Current Practices in Teaching Computational Thinking to Children: Accessibility is an AfterThought

MARTA CARVALHO, LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal
FILIPA ROCHA, LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal
JOÃO GUERREIRO, LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal
HUGO NICOLAU, ITI/LARSyS, Instituto Superior Técnico, Universidade de Lisboa, Portugal
TIAGO GUERREIRO, LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal
ANA C. PIRES, LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal

Recognition of computational thinking as a relevant skill set has increased its prevalence in school curricula, and the number of coding platforms and kits available. The inaccessibility of the latter has been a focus of recent attention resulting in the emergence of accessible approaches. Conversely, there has been limited attention to activities, how training platforms are being used in curricular practice, and how they are being adapted for children with disabilities. We present findings from a qualitative interview study with 6 IT instructors depicting their practices, experiences, and their views towards an inclusive future classroom.

Additional Key Words and Phrases: Computational Thinking, Tangibles, Education, Robots, STEM.

ACM Reference Format:
Marta Carvalho, Filipa Rocha, João Guerreiro, Hugo Nicolau, Tiago Guerreiro, and Ana C. Pires. 2018. Current Practices in Teaching Computational Thinking to Children: Accessibility is an AfterThought. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Teaching computing skills to children has emerged as a STEM discipline in school curricula. Computational Thinking is now considered a crucial skill for young children to foster their creativity, social and emotional abilities, and critical and logical thinking. In addition, children change their perspectives when learning new methodologies, which results in an overall enrichment in a variety of learning subjects [6, 18].

There is a growing number of virtual (software) [4, 5, 16] and tangible (e.g., robotic) [2, 8] programming environments to teach computational concepts – such as sequences, operations, and iterations – to children. For instance, tangibles are known to facilitate the understanding of abstract concepts with a hands-on approach enabling children to map structural and cognitive connections and develop refined motor actions, and tactile perception [10, 17, 19]. Robots, in particular, can also boost motivation and engage children in science and technology [1, 3, 15]. While some of these programming environments are already being used in schools, prior research has reported numerous accessibility concerns as most cannot be used by children with disabilities. For instance, a main focus on visual feedback makes interaction very difficult – if not impossible – for children with visual impairments [9, 11, 14, 15]. As a result, novel research prototypes that leverage and promote other senses and abilities have been implemented [7, 12, 13]. Despite the emergence of innovative solutions, these are mostly available in research contexts and therefore it is not clear how instructors are currently teaching computational skills to children with disabilities.

In this paper, we explore the current strategies employed to teach computational concepts to young children, by interviewing six IT instructors from different schools in Portugal. In particular, we wanted to understand their teaching strategies, what concepts they teach and which methods they found most important. Most importantly, we wanted to understand their prior experiences teaching children with disabilities, the technological barriers they (and children) face, and their vision for an inclusive classroom.

2 METHOD

We conducted in-person interviews with six IT instructors, aged between 22 and 57, teaching computing classes (P1-6) to children with ages ranging from 6 to 12 years old. Their experience in computing/programming ranged from 1 to 37 years. While two of them were teaching mainstream classes, four only had experience in extracurricular courses. Mainstream classes had an average of 30 children in the classroom, whereas extracurricular classes had 3-12 children in the same room. All the conducted interviews were audio-recorded and transcribed along with emotional expressions. Data was then subject to a thematic analysis.

3 FINDINGS

3.1 Practices and Strategies in Teaching Computational Concepts

Teaching Strategies All of them found that presenting the goal of the lesson at the start of class was helpful to students: "*I would say what we were going to do, a game, an app, that the goal was that*" (P4). Four out of six instructors mentioned that the lesson plan should start with basic concepts and as the class evolves the complexity would also increase: "*We start with very short things and as the year goes on [...] the more complex the task would be*" (P5). P1 and P5 prefer having short-term projects to teach a concept that students then have to apply and P3, P4 and P6 start by giving a challenge where children need to understand the concept they are going to use: "*I would try to explain a lot... make them feel they really need to know [that concept]*" (P3).

Connection to Real-World Experiences All IT instructors highlighted the importance of making associations to the real-world and daily life: "*Everything we can explain with things from daily-life, I think it's easier for children*" (P6). For instance, P1 describes an example "*[the robot] count, for example, if you were a firefighter, [...] how many people are in that building, to let you know how many people need to be saved. [...] because [the robot...] is on a mission and we're not seeing what dash is seeing, it needs to count the number of [people] detected with its sensors and return that value to us*".

