

# CodeRhythm: A Tangible Programming Toolkit for Visually Impaired Students

Zhiyi Rong

School of Creative Media, City University of Hong Kong  
zingaiyung@hotmail.com

Ngo Fung Chan

School of Creative Media, City University of Hong Kong  
hugochan525@gmail.com

Taizhou Chen

School of Creative Media, City University of Hong Kong  
taizhou.chen@my.cityu.edu.hk

Kening Zhu

School of Creative Media, City University of Hong Kong  
keninzhu@cityu.edu.hk

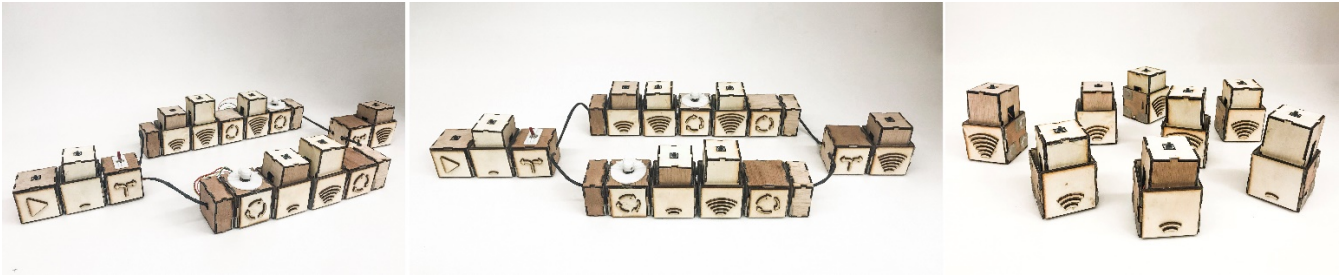


Figure 1: Pictures of CodeRhythm.

## ABSTRACT

Block-based programming toolkits are widely used to nurture computational literacy in the young generation. However, they may not be suitable for young blind and visually impaired (BVI) students as these toolkits mostly rely on visual cues in the whole manipulation process. We present CodeRhythm, a tangible programming toolkit for engaging BVI users to learn basic programming concepts by creating simple melodies.

## CCS CONCEPTS

• **Human-centered computing** → Accessibility; Accessibility systems and tools.

## KEYWORDS

Accessibility, tangible user interfaces, computer science education, multi-sensation interaction, inclusive design

## ACM Reference Format:

Zhiyi Rong, Ngo Fung Chan, Taizhou Chen, and Kening Zhu. 2020. CodeRhythm: A Tangible Programming Toolkit for Visually Impaired Students. In *The eighth International Workshop of Chinese CHI (Chinese CHI 2020)*, April 26, 2020, Honolulu, HI, USA. ACM, New York, NY, USA, 4 pages. <https://doi.org/10.1145/3403676.3403683>












Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).  
*Chinese CHI 2020, April 26, 2020, Honolulu, HI, USA*

© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
ACM ISBN 978-1-4503-8815-3/20/04...\$15.00  
<https://doi.org/10.1145/3403676.3403683>

