
LAUTSPRECHER
RGB LED
BATTERIE
CALLIOPE
mini

Swamy Ananthanarayan and **Susanne Boll,**
University of Oldenburg

Physical Computing for Children:

Shifting the Pendulum Back to Papertian Ideals

Insights

- Physical computing kits should be dependent on the developmental considerations of children.
- Lack of visibility and trying to make things "just work" hinders the exploration of programming concepts.
- Learning programming should be a healthy side effect of learning to solve problems and expressing ideas through language.

These days, children have many types of physical computing platforms, robotics devices, and computational toys available to them in both educational and domestic settings. The abundance of these devices for children are the fruits of research initiatives that began in the late 1960s. Seymour Papert's constructionist principles about actively "creating and constructing meaningful" [1] artifacts as a way to build knowledge stands out among various learning theories. These ideas actively engage the learner in the learning process, but more important, help them create a "public entity" [2] that can be shared socially. More recently, Papert's principles have been repopularized by the Maker movement, which has focused on democratizing technology for various user groups. Particular to children, these technologies have been used to educate children in computational thinking. The assumption has been that by helping children make computationally intelligent artifacts, the children would in turn learn computational thought. So, how far have we come in realizing Papert's vision of student-centered, discovery-based learning?

We recently explored this question with a fairly new physical computing

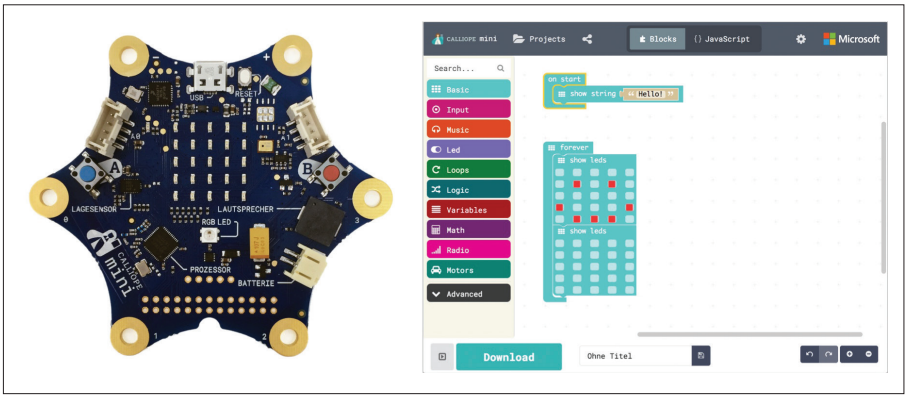

Figure 1. The Calliope mini physical computing platform with a Web-based programming environment that uses blocks to represent program instructions.

platform called the Calliope mini (calliope.cc) with a small group of primary-school children. Much like its ancestor, the BBC micro:bit, it is designed to introduce children, elementary-school age and higher, to actively get involved in building new computational applications and artifacts. In this article, we aim to initiate a discussion about the design of physical computing platforms for children. Based on our preliminary work, this area could be suffering from a version of the QWERTY phenomenon—importing technologies from other professional fields without the necessary adaptations for their use by children.

EXPLORING CALLIOPE

Shaped like a six-pointed star (Figure 1) with connection points at the edges, the Calliope integrates 25 red LEDs, an RGB LED, two push buttons, a small piezo speaker, a motor driver, wireless Bluetooth communication, and a variety of sensors (e.g., microphone, accelerometer, compass, temperature, brightness). It can be programmed from any Web browser using a block-based language and uses a micro USB cable for program transfer.

Our goal was to assess how well Calliope integrated with children's approaches to crafting and making things, and also how well it supported their programming capabilities. Did it facilitate the discovery of programming-language concepts? In partnership with a local elementary school, we organized an after-school Calliope workshop with 10 children (boys and girls, five each) who were 8 or 9 years old. The two-hour workshops occurred twice a month from late August to early January; we were able to hold a total of 10 sessions.

