem-solving education.

In a general sense, Computational Thinking may be summarized as a problem solving method, based on typical computer programming related skills. Wing (2006) defined not only the term, but also specified the six initial cornerstone features that support it:

● *Decomposition:* breaking a problem or system down into different and easier to solve chunks using a "divide and conquer" strategy;
● *Pattern recognition:* finding similarities between problems that have been successfully solved to reuse successful strategies;
● *Abstraction:* filtering information by eliminating pointless or redundant elements, in order to simplify a problem and concentrate on what is important;
● *Algorithmic thinking:* developing a set of clear, unambiguous rules or instructions that solve a problem or accomplish a goal;
● *Logical reasoning:* analyzing and understanding the problem in hands to deduct as much new information based on existing knowledge in a clear and organized way;
● *Evaluation:* verifying if the proposed solution is effective and efficient to some specified requirement.

Wing also emphasized that Computational Thinking must be taught from early ages in order to improve its efficacy. Nonetheless, adequate educational material, tools and training programs are paramount to success in this endeavor. Also fundamental is a better preparation of educators to face these challenges, paving the way for students to reason according to CT.

There are several resources to aid in training CT, either common digital games—such as Lightbot [1] and Code.org [2]—supported on paper—like FlorestaBook (Martinelli, Zanki, Cordenonsi, & Bernardi, 2021) and ALGO+RITMO (Silva, Moura, Paula, & Jesus, 2019)—or robot-based devices—such as Robot KIBO (Sullivan, Bers, & Mihm, 2017) and Robot DOC (Silva, Fonseca, Costa, & Martins, 2021). However, there is one pervasive problem with all of these resources: they are inherently visual. This fact undermines their accessibility to visually impaired people.

That is not to say that the current scenario for CT training of visually impaired people is empty. A few proposals for fulfilling this lack have been published and tested, such as Story Blocks (Koushik, Guinness, & Kane, 2019), TACTOPI (Abreu, Pires, & Guerreiro, 2020), ACCembly (Rocha, Pires, Neto, Nicolau, & Guerreiro, 2021), Project Torino (Morrison *et al.*, 2020) and the works of Sabuncuoglu (2020). While these are undoubtedly important contributions, it is clear that the landscape of non-visually centered educational resources is still lacking and could be improved with more ideas, projects and general contributions. This fact led to the creation of Melodic, a system consisting of both software and hardware

[1] Accessible at: https://lightbot.com/

[2] Accessible at: https://studio.code.org/courses/

components that combine music, audio and tactile feedback to engage and train visually impaired people in Computational Thinking.

This article presents both a general overview of the challenges concerning the education of visually impaired people, and Melodic proposal to aid in overcoming them. It is organized into six sections: Section 2 contains a bibliographic review about challenges and limitations that visually impaired people face when learning; Section 3 presents Melodic and its components; Section 4 reports on the test of Melodic effectiveness through the physical implementation of an established CT evaluation technique; Section 5 discusses recent efforts being carried in a new system named PathIt, designed to complement Melodic work; finally, Section 6 concludes the paper with the closing arguments, while also presenting future projects.

2. Visually Impaired People

Visual Impairment (VI) is a significant sensory disability that affects millions of individuals globally. It encompasses a spectrum of VI, ranging from low vision, characterized by partial visual functionality and challenges in perceiving visual details, to legal blindness or total blindness, representing severe vision loss or the complete absence of visual perception (World Health Organization, 2022).

The experience of individuals with VI markedly differs from that of sighted individuals, particularly in terms of their perception and interaction with the world. Unlike sighted individuals who heavily rely on vision from an early age for learning and exploring the environment, individuals with VI learn and navigate their surroundings primarily through their other senses, including hearing, touch, taste, and smell.

However, it is important to recognize that individuals with visual impairment are capable of learning and acquiring knowledge just as effectively as their sighted counterparts (Pogrund, Fazzi, & American Foundation for the Blind, 2002). They possess the same intellectual abilities and potential for academic achievement. Unfortunately, the learning environment and educational materials are often designed with a heavy reliance on visual cues, which can create barriers and hinder the educational journey of students with VI.

Teaching individuals with VI requires a customized approach that addresses their distinct learning requirements. To create effective learning resources, several factors must be considered to cultivate meaningful engagement and facilitate successful learning outcomes for visually impaired learners.

One crucial aspect is the use of tactile artifacts (Alotaibi, Al-Khalifa, & AlSaeed, 2020) and materials. Texture plays a significant role in providing sensory information and can enhance the learning experience for individuals with visual impairments. By incorporating various textures, such as rough, smooth, or bumpy surfaces, tactile artifacts can convey different concepts or represent objects. For example, raised symbols or Braille can be utilized to provide tactile representations of letters, numbers, or mathematical symbols. These tactile cues allow learners to explore and interpret information through touch, enabling them to access content that is typically conveyed visually.