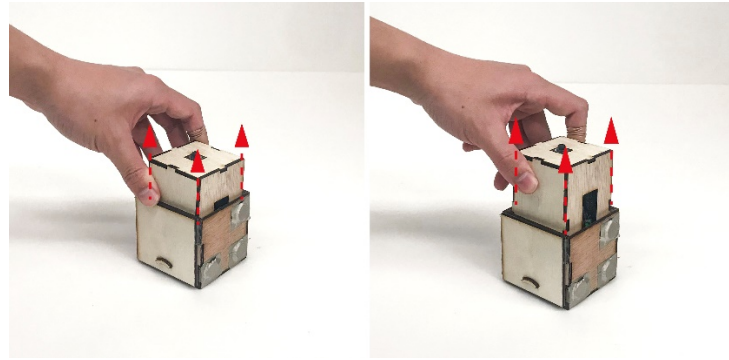
## 1 INTRODUCTION

Over the past decade, block-base programming [3, 11, 14, 17] is becoming incrementally pervasive in schools, empowering students to think creatively and systematically. Meanwhile, tangible user interfaces (TUIs) [10] have demonstrated their benefits in many aspects [6, 19–21]. With the application of TUIs for education, some new possibilities and definitions were added to block-based programming language, which provided effective and inviting pathways for learning [4, 7–9, 22]. More and more effort has also been made to engage students with special educational needs to learn basic programming knowledge [1, 2, 12, 13]. However, they mainly targeted more mature students such as high-school and college students with prior programming skills, which may result in barriers for novice BVI learners. The popular kids-coding toolkits may not be suitable to novice learners with visual disability, since these tools are primarily designed for sighted students with large amount of visual cues. To address this issue, some effort has been made to address these tangible blocks to enable BVI children to learn basic programming literacy by connecting cables between blocks and creating audio feedback [16, 18]. However, they required young BVI users to connect cables and sockets, and this process may place challenges to young BVI users whose body and spatial awareness may be affected along with the visual impairment [5].

We present CodeRhythm (Figure 1), a tangible programming toolkit for engaging visually impaired students to learn fundamental programming concepts by creating a simple melody. We adopt the form factor of blocks as fundamental elements to represent codes. The whole toolkit contains a set of blocks comprising tangible syllables blocks- do, re, mi, fa, so, la, ti - and several distinctive function blocks, representing the programming concepts of execution, looping, conditional branching. Compared to existing works,

IMAGE OF BLOCK											
NAME OF BLOCK	Do	Re	Mi	Fa	So	La	Ti	Pause	Play	Switch	Loop
DESCRIPTION OF BLOCK	Represent Note - Do	Represent Note - Re	Represent Note - Mi	Represent Note - Fa	Represent Note - So	Represent Note - La	Represent Note - Ti	Represent Pause Between Notes	Act as Execution Function	Act as Conditional Branching Function	Act as Repetition Function

**Figure 2: CodeRhythm contains two categories of blocks: syllable blocks - do, re, mi, fa, so, la, ti, and distinctive function blocks - Start, Switch, Loop, Pause**



**Figure 3: User can directly control the parameter value of duration by pulling and pushing a protruding cube on the top of each syllable block.**

CodeRhythm uses embedded magnets to connect the blocks. Our preliminary user study suggested that using magnets as the connection is helpful and preferred by BVI users. In addition, compared to existing musical programming blocks, which may only play the melody after full assembly, each syllable block of CodeRhythm can provide independent audio feedback as a complement to the tactile patterns, helping users distinguish syllable blocks. We also design the push- and-pull feature for adjusting the duration parameter of syllable, providing more variation on the sound-based programming. Lastly, CodeRhythm diverse functions, such as Switch function, which is not utilized in existing works, to provide a more inviting and diverse combination of tangible blocks. In this paper, we will describe the design features of CodeRhythm, and discuss the initial user feedback and future work.

## 2 SYSTEM OVERVIEW

CodeRhythm contains two categories of blocks (Figure 2): syllable blocks - do, re, mi, fa, so, la, ti, and distinctive function blocks - Start, Switch, Loop, Pause. All of the blocks are connected by magnets. Each syllable block represents a specific note in the melody, and the user can directly control the parameter value of duration by pulling and pushing a protruding cube on the top of each syllable block (Fig. 3).

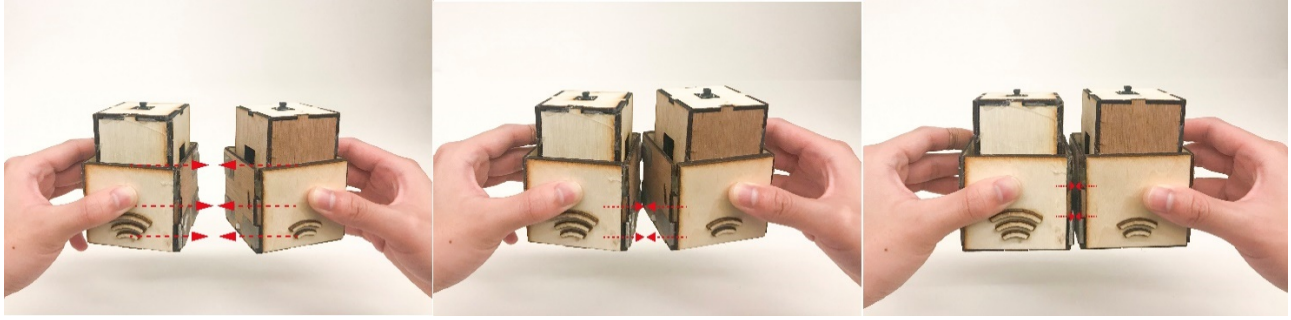
Each syllable block contains an Arduino Mini board and a speaker, enabling the user to test the audio feedback immediately without connecting all sets of blocks. The Start block acts as an

execution block, which locates on the left side of the overall sequence. The button on the top allows users to start the program. Switch Block is designed to create the conditional branching function, which enables the user to choose different programming paths. The Loop block is to repeat the syllable blocks attached to it. Pause block could be used with syllable blocks to set a duration of pause between two syllable blocks.