The workshops were designed to follow constructionist principles, with the intent of helping children discover Calliope for themselves through creative experimentation rather than through lectures or step-by-step guidance [2]. As adults, our roles were more mediational than instructional. Although we periodically provided brief overviews of Calliope's capabilities through examples, children were left to explore Calliope on their own, with two to three adults as roaming facilitators. To provide a common shared goal for the children, we presented two broad gender-neutral project topics. From August to October, the theme was to develop an interactive Halloween costume, and from November to January, children worked on creating an interactive backpack accessory. During each workshop, we maintained a communal area consisting of a poster and a table where children could display their work and discuss their ideas (Figure 2). This is in line with an important aspect of Papert's philosophy, which includes the public nature—the presentation and sharing—of artifacts that children create [3]. To scaffold the design process, we provided paper models for prototyping design ideas for the two project topics prior to implementation (Figure 2).

The workshops were designed to follow constructionist principles, with the intent of helping children discover Calliope for themselves through creative experimentation.

COSTUMES AND BACKPACKS

We found children to be enthusiastic and eager to receive their own personal Calliopes; they even personalized their Calliope cardboard boxes with symbols and drawings. Moreover, children were excited to immediately start exploring the programming environment. While some of this enthusiasm settled over time, as the novelty of the new platform wore off, kids were generally up for trying new things every session. For example, the discovery of how to generate tones on the piezo speaker rippled across the classroom until the majority of Calliopes were generating some melody or tune. For some children, the Calliope could simply not keep up with their imagination, with the platform's actual capabilities falling short of their expectations. One girl dropped out of the class during the Halloween costume project because she wanted to make a Darth Vader voice modulator, a task that is hard to accomplish without significant work. For the other students, we observed a period where expectations were recalibrated as they slowly learned what the Calliope could and could not do.

While making the costumes, children primarily employed the light display and the RGB LED to in some way accentuate the crafted costume. For example, one 8-year-old girl made butterfly wings with a small cutout in the middle to display a heart on the Calliope LED display (Figures 3a, 3d). A few children also used the buttons and the piezo speaker to make the costume a bit more interactive. In the grim reaper costume showcased in Figures 3b and 3c, the Calliope not only displays a sickle but also reacts to buttons on either side of the board. If button A is pressed, the LED

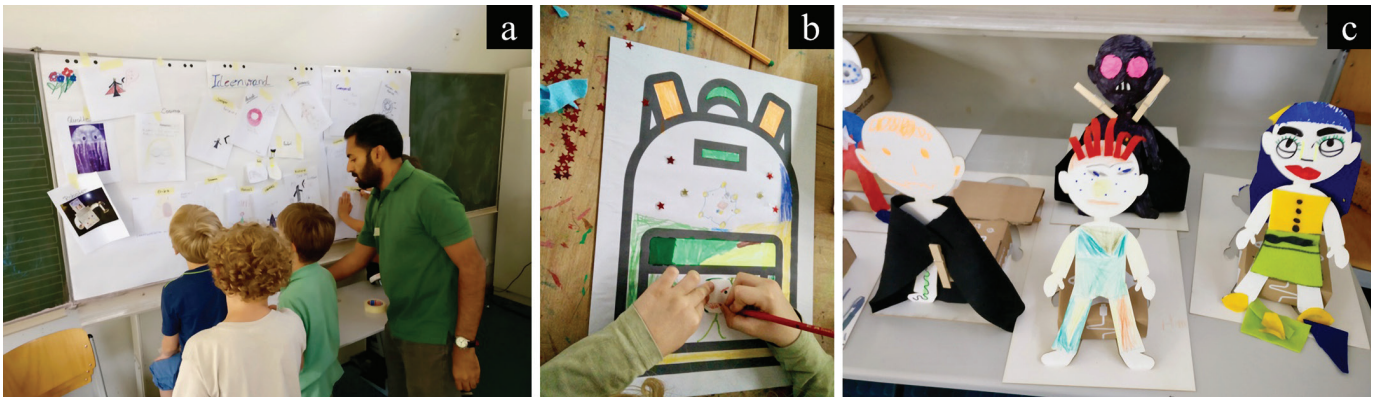

Figure 2. Sharing and scaffolding the design process: a) communal poster for displaying ideas b) backpack paper-based template for prototyping c) popup figure for costume design.

display scrolls the text "I kill you," and if button B is pressed, it plays an ominous melody. Although children experimented with the sensors, particularly the light sensor and the accelerometer, they were hesitant to use them for this project. Perhaps an interesting aspect was that children did not really distinguish between the different sensors and considered the Calliope itself to be the sensor.