In addition to tactile cues, auditory cues and guidance are vital for individuals with visual impairments (Taylor & Sternberg, 1989). Auditory cues can provide essential information, feedback, and instructions. For instance, in a classroom setting, teachers can use verbal descriptions to supplement visual demonstrations or gestures. Providing clear and concise auditory cues helps students with VI understand the context, follow instructions, and engage in activities effectively. Moreover, audio resources, such as recorded lectures, audiobooks, or digital voice assistants, can further support learning by providing auditory access to educational materials.

Assistive technology plays a crucial role in facilitating the learning process for individuals with visual impairments (Hallahan & Kauffman, 2006). Advancements in technology have led to the development of various tools and devices that offer accessibility features. For example, screen readers and screen magnifiers enable visually impaired learners to access digital content by converting text into speech or enlarging text and graphical elements on electronic devices. Additionally, refreshable Braille displays allow individuals to read digital text through touch-based Braille output. These assistive technologies empower visually impaired learners by providing them with independent access to educational resources, digital platforms, and online learning materials.

Designing inclusive learning resources that prioritize non-visual sensory modalities and embrace a multi-sensory approach is essential for providing equal learning opportunities and enhancing the educational experience for individuals with visual impairments. Embracing inclusivity in education not only benefits individuals with VI but enriches the educational journey for all learners, fostering a more inclusive and holistic learning environment.

3. The Melodic System

Melodic is a multi-platform system[3] that aims to support the complex tasks of teaching and learning Computational Thinking, specially targeted at aiding visually impaired and musically inclined people (Costa, Araújo, & Henriques, 2021). It is based on a predefined set of physical and tactile blocks that represent musical operations (Figure 1) and a mobile App that identifies the spatial distribution of these blocks to compose equivalent melodies.

3.1. The Blocks

The blocks that represent musical operations are integral to the Melodic System. Figure 1 shows all currently implemented blocks, ranging from playback and loop operations (1), instrument selection (2), numbering and counting (3), musical notes (4) (beginning with *Dó* and ending with *Si*}), and velocity (5). These blocks incorporate two different roles: instructions used by users to program the melody—akin to mnemonics in assembly or keywords and tokens in programming languages—and a translation mechanism, converting from a pure physical representation into an actual melody, communicating to the Melodic App.

Figure 1. Melodic blocks used to compose musical sequences.

Being a system focused on training visually impaired students, Melodic blocks are constructed in a specific way to facilitate their handling. Several characteristics are important and help users to identify each kind of block:

● *Material:* wood was chosen as standard material for its natural texture, ease of shaping, low cost, and high durability and availability. Although wood is the standard choice, any other kind of easy-to-shape and affordable material could also be used, such as clay as an example. Another possibility would be to 3D print the blocks using different filaments, which would cost more, but generate higher quality blocks;
● *Texture:* block types (numbers, notes and special operations) are carved with different textures to be promptly identifiable by touch;
● *Color:* two contrasting tones of wood (natural or dyed) are applied in the construction to aid identification by low-vision users;
● *Shape:* blocks are formed of 3D wooden icons (shapes) on top of a standard circular base, providing instant feedback as to which specific operation a certain block represents.

The translation from a block's physical entity to a logical representation in the Melodic App is achieved through QR Codes glued to the bottom of each piece, such as shown in Figure 2. The Melodic App asks the user to scan these codes in order to identify each block, who in turn, must take care to scan only one QR Code

[3] Accessible at: https://epl.di.uminho.pt/~gepl/Melodic/

at a time to avoid misidentifying blocks.

Figure 2. Top and bottom of a block, showing the QR Code used to identify each block.

Inclusion was a centerpiece to the design of the Melodic System, both in accessibility and affordability. The design of the physical blocks and the use of low-cost, highly available materials and simple building techniques support this goal in such a way that even without modern fabrication techniques, such as 3D printing, people should be able to construct the blocks by hand and use the system to support Computational Thinking education.

3.2. The App

The mobile App is the digital counterpart to the implementation of the Melodic System. Its intended use is to scan a distribution of blocks, in an order dictated by the user, and playback either the resulting melody or a message explaining that some error has occurred.

To minimize incompatibilities between different mobile devices' brands and models, Melodic App was constructed using the React Native framework[4], which uses JavaScript code to build apps for different platforms.

3.3. Use and Examples

The use of the Melodic System may be described in three main steps:

1. A sequence of blocks is distributed on a surface in a logical manner to form a melody. There is no obligation as to which kind of relative positioning is used (vertical, horizontal, tree-like etc.), but to avoid errors in the next step, a common rule for distribution may be established by the user;
2. Each block's QR Code in the sequence is scanned through the mobile camera (Figure 3), taking care to avoid showing the App more than one QR Code at a time. To this intent, it is useful to attach the QR Codes to the blocks' bottoms and turn them over one at a time;
3. The App recognizes the sequence and plays back the resulting melody (or the error message).