### 2.1 Inclusive Design Features

In this section, we will describe several inclusive design features of CodeRhythm.

**2.1.1 Clear Embodiment of Block Character.** We choose cubic block as a primary design metaphor because the flat interface of the cubic block can be combined quickly and adequately with the embed magnet. Also, this cubic metaphor can make the overall toolkit congruent in form factor. The Start block is a single cubic block, and the Syllable block has a protruding cube on the top, which can be pulled and push. For the Switch Blocks, we use one block as an original block and two small blocks as end blocks, which connected with physical cables, to show the conditional branching action in a tangible way. For the Loop block, we use a cable to combine the start block and end block, indicating that the syllable blocks set between the two blocks would be manipulated to repeat. As a result, BVI users can distinguish the primary function of each type of block easily and quickly.



**Figure 4: Demonstration of assistive block connection by magnet and conductive tape.**

**2.1.2 Assistive Block Connection.** Wire connection often causes BVI users to struggle to locate where they should plug the wire into sockets. It also disables them to distinguish if the direction of the wire is right. In contrast to using the wire as connection of blocks, each of the block in our system is connected by the embed magnet inside the interface. Regarding the signal transmission issue, we choose conductive tape as the medium. The conductive tape is attached to the block's surface, which performs well in transmitting the signal when we conduct experiments and user tests.

One of the essential advantages is that with the use of the magnet, we can decrease the difficulty of connection as much as possible (Figure 4). In this way, the user can assemble two blocks if the direction of embedded magnets matches. The magnetic force also acts as an indication of the correct connection for BVI users. (Otherwise, two blocks with the wrong connection would repulse each other.)

Another advantage is that using magnet and conductive tape as a connection method can shorten the connection length compared to wire connection. In this way, the connected blocks would form a continuous interface, enabling BVI users to follow and trace the tactile patterns rapidly.

**2.1.3 Identify Block's function by tactile and audio feedback.** Touching and recognizing tactile patterns is one of the essential ways for BVI users to distinguish blocks. Since not all children learn to read Braille, we design the easily distinguish symbols as the tactile patterns and attach them to the surface of corresponding blocks. For instance, we use a triangle as the symbol of the Start function, and we use two arrows pointing to different directions to represent the Switch function.

Besides the tactile patterns, to create a more inviting and efficient interaction mechanism for the syllable block, we install an Arduino Mini board and speaker module into the block, which controls the block independently, enabling the user to experience the audio feedback immediately without connecting all set of blocks. For example, the BVI user can pick up a syllable block, touch the tactile pattern on it and recognize which note it represents, and press the bottom on the top side to listen to the sound of the note immediately. At the same time, the user can also push or pull the protruding box to experience the duration of the note immediately. With the combination of tactile and audio sensation, CodeRhythm creates a more exciting and intuitive interaction mechanism, which provides a complementary approach to the blocks with only tactile function.

**2.1.4 Possibility of Personalization and Customization.** In addition to designing and developing syllable blocks whose duration can be adjustable, we also consider creating blocks that can provide a personalized and customized interaction experience. Currently, we are developing and testing the recording block (Figure 6), which can record and replay sound that the user intent to customize. Mixing a recording block with syllable blocks can enable a more flexible interaction scenario and cultivate user's creativity.

**2.1.5 Engage Diverse Users for Collaboration.** Although CodeRhythm is designed for BVI students, we still want to make it as inclusive as possible, that is, engaging more diverse users in the learning process. Since the tactile symbols in our system are from ordinary and simple visual elements, they are also accessible for sighted students. Thus it provides an inclusive learning environment in which BVI students and sighted students can manipulate the toolkit collaboratively.