Exploration and Use of Concrete material To introduce programming concepts, two participants said they first started with an exploratory class, "*the first lesson is the exploratory lesson [...] explore and teach me what you have discovered*" (P2). The participants mentioned when learning a new concept "*more concrete examples help them understand*"(P6). P3 mentioned that "*I always try to fit in with what they like and with real life. And always making associations of concepts with other things that are easier for them to understand.*". P5 also highlighted that she makes children use their body to reinforce this connection to real-world knowledge "*[...] another thing I do [...] is using the body so that they can minimally dimension the space, namely, for example, doing routines of [...] walking around this table [...] in which they have to schematize the number of steps, so they understand how many steps, their foot size.*".

They also mentioned that the use of concrete material in teaching has several advantages, in particular, facilitating abstraction and allowing to "*interact with the environment, with the inputs themselves*"(P1); and "*the robot allows* *visualization of the output*" (P1). One of the participants also mentioned that tangibles excite students, and: "*they would get motivated and learn more easily*" (P3).

Analogies to Previous Knowledge All IT instructors mentioned the importance of using analogies for teaching programming concepts, especially complex ones. P2 emphasized that "*the little ones have this need to relate everything they learn with something they already know*" - P2.

One of the most mentioned analogies by instructors was related to teaching the concept of variable, for instance, P4 that uses a box of pens/beads, and explains to children "*this is a box, it keeps things, but only keeps one at a time*" or P1 that explains: "*the pants pocket can be a variable, and therefore I keep putting cards in my pocket to count the number of trees I saw and therefore my pocket serves as a variable [...] I put cards whenever I see something, I can go putting cards or taking cards out of my pocket. And so I get to the end and I know how many trees I saw because I count the number of cards [...] and this is a variable.*

P2 said that to teach the concept of algorithm, she use the analogy of the cake recipe, of changing a light bulb, the GPS: "*GPS teaches you the way, street by street, go that way,[...] he is teaching you the algorithm*". P1 referred to the glossary for teaching the loop, a book about a cat looking for mysteries for debugging, a deck of cards and library for sorting, recipes for algorithms and functions and song-chorus also. For instance, P1 uses a storybook to explain debugging: "*a kitten who was very curious and was always solving mysteries, to explain what debugging was. [...] When debugging we are always looking for clues and trying to solve, improve, and we are always trying to see how the program improves*".

3.2 Accessibility of Computing Classes

Accessibility is an AfterThough. All participants stated that accessibility was not a particular concern in their classes, even though they all had taught students with special needs. Their experience was mainly related to children with a Functional Neurological Disorder such as Autism Spectrum Disorder, Attention Deficit Hyperactivity Disorder and Asperger Syndrome, and few had experience with children with sensory impairments, such as children with hearing or with visual impairments. IT instructors mentioned that besides the fact that technology is not accessible yet for children with special needs, IT instructors do not have special training (although they have the support of special needs (SNE) educators in reference schools) to provide children with an optimal learning experience. Participants referred to a lack of pedagogical training which lead them to adapt their classes based on their prior experience and SNE suggestions.

Participants mentioned that their lesson plan was visually demanding and acknowledged that their classes were not inclusive, mentioning that they did not have the tools to be accessible for children with vision impairments: "*The course I was teaching was not accessible at all (...) because it was very visual.. the components, everything*"(P3).

Teaching children with mixed-abilities. All participants mentioned that each child and class were different and they preferred to adopt strategies for students with greater difficulties (whether they have disabilities or not) – such as giving extra time or assigning pairs with different knowledge levels – rather than reducing the content: "*So I am not being reductive in terms of subject matter, the only thing I'm giving them is more time*" (P5). Overall, the general class dynamics did not change due to having students with disabilities, but participants mentioned several approaches, with different levels of inclusiveness, such as having "*his father was always with him*" -P1, "*assigning a pair is often a good idea. There is the concept of pair programming*" -P2; move the student to a class of a lower level "*pass him on to the kids class which is something more visual*" - P6; and give the student the freedom to manage their time autonomously,"*in a more autonomous way on his part, he would explore things on his own*" -P3.