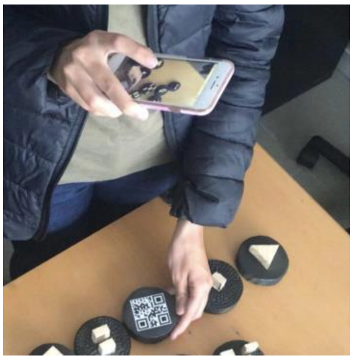

Figure 3. Blocks scanning process that allows the App to identify the sequence.

In a teacher-guided situation, exercises might be conducted using either pre-recorded melodies or by constructing the required melodies live using the blocks and the App. In the later case, the teacher should require students to only listen to the melody and not show them the sequences.

A simple melody that could be used as an introductory example or exercise is shown at the top of Figure 4. This sequence, scanned from left to right, indicates that a new melody is starting (the *Play* block), then a *Dó* note is played, followed by the *Mi* note. The sequence finishes the melody with the *Stop* block. As long as the user (student or teacher) scans the blocks in the right order (*Play*, *Dó*, *Mi*, *Stop*), the melody should be correct.

Complex melodies may Computational Thinking concepts such as change of state (by changing the instruments or the speed) and repetition (through the *Loop* or *X times* blocks). The bottom image in Figure 4 shows a complex sequence that changes the state of the melody via the *Fast* block, defines the instrument to be played with the *Piano* block and applies repetition using the *Three times* block before the *Mi* note.

[4] Accessible at: https://reactnative.dev

Figure 4. Melody sequences ranging from simple to complex.

3.4. Formal Melodic Language Definition

The fact that a melody is logically created using concepts such as changes of state and repetitions indicates that the Melodic sequences obey an intrinsic language that can be formally defined as such:

Terminals: START, STOP, REPEAT, EREP, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, SPEED1, SPEED2, SPEED3, DO, RE, MI, FA, SOL, LA, SI, GUIT, PIANO, FLUTE;

Productions:
- **Prog:** START Inst+ STOP;
- **Inst:** Note|NoteRep|Speed|Instrument|Cycle;
- **NoteRep:** Note Num;
- **Cycle:** REPEAT Num CycleInst+ EREP;
- **CycleInst:** Note|NoteRep|Speed|Instrument;
- **Note:** DO|RE|MI|FA|SOL|LA|SI;
- **Num:** 1|2|3|4|5|6|7|8|9|10;
- **Speed:** SPEED1|SPEED2|SPEED3;
- **Instrument:** GUIT|PIANO|FLUTE;

This formal definition supports the main objective of teaching and learning Computational Thinking. It proves that a melody correctly constructed through Melodic blocks is an actual algorithmic implementation, akin to the source code for an application written in a computer programming language [5]. For more detailed information on the Melodic System, see (Costa *et al.*, 2021;Costa, 2022).

4. Computational Thinking Assessment

Melodic contribution to teaching and learning Computational Thinking was assessed through a direct, progressive and extensive approach:

1. Users participated in a diagnostic session in order to assess their Computational Thinking skills before Melodic was introduced (Subsection 4.4);
2. All of Melodic components were explained and a series of exercises applied aiming to develop and train their Computational Thinking skills. These exercises ranged from the construction of a very simple, one note sequence, to more complex, iterative and sophisticated melodies;
3. Finally, a sequence of Computational Thinking tests was applied in an evaluation session to observe the users' progress.

The tests mentioned in item 3 were carefully selected in order to obtain an overview of how Computational Thinking development occurred as a result of Melodic use.

A structured strategy was defined in order to identify Melodic contribution to Computational Thinking development. Instead of using Melodic itself as an evaluation tool, which could bias the success rate by repetition and familiarity with the system, the assessment was based on simple logical exercises, designed specifically for this purpose (Zapata-Cáceres, Martín-Barroso, & Román-González, 2020). These exercises were proposed to understand users' reasoning and algorithmic design abilities.

Two different kind of exercises were applied:

1. Simple and direct exercises to test pattern matching and algorithmic structures recognition, such as the one shown at the top of Figure 5. This kind of exercises was proposed to evaluate a user's capacity to correlate the reasoning behind the drawing sequence—a line drawn on the left side of the image—and the directions that should be taken in order to draw it—the arrows in columns A, B and C;
2. Complex algorithmic reasoning exercises, where a valid path had to be constructed in order to reach a

[5] That is not to say that Melodic formal definition generates a language that is computationally equivalent to a lambda-complete programming language.

destination. This kind of exercise, as shown at the bottom of Figure 5 demands users to construct the path from beginning to end positions, without a pre-established guide. The only requirements are that the path should start at the source—the chick—follow only connected cells, avoid obstacles—the cat—go through required cells—the brown chick—and stop at the destination—the hen.