Children, however, ventured beyond their comfort zone during the backpack accessory project. The use of sensors was more prevalent, and children who were initially using the Calliope as a display experimented with buttons and audio. For example, the 8-year-old girl who had made the butterfly wings (Figure 3d) utilized the Calliope this time to play a tone and display a little figure when button B was pressed (Figure 4a). Three of the projects used the light sensor to turn the Calliope into a small flashlight to better illuminate the contents of the pack when it was opened (Figure 4b). A few children experimented with external LEDs to aesthetically accentuate their backpacks (Figure 4c); these lights were connected with wires to the output ports and often were controlled via the Calliope's buttons. For three children, the accelerometer was used to idiosyncratically display various images on the LED screen or turn the RGB LED different colors. For example, one child displayed a little stick figure when the backpack was shaken.

SEPARATION OF CRAFTING AND PROGRAMMING

An interesting observation from our workshop was that although children were excited to program the Calliope and integrate it into their projects, there was a distinct separation between crafting and programming. During sessions, children focused on one or the other. Typically, children would craft their artifacts or costumes and then figure out how to integrate the Calliope afterward. This could be in part due to their still-maturing planning capabilities, but it could also

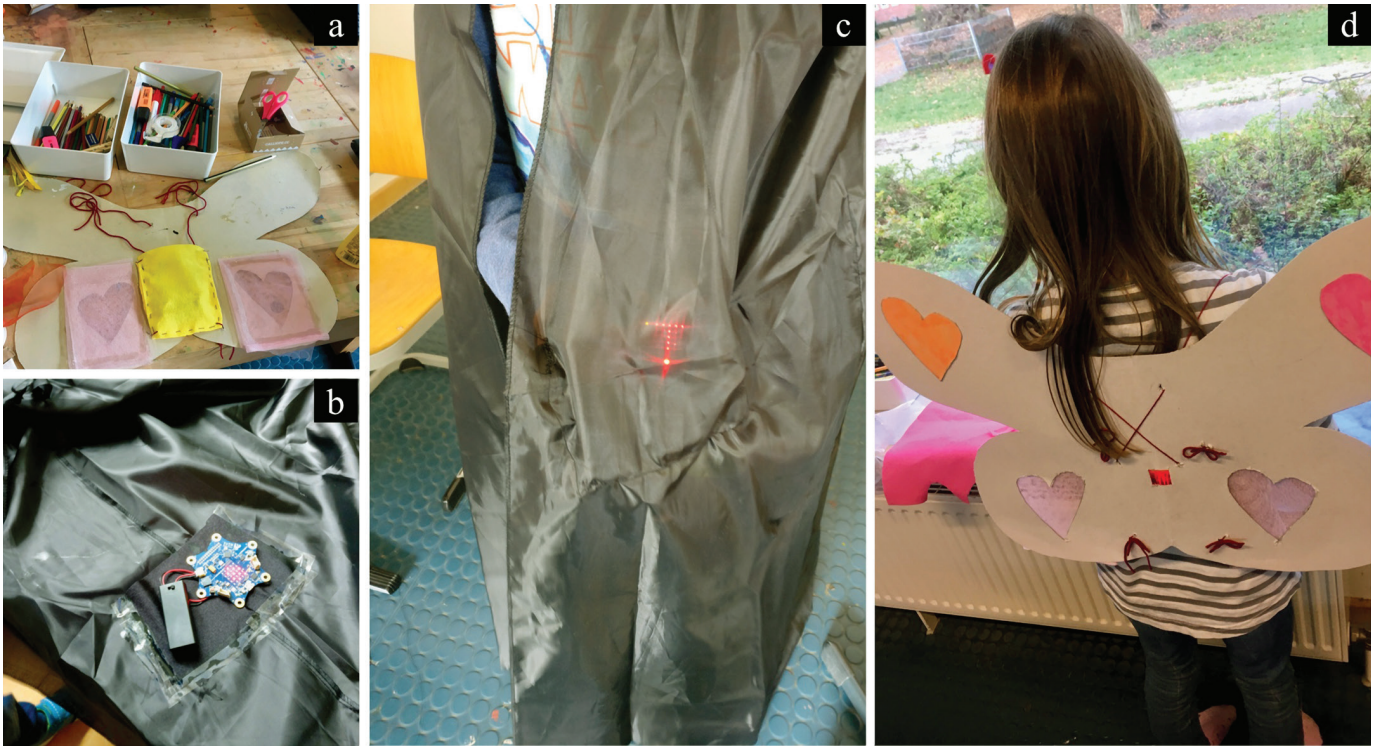

Figure 3. Calliope Halloween costumes: a) butterfly wings with Calliope integrated in a yellow felt pouch b) grim reaper cape with Calliope integrated in a fabric pocket c) a sickle shown on the LED display d) a heart shown on the LED display in the middle of the butterfly wings.

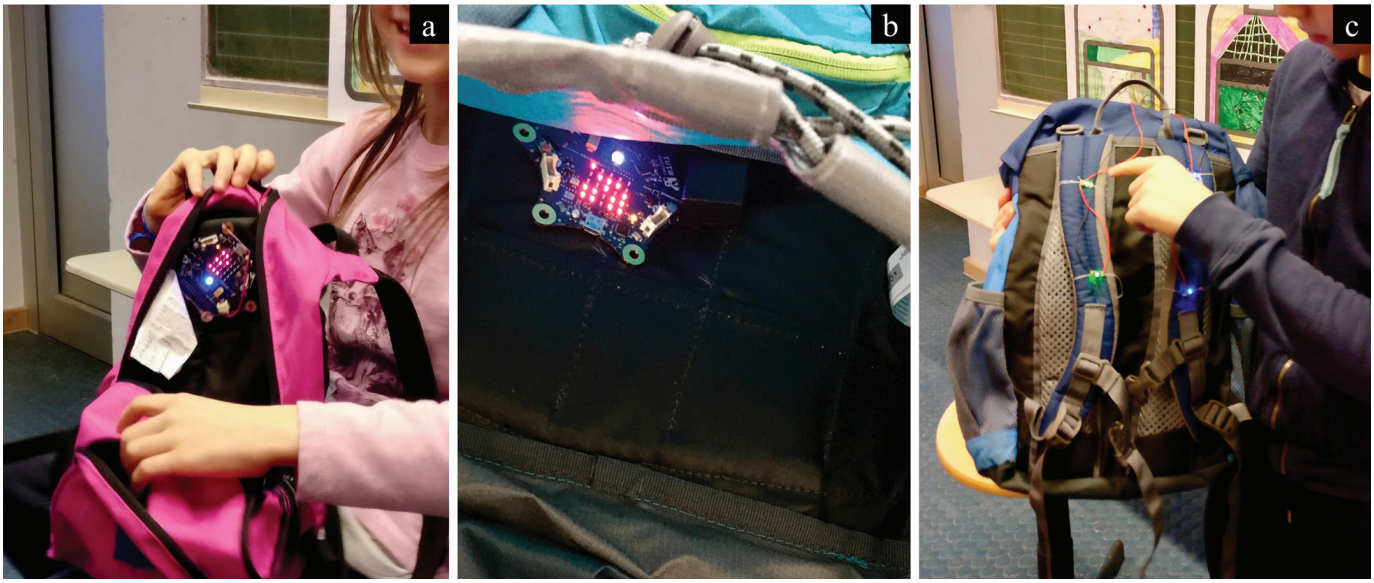

Figure 4. Backpack accessories: a) Calliope sewn inside a backpack, utilizing RGB LED and display b) Calliope used as a mini flashlight to illuminate a backpack's interior based on light sensor values c) backpack fitted with external LEDs controlled by the Calliope's buttons.