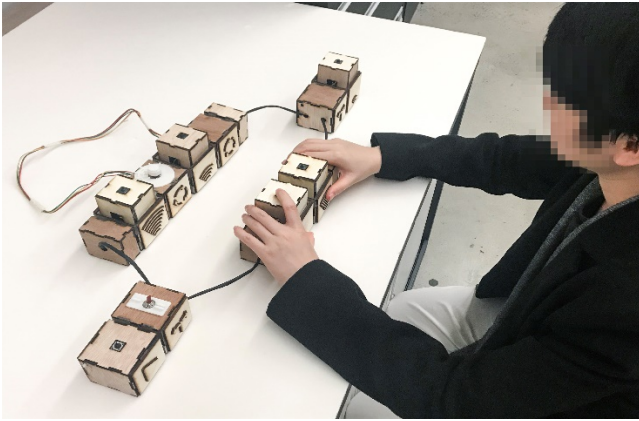
### 3 PRELIMINARY USER FEEDBACK

The initial prototype was demonstrated to a 30-year-old female with visual disability who was a BVI educator with a specialization in the music field. (Figure 7) The researchers first introduced the idea and goal of CodeRhythm and explained the function of each block. During this introduction session, researchers encouraged the participant to pick up each type of block, touch, and interact with it. Then the participant was told to connect Start Block and one of the syllable blocks, which enabled the user to create the first and basic program intuitively.

After the brief introduction, the researchers picked up one Start Block and two syllable blocks, connecting them with Switch and Loop Block, respectively. As a result, two different simple melodies were created, which introduced the participant how these two functions work in the program. In this process, the participant was told to touch and experience these two sequences of blocks.

In the next part, the researchers assemble a Start block, five syllable blocks, a Switch block, and a Loop block, to demonstrate a more diverse melody for participants. After experiencing and interacting with the overall effect of blocks, the participant was encouraged to create a personalized melody with assembling any types of blocks. In the end, we interviewed the participant about the usability challenge, the key advantages of the toolkit, and potential improvement in the future.





**Figure 5: Here our participant is creating a personalized melody.**

### 3.1 Finding

In the experience process, the participant was able to assemble blocks to create a simple melody successfully. Also, the participant could connect blocks easily without much hesitation and exploration. When tracing the block sequence by touching, the participant was able to distinguish the different characteristics of each type of block clearly.

In the interview session, the participant commented that the experience was fun and inviting. Notably, the participant thought the audio feedback, which matched the tactile pattern on the syllable blocks, was surprising and helpful. Regarding the connection with magnet and conductive tape, the participant appraised that it was beneficial and eliminated the exploration time. Also, the participant noted that we could also utilize the toolkit to help BVI students understand electric literacy since the connection was one of the significant parts of electric, and our method of using magnet and conductive tape performed well in decreasing the connection difficulties. The participant mentioned that mastering all of the concepts and functions and manipulating them in a short time was challenging and difficult, which required high-level cognitive ability. Regarding this challenge, the participant suggested that it was better to design a well-structured and detailed curriculum and applied it to a workshop of multiple sessions. For the improvement of expressiveness, the participant suggested that we can add some more diverse and personalized tunes to provide a more compelling and exciting experience. Regarding the teaching goal of CodeRhythm, the participant noted that it was useful and inviting to learn the basic and simple concept, but not proper enough for learning the advanced and traditional computational literacy.

## 4 DISCUSSION AND FUTURE WORKS

We present CodeRhythm, a tangible programming toolkit for engaging blind and visually impaired (BVI) students to learn basic computational concepts by creating simple melodies. Our preliminary user study shows that CodeRhythm is an inviting and inclusive toolkit to learn computational concepts, which decreases the usability burden as much as possible. In the future, we will design and

incorporate a well-structured and detailed curriculum into workshops of multiple sessions to teach BVI students computational thinking with CodeRhythm systematically. We will also develop blocks with more various tunes and blocks with recording and re-playing functions to expand the expressiveness of CodeRhythm. While our toolkit is designed for learning computational thinking, we are pleased to see the potential for incorporating CodeRhythm into electric literacy teaching for BVI students.