Although participants mentioned they have previously taught students with hearing, visual or motor disabilities, they did not have much experience doing it and the necessary tools, such as accessible hardware. In those cases, students used their own tools, such as a "*laptop because it had software that had a much better magnifying glass than ours alongside a specified color scheme*" (P2); and "*an adapted mouse*" (P4) for a student with a motor impairment.

Tangibles for accessibility All instructors highlighted the potential of tangibles to teach computational concepts. When talking about children with disabilities, participants focused mostly on the advantages for children with visual impairments. In particular, tangibles have the potential for multimodal feedback, which may ensure solutions that fit classes with children with mixed-abilities. P1 highlighted that children with visual impairments could leverage both touch and sound, for instance when interacting with a robot: "*the robot moves and they go after it and change positions*". As an interactive tool, robots can also support multiple functions that children can program themselves: "*children can program the sound [...] also have lights*" (P1). In addition, robots were mentioned also as a way to promote "*collaboration between colleagues*" (P5). In fact, tangibles in general may be an important tool to promote collaboration between mixed-ability students, as long as it is available to all students.

Another advantage of tangibles is to provide a physical history of actions, which may be beneficial for children to keep track of prior actions and to be always aware of the context.

4 CONCLUSION

We interviewed six instructors to investigate the current strategies for teaching computing skills to children aged 5-12 years old. Particularly, we aimed to understand what concepts are taught and how. Furthermore, our goal was to get their perspectives on accessibility by going through their prior experiences with children with disabilities, focusing on the methods used to teach children with mixed-abilities.

All participants had different experiences and, therefore, unique teaching strategies and approaches. Thus, future tools and technologies should strive for flexibility to accommodate multiple teaching practices. Nevertheless, instructors often start with fundamental concepts such as sequences and conditionals, and progressively move to more complex ones, like debugging, variables, and functions. IT instructors afford children to explore the coding environment and to use concrete material to facilitate abstraction. They mentioned the importance of connecting the learning goals to real-world experiences and of using analogies to children's previous knowledge to facilitate young children's learning and engagement. Future technology developments should explore the connections between children with real-world experiences, analogies to children's previous knowledge and the use of concrete objects. IT instructors adapt their classes depending on the students, their difficulties and special needs, consistently suggesting accommodations for children with special needs. They preferred to adapt tasks, and the duration of an activity, and use different tools rather than reduce the content or the concepts to be taught. However, they acknowledged that current practices are not suited for the inclusion of children with special needs since the tools and materials are not accessible.

All participants saw the potential in adopting tangibles as a teaching and inclusion tool for mixed-ability classes, particularly robots. Instructors highlighted that tangibles are engaging and motivational for children, promote collaboration within the classroom, and allow children with (visual) impairments to better contextualize what they are programming.

5 ACKNOWLEDGMENTS

We kindly thank all the instructors who participated for sharing their perspectives with us.This work was supported by FCT through LASIGE Research Unit funding, ref. UIDB/00408/2020 and ref. UIDP/00408/2020.

REFERENCES

[1] Alissa N. Antle. 2007. The CTI Framework: Informing the Design of Tangible Systems for Children. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction* (Baton Rouge, Louisiana) *(TEI '07)*. Association for Computing Machinery, New York, NY, USA, 195–202. https://doi.org/10.1145/1226969.1227010

[2] L. G. C. Anzoategui. [n.d.]. Cubetto. https://www.primotoys.com/.

[3] Mayara Bonani, Raquel Oliveira, Filipa Correia, André Rodrigues, Tiago Guerreiro, and Ana Paiva. 2018. What My Eyes Can't See, A Robot Can Show Me: Exploring the Collaboration Between Blind People and Robots. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility* (Galway, Ireland) *(ASSETS '18)*. Association for Computing Machinery, New York, NY, USA, 15–27. https://doi.org/10.1145/3234695.3239330

[4] Louise P. Flannery, Brian Silverman, Elizabeth R. Kazakoff, Marina Umaschi Bers, Paula Bontá, and Mitchel Resnick. 2013. Designing ScratchJr: Support for Early Childhood Learning through Computer Programming. In *Proceedings of the 12th International Conference on Interaction Design and Children* (New York, New York, USA) *(IDC '13)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/2485760.2485785