The complexity of each exercise was defined by the presence of elements in the grid, such as obstacles and required cells, and the available options of answers, that could demand recognition of algorithmic concepts such as repetitions and blocks of operations. An example of the latter is shown in option A at the bottom of Figure 5, where the first two arrows are grouped inside a rectangle—indicating that they form a block of operations—that should be repeated—represented by the preceding “2x” label.

While these exercises were used as a basis for testing users' Computational Thinking development after practicing with Melodic, they were not applied in their original form. Since Melodic is a system aimed at training Computational Thinking to visually impaired people, a series of adaptations had to be made in order to reach its intended audience.

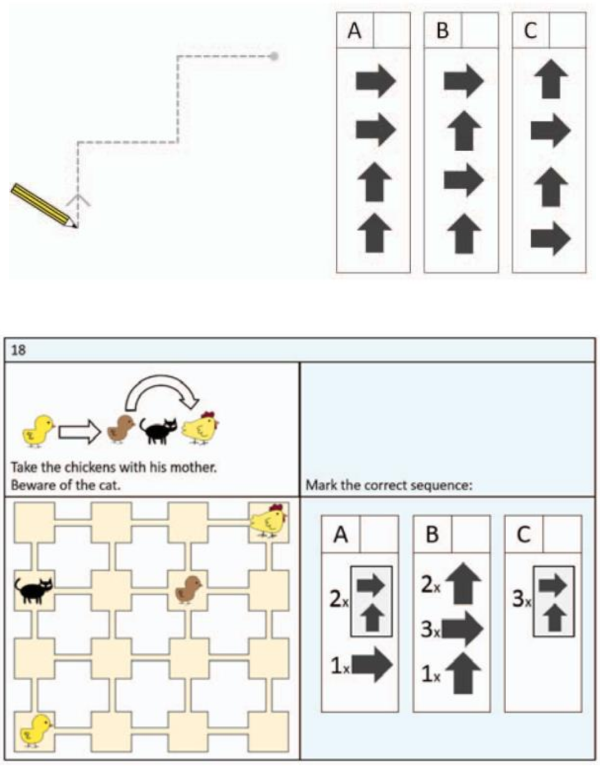

Figure 5. A simple exercise (top) to identify which of the proposed sequences (A, B or C) would draw the line on the left and a complex exercise (bottom) to identify a valid path between a source and a destination, while avoiding obstacles (the cat) and attending to required points (the brown chick) (from Zapata-Cáceres *et al.* (2020)).

4.1. Adapting the Exercises to Visually Impaired Users

Just as it was done with the blocks, the grid and the answers had to be physically represented through the use of tactile materials, such as wood or clay. For the same reasons stated in Subsection 3.1, wood was chosen as the default material for the construction. Its texture, versatility and availability made it the optimal choice to implement the grid, the paths and the elements in the cells. Figure 6 shows the engraved wooden grid (board) and the pieces used to represent both the problems and the answers of the exercises. As can be seen in the image, there are chiseled channels that represent the connections between cells, which in turn are indicated by square recesses. There are also wooden pegs (dowels) to construct the path and four wooden pieces (shown in Figure 7): circles for the source, triangles for the destination, squares to define obstacles, and crosses to establish required cells.

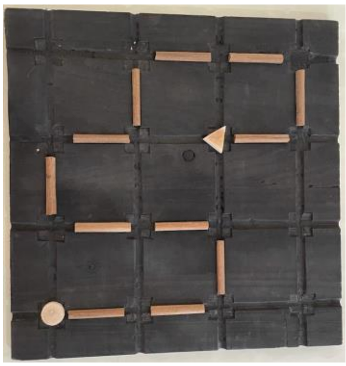

Figure 6. Example of a simple exercise implementation using the physical grid and its parts.

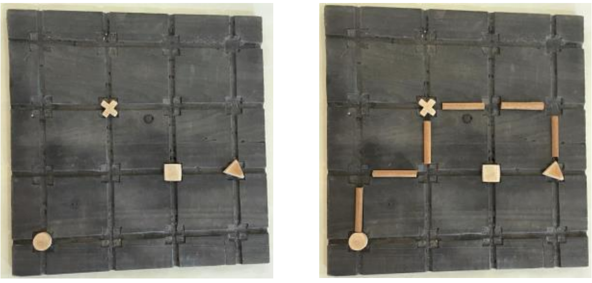

Figure 7. Complex exercise implementation using all possible pieces. This represents the most complex level of exercise. Initial state on the left and a valid answer to the right.

