LinkBricks: A Construction Kit for Intuitively Creating and Programming Interactive Robots

Jiasi Gao[1,2,3*] Meng Wang[1,2,3*] Yaxin Zhu[2] Haipeng Mi[1]

***Abstract*— This paper presents *LinkBricks*, a creative construction kit for intuitively creating and programming interactive robots towards young children. Integrating building blocks, a hierarchical programming framework and a tablet application, this kit is proposed to maintain the low floor and wide walls for children who lack knowledge in conventional programming. The blocks have LEGO-compatible interlock structures and are embedded with various wireless sensors and actuators to create different interactive robots. The programming application is easy-to-use and provides heuristics to involve children in the creative activities. A preliminary evaluation is conducted and indicates that *LinkBricks* increases young children's engagement with, comfort with, and interest in working with interactive robots. Meanwhile, it has the potential of helping them to learn the concepts of programming and robots.**

I. INTRODUCTION

Computationally enhanced construction kits are becoming more and more popular in STEAM-related domains in the past decades. These kits are often composed of building blocks which can be assembled in a variety of ways to serve multi interactive manners[1]. Some kits also provide tools for children to program their constructions[2], [3]. Many prototypes, as well as extra curriculum, explored the constructive and creative learning[4], [5] using these new interactive tool kits. And lots of researches indicated such kits bring tremendous strengths for children, such as problem-solving[6], reflective thinking[7], imagination[8], etc.

Despite its predominant proliferation among computing education, the construction kits remain some limitations that hinder the broader use of population, especially for those young children who lack fundamental knowledge and practices in engineering and programming. As we have discovered, a majority of current tools target at primary and secondary school students. As for preschool children, traditional kits tend to be aesthetically and behaviorally limited[9]. The key issues come out about the high barriers in completing the tasks during the process of programming and assembly manipulation[10]. This motivates us to design a suitable kit for preschoolers, and our research questions are:

- How to make an appropriate abstraction of computing concepts to reduce programming barriers of young children.
- What are the proper design strategies of building blocks and programming environment to facilitate constructive tasks.

*Contributes equally
[1] Academy of Arts and Design, Tsinghua University, China.
[2] The Future Laboratory, Tsinghua University, China.
[3] Lab for Lifelong Learning, Tsinghua University, China. Emails: gaojs18@mails.tsinghua.edu.cn, mengwangthu@tsinghua.edu.cn, zhuyaxin1994@mails.tsinghua.edu.cn, mhp@tsinghua.edu.cn

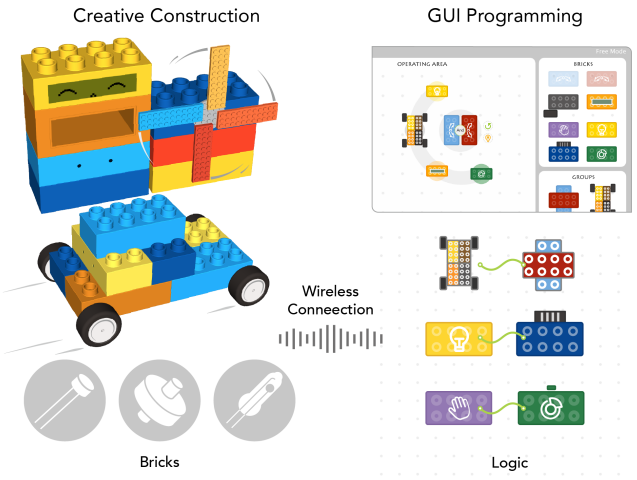

Fig. 1. *LinkBricks* overview. Children construct interactive robots using building bricks (left), and deploy them with a brick-based programming application (right).

A. Contribution

In this paper we propose a novel construction kit, *LinkBricks* (Figure 1), which provides LEGO-compatible computational blocks for physical construction and an easy-to-use programming application that reaches a sweet spot between structured and open-ended programming activities. Our contributions are summarized as follows:

- A set of bricks embedded with wireless sensors and actuators, enabling children to build a variety of interactive robots without the constraint of wires.
- An application based on a hierarchical programming framework, allowing young children to create complex behaviors by simple interactions.