## REFERENCES

- [1] C. M. Baker, L. R. Milne, and R. E. Ladner. 2015. StructJumper: A Tool to Help Blind Programmers Navigate and Understand the Structure of Code. In *Proc of CHI '15*, 3043–3052.
- [2] J. P. Bigham, M. B. Aller, J. T. Brudvik, J. O. Leung, L. A. Yazzolino, and R. E. Ladner. 2008. Inspiring Blind High School Students to Pursue Computer Science with Instant Messaging Chatbots. In *Proc of SIGCSE '08*, 449–453.
- [3] K. Brennan and M. Resnick. 2012. New frameworks for studying and assessing the development of computational thinking. In (AERA 2012), 1–25. Retrieved September 18, 2017 from <http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf>
- [4] Leah Buechley, Mike Eisenberg, Jaime Catchen, and Ali Crockett. The LilyPad Arduino: using computational textiles to investigate engagement, aesthetics, and diversity in computer science education. 2008. In *Proc of the CHI '09*
- [5] J.M. Coleman. (2017). The use of music to promote purposeful movement in children with visual impairments. *Journal of Visual Impairment & Blindness*, 111 (1), 73–77.
- [6] A. Dancu, C. Hedler, S. A. Nielsen, H. Frank, K. Zhu, A. Pelling, A. A. Ünliker, C. Carlsson, M. Witt, and M. Fjeld. 2015. Emergent Interfaces: Constructive Assembly of Identical Units. In *Proc of CHI EA '15*. 451–460.
- [7] Michael S. Horn and Robert J. K. Jacob. 2007. Tangible programming in the classroom with Tern. In *CHI '07*
- [8] M. S. Horn, E. T. Solovey, R. J. Crouser, Robert J.K. Jacob. 2009. Comparing the use of tangible and graphical programming languages for informal science education. In *Proc CHI '09*
- [9] F. Hu, A. Zekelman, M. Horn, and F. Judd. 2015. Strawbies: explorations in tangible programming. In *Proceedings of the 14th International Conference on Interaction Design and Children (IDC '15)*. ACM, New York, NY, USA, 410–413.
- [10] H. Ishii. 2008. The tangible user interface and its evolution. *Commun. ACM* 51, 6 (June 2008), 32–36.
- [11] F. Kalelioğlu. 2015. A new way of teaching programming skills to K-12 students: Code. org. *Computers in Human Behavior*, 52, 200–210.
- [12] S. K. Kane and J. P. Bigham. 2014. Tracking @Stemxcomet: Teaching Programming to Blind Students via 3D Printing, Crisis Management, and All Twitter. In *Proc of SIGCSE '14*, 247–252.
- [13] S.K. Kane, V. Koushik, and A. Muehlbradt. 2018. Bonk: accessible programming for accessible audio games. In *Proc of IDC '18*, 132–142.
- [14] C. Kelleher, R. Pausch, and S. Kiesler. 2007. Storytelling alice motivates middle school girls to learn computer programming. In *Proc of (CHI '07)*.
- [15] Richard E. Ladner. 2015. Design for User Empowerment. *interactions* 22, 2: 24–29.
- [16] C. Morrison, N. Villar, A. Thieme, Z. Ashktorab, E. Taysom, O. Salandin, D. Cletheroe, G. Saul, Al. F. Blackwell, D.n Edge, M. Grayson & H. Zhang (2018): Torino: A Tangible Programming Language Inclusive of Children with Visual Disabilities, *Human-Computer Interaction*
- [17] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, Eric Rosenbaum, J. Silver, B. Silverman, and Y. Kafai. 2009. Scratch: programming for all. *Communications of the ACM* 52, 11: 60–67.
- [18] A. Thieme, C. Morrison, N. Villar, M. Grayson, and S. Lindley. 2017. Enabling Collaboration in Learning Computer Programming Inclusive of Children with Vision Impairments. In *Proc of DIS '17*
- [19] K. Zhu, A. Dancu, and S. Zhao. 2016. FusePrint: A DIY 2.5D Printing Technique Embracing Everyday Artifacts. In *Proc of DIS '16*
- [20] Kening Zhu, Taizhou Chen, Feng Han, and Yi-Shiun Wu. 2019. HapTwist: Creating Interactive Haptic Proxies in Virtual Reality Using Low-cost Twistable Artifacts. In *Proc of CHI '19*
- [21] Kening Zhu, Hideaki Nii, Owen Noel Newton Fernando, and Adrian David Cheok. 2011. Selective inductive powering system for paper computing. In *Proc of ACE '11*
- [22] Kening Zhu, Xiaojuan Ma, Gary Ka Wai Wong, and John Man Ho Huen. 2016. How Different Input and Output Modalities Support Coding as a Problem-Solving Process for Children. In *Proc of IDC '16*