All parts were stained (or had their wood species chosen by their natural colors) to present visual contrast, aiming to aid users with low vision, while the shapes and engravings (channels and recesses) were created to support people with total vision loss. Using these physical implementations—the board, the pegs and the pieces—both kinds of exercises from Section 4 were adapted and applied.

4.2. Step 1: Simple Exercises

Exercises of the simple kind were implemented with some adaptations. Presenting multiple choices of answers as shown in columns A, B and C in Figure 5 would be inaccessible to blind users, so the final path was constructed and promptly presented such as shown in Figure 6. In order to complete this exercise, users had to completely describe the path in an algorithmic manner, such as: two times right, up, two times left etc.

Users were evaluated by both time to completion and conciseness of the answer, including the use of algorithmic structures and operation blocks (groupings). Three levels of answers were defined:

● *Basic*, for answers that missed all possible groupings of steps and algorithmic structures, being "Left, left, up, right, up, right" an example;
● *Proficient*, where single steps were grouped and repetitions were used with single inputs, such as "Two times left";
● *Expert*, for answers that grouped multiple steps and used them as inputs to other structures, effectively combining both concepts, as in "Two times (up, right)".

4.3. Step 2: Complex Exercises

For complex exercises, users were required to construct the path from a set source, to a defined destination, while also taking into consideration obstacles and required cells. Only pieces were initially placed on the board for identification. After examination, the users constructed their answers using the pegs to represent the path.

Four levels of complexity were defined for this kind of exercise:

● *Simple*, in which only the source (circle) and destination (triangle) pieces were used and users were expected to just create the shortest path between both;
● *Avoidance*, where besides the source and destination, some cells contained obstacles (square). Users were expected to create the shortest path that avoided the obstacles;
● *Requirement*, in which instead of obstacles to avoid, required cells (cross) were added. For this level of complexity, users were expect to create the shortest path between source and destination that passed through the required cells;
● *Complete*, in which all possible pieces were used to construct the most complex kind of exercise. An example of this level of exercise is shown in Figure 7.

Similarly to the simple exercise kind, the expected answers should use the lowest possible number of operations (pegs) and apply algorithmic structures, while also taking the shortest time possible to be constructed, counted from the beginning of the users' analysis to the completion of the answer.

Número 37, Enero-Junio 2023, pp 1-13

4.4. Assessment Methodology

The complete methodology was composed of a diagnostic session using the wooden grid, followed by Computational Thinking training with Melodic, and ending with evaluation exercises, once again using the wooden grid.

An incremental approach to complexity was adopted and five exercises of each kind were created, from simpler to more complex, both in diagnostic and evaluation sessions. Care had to be taken to keep complexity levels equivalent between diagnostic exercises and their evaluation equivalents, so that Melodic contribution to Computational Thinking development could be effectively measured. In order to achieve this, evaluation exercises represented rotations or mirroring of their diagnostic equivalents, such as shown in Figure 8 in which the grid was rotated twice counter-clockwise.

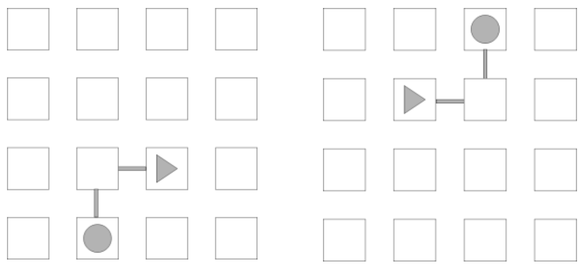

Figure 8. Rotation of the diagnostic exercise (left) to create the evaluation equivalent (right).

Users' evolution was carefully identified between the diagnostic and evaluation sessions in order to compile meaningful results for Melodic's contribution to Computational Thinking development.

4.5. Results

Two users participated in Melodic assessment, a 15 years old student and a university staff member around 40 years old, both with complete vision loss. Melodic contribution to Computational Thinking development was evaluated under three perspectives: execution time, answer complexity and number of instructions.

Execution Time

Time was measured from the beginning to the end of the exercises resolution for each user. The execution times taken to solve the simple and complex exercises are shown in Figure 9. In general, it is possible to observe that time taken to finish an exercise decreased after Melodic use (difference from the diagnostic times in blue and evaluation times in red), with just a few inversions. The first user presented worse times in the evaluation session in three simple and one complex exercises, although on average there was a 7% improvement (time reduction) for the simple exercises and 42.6% for the complex.