[5] Neil Fraser. 2015. Ten Things We've Learned from Blockly. In *Proceedings of the 2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond) (BLOCKS AND BEYOND '15)*. IEEE Computer Society, USA, 49–50. https://doi.org/10.1109/BLOCKS.2015.7369000

[6] Shuchi Grover. 2020. Designing an Assessment for Introductory Programming Concepts in Middle School Computer Science. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (Portland, OR, USA) *(SIGCSE '20)*. Association for Computing Machinery, New York, NY, USA, 678–684. https://doi.org/10.1145/3328778.3366896

[7] Varsha Koushik, Darren Guinness, and Shaun K. Kane. 2019. StoryBlocks: A Tangible Programming Game To Create Accessible Audio Stories. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300722

[8] LEGO. [n.d.]. LEGO Mindstorm. https://www.lego.com/en-pt/themes/mindstorms/about.

[9] Stephanie Ludi, Lindsey Ellis, and Scott Jordan. 2014. An Accessible Robotics Programming Environment for Visually Impaired Users. In *Proceedings of the 16th International ACM SIGACCESS Conference on Computers & Accessibility* (Rochester, New York, USA) *(ASSETS '14)*. Association for Computing Machinery, New York, NY, USA, 237–238. https://doi.org/10.1145/2661334.2661385

[10] Andrew Manches and Claire O'Malley. 2012. Tangibles for Learning: A Representational Analysis of Physical Manipulation. *Personal Ubiquitous Comput.* 16, 4, 405–419. https://doi.org/10.1007/s00779-011-0406-0

[11] Oussama Metatla and Clare Cullen. 2018. Bursting the Assistance Bubble: Designing Inclusive Technology with Children with Mixed Visual Abilities. 1–14. https://doi.org/10.1145/3173574.3173920

[12] Lauren R. Milne and Richard E. Ladner. 2018. Blocks4All: Overcoming Accessibility Barriers to Blocks Programming for Children with Visual Impairments. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3173574.3173643

[13] Cecily Morrison, Nicolas Villar, Anja Thieme, Zahra Ashktorab, Eloise Taysom, Oscar Salandin, Daniel Cletheroe, Greg Saul, Alan F Blackwell, Darren Edge, Martin Grayson, and Haiyan Zhang. 2020. Torino: A Tangible Programming Language Inclusive of Children with Visual Disabilities. *Human–Computer Interaction* 35, 3 (2020), 191–239. https://doi.org/10.1080/07370024.2018.1512413

[14] Isabel Neto, Hugo Nicolau, and Ana Paiva. 2021. Community Based Robot Design for Classrooms with Mixed Visual Abilities Children. 1–12. https://doi.org/10.1145/3411764.3445135

[15] Ana Cristina Pires, Filipa Rocha, Antonio José de Barros Neto, Hugo Simão, Hugo Nicolau, and Tiago Guerreiro. 2020. Exploring Accessible Programming with Educators and Visually Impaired Children. In *Proceedings of the Interaction Design and Children Conference* (London, United Kingdom) *(IDC '20)*. Association for Computing Machinery, New York, NY, USA, 148–160. https://doi.org/10.1145/3392063.3394437

[16] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, and Yasmin Kafai. 2009. Scratch: Programming for All. *Commun. ACM* 52, 11, 60–67. https://doi.org/10.1145/1592761.1592779

[17] Anja Thieme, Cecily Morrison, Nicolas Villar, Martin Grayson, and Siân Lindley. 2017. Enabling Collaboration in Learning Computer Programing Inclusive of Children with Vision Impairments. In *Proceedings of the 2017 Conference on Designing Interactive Systems* (Edinburgh, United Kingdom) *(DIS '17)*. Association for Computing Machinery, New York, NY, USA, 739–752. https://doi.org/10.1145/3064663.3064689

[18] Jeannette M. Wing. 2006. Computational Thinking. *Commun. ACM* 49, 3 (March 2006), 33–35. https://doi.org/10.1145/1118178.1118215

[19] Junnan Yu, Clement Zheng, Mariana Aki Tamashiro, Christopher Gonzalez-millan, and Ricarose Roque. 2020. CodeAttach: Engaging Children in Computational Thinking Through Physical Play Activities. In *Proceedings of the Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction* (Sydney NSW, Australia) *(TEI '20)*. Association for Computing Machinery, New York, NY, USA, 453–459. https://doi.org/10.1145/3374920.3374972