B. Related Work

Construction kits that are designed for building blocks have a long history in education[9]. These kits emphasize the pedagogical concept like "learning through designing"[11] and "learning by doing[12]". In recent decades, these kits has blossomed in a variety of directions using new technologies, afford different means of construction manners[13]. Early works like Programmable bricks[14] and its extensions (LEGO Mindstorms, Cricket, etc.) permit children to

design and decorate their customized construction pieces. GO[15] and Phidgets[16] are those representative works that broaden participation and allow the children to explore advanced knowledge. Despite many styles and paths these kits supported, they are mainly designed for school-age children with basic knowledge. Young children aged form 4-6 encounter many challenges in building such blocks[6], [2], the complexity of connection operation and the lack of instruction[3] frustrate the kids. So there is a essential purpose to age-suitable block design[6] for fluent experience. The latest works intend to bring out new forms, new materials and new architectures, like Topobo[17] and roBlocks[18]. These works provide a set of wireless blocks for creating an interactive robots, which is simple enough to reach a low-floor usability to facilitate the constructive activities. Our work follows the principle of simplification for children to recognize the functionality of the blocks while maintain many paths to interact with them[19].

On the other hand, The question of how to teach young children programming is a classic research topic in the field of HCI. A notable early work is the LOGO language by Papert[11]. Nowadays children can use Scratch[20] to program the behavior of a series of sprites. These early works are usually designed to run programs virtually or with an sprite (such as the turtle of LOGO). Combined with the physical construction kits, these computational and engineering concepts exposed as tacit feedback. Nowadays, many construction kits provide visual blocks programming interface which diminish the difficulties of text and symbol-based languages for young children[21]. They provide natural language description of blocks and the drag-and-drop composition interaction, making the programming procedure easier[22], but are still too hard for young children to understand. Some other programming interfaces utilize tangible concepts[23], [24], [18] to present ideas of programming the interactive system. While these programming interfaces are easy to learn and use, they are usually capable of presenting very limited logic relationships. Much more attention should be paid to the programming interface and framework, and to find a proper abstraction of computing concepts. Our work intends to provide a novel programming environment for construction kits, facilitating preschool children to create various kinds of interactive robots.

C. Design Requirements

Despite lots of work have integrated new technologies into physical objects as educational toolkit for STEAM-related teaching, the majority of them are designed for school-age children. On the contrary, our target users are preschool children of 4-6 years old who may not have much basic knowledge of technology. As the research questions mentioned above, our goal is bringing simple and extensible kits to have them involved and interested in the constructive play while learning some basic concepts and knowledge about programming and robots. The design requirements are summarized as follows:

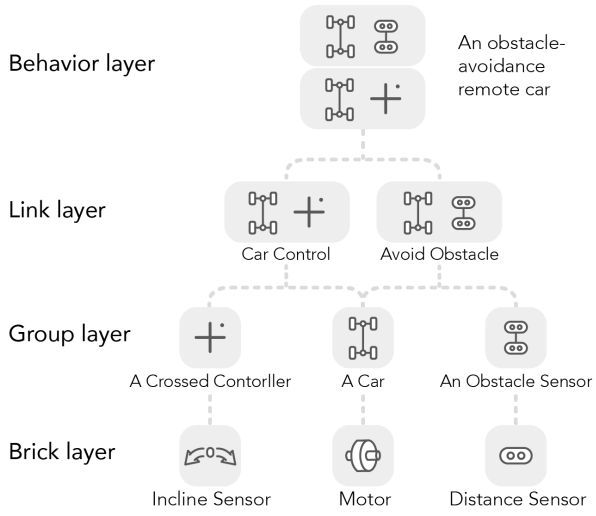

Fig. 2. The hierarchical programming framework. The four-layer design aid children to easily create complex behaviors for the interactive robot.

- Hide complex computing concepts while keep powerful ideas achievable.
- Keep the construction kits simple while remain possibilities for children to explore.
- Carefully design structured instructions to encourage early successes, and utilize scaffolding instructions to support exploration and tryout.

To achieve these design requirements, we will describe the design of programming framework, wireless constructive hardware, and GUI-based programming interface.

II. SYSTEM DESIGN

A. Programming Framework

The basic computing principles emphasize the procedural, abstraction and modularity thinking. When programming with a reconfigurable robot, it is also essential to understand how different physical parts can work together through digital data. The proposed hierarchical programming framework inherits a progressive layered structure[25] to leverage these principles while hiding some computing concepts to simplify the programming practice. It contains four layers (Figure 2): brick layer, group layer, link layer, and behavior layer.

Brick layer provides fundamental representations of physical bricks. Each brick has unique functions and properties, like the instance in the object-oriented programming language. We grip the functions and hide the detailed programming concepts like variables, values or data structures that may frustrate young children at the beginning.