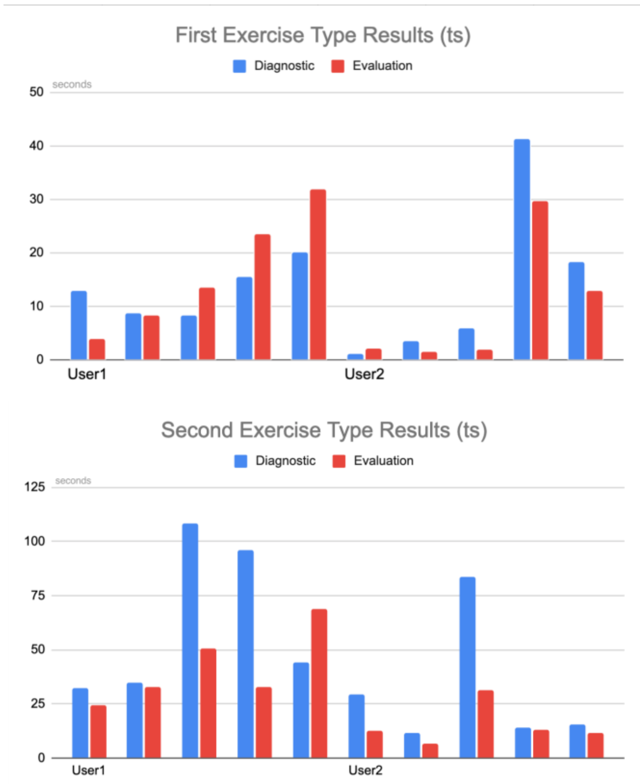

Figure 9. Execution times for all ten exercises.

Answer Complexity

Answer complexity was evaluated by asking both users to describe their reasoning to construct the exercise solutions on the board. The goal was to qualify and quantify the logic reasoning followed to build the answer. The explanation was given for the ten exercises—five simple and five complex—and the application of more sophisticated algorithmic concepts was identified. The self-described resolutions were ranked in levels of complexity, as stated in Subsection 4.2. Improvements were detected in 15% of the answers for the complex exercises. Simple exercises, given their lower challenge level, ranked equally in both diagnostic and evaluation sessions.

Number of Instructions

The number of instructions involved in each exercise solution was counted when users self-reported their reasoning for the answers, using the same strategy applied to the measurement of answer complexity. Overall, a decrease of 36\% in the number of instructions was observed between the diagnostic and the evaluations sessions. This result was only possible through the correct application of operation blocks and repetition. Both facts evidence an increase in Computational Thinking skills.

5. Technical and Conceptual Opportunities for Improvement: PathIt

The Melodic App was meticulously crafted utilizing Expo, a robust framework and platform designed for the development, construction, and deployment of universal React applications. This remarkable tool enables developers to bypass the arduous process of publishing the app on the App Store or Play Store, granting users unparalleled accessibility. By simply scanning a QR Code using the Expo GO app installed on their mobile device, users can effortlessly engage with the app's captivating features.

Unfortunately, a challenge arose after the initial release of the Melodic App when the Expo SDK utilized at that time became deprecated, resulting in the app becoming inaccessible to its user base. In response to this predicament, significant efforts were undertaken to address the matter promptly and ensure the app's continued accessibility to the public.

Therefore, the Melodic App has been recently re-released, incorporating the latest Expo SDK version, enabling users to once again enjoy its features. In a synchronized effort, the Melodic website, https://epl.di.uminho.pt/~gepl/Melodic/, has also undergone a comprehensive overhaul, incorporating the new QR Code and providing detailed instructions, ensuring a seamless and hassle-free user experience. Expanding on the achievements of the Melodic app, we are currently immersed in a captivating new venture called *PathIt*.

PathIt is an engaging and accessible game designed for both visually impaired individuals and sighted players. It offers a unique and immersive experience that combines physical game components with auditory ones from a mobile app . The game revolves around the challenge of creating a path between two positions on a game board (see leftmost image of Figure 10) while navigating through various restrictions and using sequence blocks (see the other two images of Figure 10) .

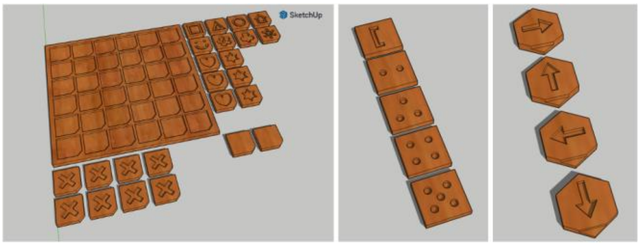

Figure 10. PathIt physical components.

The gameplay flow is as follows: players engage in a step-by-step construction process to build the game board, following voice instructions and signaling success through a shake movement. Subsequently, each challenge is presented and players utilize physical sequence blocks equipped with Near Field Communication (NFC) tags to input the desired path sequence into the mobile app. The app offers feedback and guides players through the challenges, which encompass diverse board configurations and progressively heightened difficulty levels.