be that the Calliope does not integrate well into the types of crafting activities (e.g., drawing, coloring, gluing, cutting) to which they are accustomed. The Calliope in some ways stands distinctly apart from familiar materials. Consequently, they perceive it as something that needs to be affixed, attached, or accommodated for with a special pouch. It might be beneficial, therefore, to think about how these physical-computing platforms can be integrated into children's practices.

TOO MUCH COUPLING

Perhaps an alternative way of envisioning physical computing would be to consider how we can support the kinds of things kids are interested in making and where we currently fall short. For example, one of the girls from our workshop wanted to create a superhero eye mask with a string of blue, blinking LEDs highlighting the eyes. What seems like a simple task actually requires considerable work. The LEDs need to be stitched and connected with conductive thread, and the positive and ground leads need to route to the Calliope's ground and digital out connections. Since the Calliope cannot be accommodated on the eye mask due to its size, it has to reside elsewhere on the costume. To program this implementation, children have to go to the advanced menu in the blocks-based editor, find the "digital write" block, and set the pin and its associated value. To an 8-year-old child who wants to create a Halloween costume, this entire process (the construction and the programming terminology) is both confusing and complicated. Moreover, this particular design does not employ any of the Calliope's sensors.

Another issue lies with the coupling of sensors and feedback modalities on the same board. For one, children were unable to see the individual sensors on the board and conceptually grouped all the sensors together. But more important, designs that required sensors to be separate from the feedback modalities were simply not possible. For example, one child wanted to use the accelerometer to detect hand movement but have the LED display on the backpack. This would require two communicating Calliopes, which, while possible, is much harder to realize. Even if we disregard wearable designs, a child may want to monitor the temperature outside but have the LED display inside her room.

EASY TO PROGRAM BUT HARD TO UNDERSTAND

From a programming perspective, children described the Calliope as "very easy to program," but we found they had considerable trouble explaining their code or understanding how the system worked. The concept of a forever loop was missed by most children, and they often did not understand why one was required for polled actions such as button presses. We received questions such as: How long is a loop? What if I do it 1,000 times? How many seconds is 500 ms? What does it mean to "get ambient light value"? However, despite these questions, we did find children exploring and trying the various colorful blocks in different configurations. Moreover, conditionals such as if-then statements, when associated with specific components (e.g., on button A pressed), were well understood.

While some of the programming issues can be solved by age-appropriate wording in the blocks-based environment (e.g., "turn pin 1 on" or "give power to pin 1"),

Children described the Calliope as "very easy to program," but we found they had considerable trouble explaining their code or understanding how the system worked.

there still exist several cognitive disconnects in the programming process. After putting together their code, children had to connect the Calliope via USB and download the program. This process confused children throughout the entire workshop. Finding the tiny USB port was difficult and children often forgot to connect the Calliope. They also had trouble transferring the compiled file to the Calliope USB drive on the computer. This was particularly vexing when multiple compiled files from prior programs were already present in the downloads folder. Children also did not understand why the Calliope did not work when the code was downloading. Although there is a built-in Calliope simulator in the programming environment, it could not simulate the sensors, so children had trouble figuring out what the sensors were doing without downloading the program.

LACK OF VISIBILITY

The key point here is that children lacked visibility into what the Calliope was doing, and therefore had trouble debugging their code. Often, when kids encountered a problem, they hit a button, shook the device, observed what others were doing, or simply asked one of the adults. Many times, they did not even know that their code had errors. As one exasperated student put it, "Why can't you tear out the heart of the Calliope and examine it?" This statement was prompted by the fact that the transfer of code from the programming environment to the hardware was unidirectional. There is, for example, no way to look at the code on an already programmed Calliope by connecting it via USB. The child must remember to save the latest version of the program on a shared network resource. As one child remarked, "Maybe if you connect [the Calliope], the program should automatically show up. I don't know whether the program on the Calliope is already the right program."