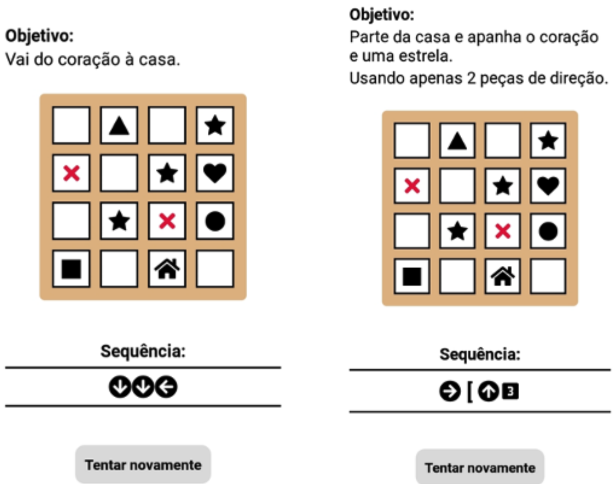

Figure 11. PathIt app challenges.

Figure 11 presents two challenges from the PathIt app to exemplify different levels of complexity and restrictions. The left image of Figure 11 depicts a straightforward challenge where the goal is to navigate from the heart symbol to the house symbol, with only two prohibited spots on the board.

On the right side of the same Figure 11, the image showcases a more intricate challenge that incorporates additional constraints. The task is to start from the home symbol, collect one heart and one star, and accomplish the objective using a maximum of two direction blocks. This challenge encourages the use of loop structures, fostering the application of Computational Thinking concepts.

PathIt promotes the development of Computational Thinking skills through its gameplay features. Users enhance their decomposition skills by breaking down the challenge into smaller steps when establishing a path. They practice abstraction by focusing on key points and disregarding irrelevant information. Algorithmic thinking is fostered through the use of repetition blocks to construct a path within imposed restrictions. Pattern recognition is honed as users recognize and reuse patterns between levels. Logical reasoning is encouraged when making decisions aligned with the main goal and constraints. Lastly, users engage in evaluation by assessing their decisions to optimize efficiency in achieving their position.

PathIt not only promotes the development of Computational Thinking skills but also strives to provide an inclusive gaming experience for all players. By engaging in the game's challenges and mechanics, users can enhance their Computational Thinking abilities while enjoying an accessible and inclusive environment.

6. Conclusion

Visually impaired people should have the same learning opportunities as everyone else. For this, it is important to design and build learning resources that are inclusive, that is, that can be used by people with or without visual impairments in their learning process.

Melodic is a learning resource that aims to allow people with or without visual impairments to train their Computational Thinking (CT) skills. Melodic is made up of blocks with symbols that have special meaning (musical notes, musical instruments, repetition, ...), and a mobile application that reads the sequence of blocks built by the learner. After reading the sequence with the app, the sequence is converted into sounds that play a melody (the origin of the LR name).

To assess the efficiency of Melodic, a three-step evaluation approach was created: diagnostic assessment of users' CT skills; training of CT using the Melodic system; and a final evaluation to measure the participants' progress. We only got two participants (one young and one adult) to test the Melodic system following the approach summarized above. The evaluation of Melodic CT training capabilities was measured through three parameters: time spent to finish the exercise; response complexity, considering the answer given to the exercise; and number of instructions of the resolution.

Regarding the execution time, we concluded that in general the exercise completion time decreased after Melodic training. The complexity of the responses was assessed by asking users to describe their reasoning for constructing the responses on the board. In the complex exercises, improvements were detected for 15% of the answers. For simple exercises improvements were not detected. The number of instructions was calculated when users self-reported their reasoning for the responses. It was concluded that in general there was a 36% decrease in the number of instructions between the diagnostic and final evaluation sessions.

However the main contribution of this paper is not the assessment of Melodic, which we recognize to be in a preliminary stage requiring its application to a larger set of participants. Instead, the main contribution is the definition of a *validation strategy for learning resources* aimed at training CT. This is, in our opinion, the focus of the work here reported, to enable us to assess properly the effectiveness of the tools we are developing to prepare students to solve problems using a computer. We do not state that the strategy proposed is free of flaws. We recognize that the evaluation techniques and tools applied to assess Melodic effectiveness might have biased the results. One feasible way to minimize this effect would be to apply a controlled study, measuring the relative evolution of a group that uses Melodic when compared to a group that does not. While not a trivial endeavor, this kind of assessment would further strengthen our current strategy. However the feedback from *Irís Inclusiva* and the people who participated was positive, as they considered that Melodic has potential, it is easy to use, and that it helps to train various CT skills as well as music.

After testing Melodic and improving it, we decided to design and build yet another inclusive resource to train CT—PathIt. PathIt is a game composed of a physical board with symbols, a mobile application to guide the challenges, and physical pieces to solve the challenges. PathIt was designed and conceived following the requirements suggested by the associations: *Irís Inclusiva* and *ACAPO*.
Both Melodic and PathIt are inclusive resources in the sense defined above and we plan to use it with children without visual problems.

As future work, we intend to finalize the implementation of the PathIt project. We consider it necessary and urgent to design and carry out more experiments (with training plans and evaluation tests) to complete the evaluation of Melodic effectiveness and evaluate PathIt. In addition, we consider it important to build usability and feedback surveys for both resources. In order to extend Melodic potential to train CT skills, we intend to create conditional and concurrency blocks to play different pieces of music in parallel and train traditional conditional algorithmic control structures. Finally, we intend to implement a computer vision module in the App to identify each block by its top, improving usability. As the blocks have different shapes and contrasting colors, the user would not need to turn the block over to scan, completely avoiding the use of QR Codes and facilitating the use of Melodic.

Acknowledgments

This work has been supported by *FCT – Fundação para a Ciência e Tecnologia* within the R&D Units Project Scope: UIDB/00319/2020

References

Abreu, L., Pires, A. C., & Guerreiro, T. (2020). TACTOPI: A playful approach to promote computational thinking for visually impaired children. *Proceedings of the 22nd International ACM SIGACCESS Conference on Computers and Accessibility*. Presented at the 22nd International ACM SIGACCESS Conference on Computers and Accessibility, Virtual Event, Greece.

Alotaibi, H., S. Al-Khalifa, H., & AlSaeed, D. (2020). Teaching Programming to Students with Vision Impairment: Impact of Tactile Teaching Strategies on Student’s Achievements and Perceptions. *Sustainability*, *12*(13), 5320.

Costa, R. (2022). *Melodic, using music to train visually impaired kids in computational thinking* (Mac Thesis). Universidade do Minho.

Costa, R., Araújo, C., & Henriques, P. R. (2021). Melodic - teaching computational thinking to visually impaired kids. *Proceedings of the Second International Computer Programming Education Conference (ICPEC 2021)*, *91*, 4:1–4:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

Hallahan, D., & Kauffman, J. (2006). *Exceptional Learners: Introduction to Special Education* (10th ed.). Boston: Pearson/Allyn and Bacon.

Koushik, V., Guinness, D., & Kane, S. K. (2019). StoryBlocks: A tangible programming game to create accessible audio stories. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–12. Glasgow: Association for Computing Machinery.

Martinelli, A., Zanki Cordenonsi, Andre, & Bernardi, G. (2021). Floresta Book – Jogo de tabuleiro para a introdução do pensamento computacional nas séries iniciais do ensino fundamental. *Temática*, *17*, 206–222.

Morrison, C., Villar, N., Thieme, A., Zahra Ashktorab, Taysom, E., Salandin, O., Zhang, H. (2020). Torino: A tangible programming language inclusive of children with visual disabilities. *Human–Computer Interaction*, *35*(3), 191–239.

Pogrund, R. L., Fazzi, D. L., & American Foundation for the Blind. (2002). Early Focus: Working with Young Blind and Visually Impaired Children and Their Families. American Foundation for the Blind.

Rocha, F., Pires, A. C., Neto, I., Nicolau, H., & Guerreiro, T. (2021). Accembly at home: Accessible spatial programming for children with visual impairments and their families. *Interaction Design and Children*, 100–111. Athens, Greece: Association for Computing Machinery.

Sabuncuoglu, A. (2020). Tangible music programming blocks for visually impaired children. *Proceedings of the Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction}*, 423–429. Sydney: Association for Computing Machinery.

Silva, R., Fonseca, B., Costa, C., & Martins, F. (2021). Fostering computational thinking skills: A didactic proposal for elementary school grades. *Education Sciences*, *11*(9).

Silva, V., Moura, H., Paula, S., & Jesus, Â. (2019). ALGO+RITMO: Uma proposta desplugada com a música para auxiliar no desenvolvimento do pensamento computacional. *Anais do XXV Workshop de Informática na Escola*, 404–413. Brasília: SBC.

Sullivan, A. A., Bers, M. U., & Mihm, C. (2017). Imagining, playing, and coding with KIBO: using robotics to foster computational thinking in young children. *International Conference on Computational Thinking Education (CTE2017), 110*.

Taylor, R. L., & Sternberg, L. (1989). Students with Visual Impairments. In *Exceptional Children: Integrating Research and Teaching* (pp. 290–315). Springer.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(March 2006), 33–35. World Health Organization. (2022). ICD-11 - Mortality and Morbidity Statistics. Retrieved from https://icd.who.int/browse11/l-m/en

Zapata-Cáceres, M., Martín-Barroso, E., & Román-González, M. (2020). Computational thinking test for beginners: Design and content validation. *Proceedings of the 2020 IEEE Global Engineering Education Conference (EDUCON)*, 1905–1914.