A DISTRIBUTED APPROACH

One possible solution that addresses some of the problems discussed above would be to design a wireless kit of electronic blocks that were independent of each other and did not require a master microcontroller. Each sensor, LED, button, or motor would contain its own microcontroller, wireless adapter, and battery. These blocks could operate independently or communicate with each other wirelessly; therefore, a button on one side of the room could turn on a motor on the other side without a master microcontroller or wires in between.

Similarly, it should be quite possible to program the individual blocks wirelessly. This should greatly reduce the frustration kids often feel in waiting for the program to download. Moreover, the effects of the program cannot be seen in real time. This could be solved by switching to an interpreted language, where each independent module runs an interpreter that evaluates program statements in real time, wirelessly. Such a solution would not only facilitate debugging but also allow children to explore different parameters more quickly.

TOWARD PAPERT'S VISION

While children in our workshop had fun learning to use the Calliope, we still have some way to go in the realization of physical computing kits for children. Perhaps the real issue between Papert's vision and some of today's physical-computing platforms is philosophical. Platforms like the Calliope are really geared toward teaching through instruction rather than discovery. This is perhaps because of the current rhetoric around teaching children programming. There is an implicit urgency that betrays the economic necessity for more computer scientists. Contrast this with Papert's vision in *Mindstorms* [3], where programming is seen as an aid to a child's self-knowledge and an understanding of his or her own mind. Competence in programming is almost a healthy side effect of learning to solve problems and expressing ideas through language (a programming language just happens to be one of many).

Admittedly, the ideas presented here are not particularly new. The design approach we advocate hearkens back to older themes that focus on *selective exposure* of complexity [4] dependent on the developmental considerations of children. This exposes children to powerful ideas because less time is spent just trying to make things work. As Yasmin Kafai and Mitchel Resnick have argued, "Children don't get the ideas; they make ideas" [5]. And though learning can be joyful to many children, a lot of physical-computing educational activities do not necessarily facilitate the understanding required for learning the subject matter.

ACKNOWLEDGMENTS

We owe a big thank you to Torben Wallbaum, Meret Lindanis, and Erika Root for helping with the user study and crystalizing many of the issues children were having. We would also like to thank the late Michael Eisenberg for the invaluable discussions that led to these reflections.

ENDNOTES

1. Resnick, M. and Ocko, S. LEGO/Logo: Learning through and about design. In *Constructionism*. I. Harel and S. Papert, eds. Ablex Publishing, Norwood, NJ, 1991.
2. Harel, I. and Papert, S. *Constructionism: Research Reports and Essays, 1985-1990.* Ablex Publishing, Norwood, NJ, 1991.
3. Papert, S. *Mindstorms: Children, Computers, and Powerful Ideas.* Basic Books, New York, NY, 1980.
4. Blikstein, P. Gears of our childhood: Constructionist toolkits, robotics, and physical computing, past and future. *Proc. of the 12th International Conference on Interaction Design and Children*. ACM, New York, 2013, 173–182.
5. Kafai, Y. and Resnick, M. *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*. Lawrence Erlbaum Assoc., Mahwah, NJ, 1996.

Swamy Ananthanarayan is a postdoctoral researcher in the Media Informatics and Multimedia Systems Group at the University of Oldenburg, Germany. His research focuses on designing and evaluating tangible interactive systems that have high social impact, particularly for children, older adults, and underserved populations.
→ s.ananthanarayan@uni-oldenburg.de

Susanne Boll is a professor of media informatics and multimedia systems in the Department of Computing Science at the University of Oldenburg, Germany. She heads the Interactive Systems Group, which, among other topics, focuses on ambient, mobile, and tangible interfaces for children and older adults.
→ susanne.boll@uni-oldenburg.de

DOI: 10.1145/3386235