Group layer represents a combination of physical bricks. A group may contain two or more bricks, usually the ones who have similar functions. According to the type of bricks included, a group may provide unique properties or actions. For example, if two incline (1D rotation) sensors are assembled as a cross in a group, it can provide a 2D rotation value to imitate a crossed controller. If four wheels are included in a group, it can provide direct action such as ”go forward” for execution. A group may contain actuators

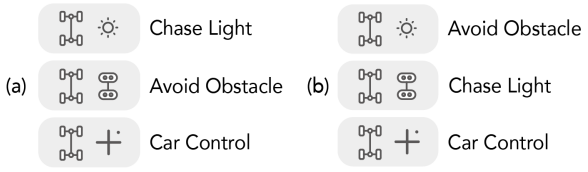

Fig. 3. Examples of behavior stacks. Reordering the priorities will lead to different "personalities" such as (a) bump against the light source when chasing, or (b) stop before the light source when chasing.

or sensors, but it can not contain both an actuator and a sensor simultaneously.

Link layer allows children to connect bricks/groups to create a control flow. We follow the component-based programming pattern and hide the complex logical statements. Children can easily create simple control logic (e.g. switch on/off) as well as complex ones (e.g. direction control of a car) by associating different groups. Each link usually connects an input group (with sensors) to an output group (with actuators). However, if more than one link is defined for the same group, conflicts may arise. Thus, we need higher-level management of these conflicts.

Behavior layer allows children to create multiple behaviors, or even present complex behavior logic while link layer defining simple acts. We adopt a simplified subsumption architecture [26] to define the logic of complex behavior. Links with the same output group are sorted in a stack. The order of each link indicates its execution priority. The link put on the top has the highest priority. Figure 3 shows examples of complex behavior stacks. A "car control" and an "avoid obstacle" behavior can create an obstacle-avoidance remote car since the "avoid obstacle" has a higher priority. If another most top priority behavior of "chase light" (a group of two light sensors controls the direction of the car) is added, the car will run to chase a glowing object but avoid other obstacles (Figure 3.a). However, this will lead to a collision against the glowing object. If the priorities of "chase light" and "avoid obstacles" are swapped (Figure 3.b), the car will stop before the glowing object.

B. Hardware

LinkBricks can be divided into three categories: sensor bricks, actuator bricks, and accessory bricks. Each brick is embedded with a sensor/actuator, a rechargeable battery, and a Zigbee wireless module to transmit data to a tablet. When bricks are disconnected from network, they enter sleeping mode and offer more than two week's standby time.

Sensor bricks are embedded with sensors to detect physical interaction and environment changing (Figure 4.left). The sensors could be a button, a knob, an infrared reflection sensor, a motion sensor for tapping, or an incline sensor for the detection of rotation, etc. Unexposed sensors are instructed by stickers outside.

Actuator bricks are embedded with actuators to create physical output according to changes in digital information (Figure 4.middle). A DC motor is embedded into a brick, which enables movement of a wheel or other connected parts.

Fig. 4. Sensor bricks (left): tap sensor, incline sensor, button; Actuator bricks (middle): DC motor, wheel, LED; Accessory bricks (right): cephalic horn, bug antennae.

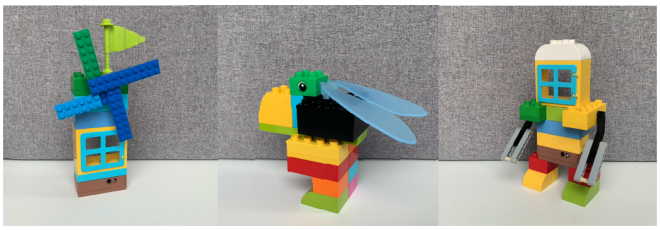

Fig. 5. Examples of interactive robots built by LEGO Duplo and LinkBricks.

Other actuators like LED, speaker and mini screen is also included.

Accessory bricks aim to augment the characteristics of a construction (Figure 4.right). The outer appearance of an accessory brick is designed to fit a particular kind of figure, and the embedded sensor/actuator also provides special corresponding functions. For example, if a kid wants to build a robot insect, s/he may need a pair of antennae, a huge mouth, a couple of wings, or a long cephalic horn. We have designed a number of accessory bricks to provide such features. A movable cephalic horn enables a rhinoceros beetle to raise other bricks. A pair of spring switches extend from an antennae brick, providing not only the appearance but also the touch sensor's function.

Besides, all bricks are compatible with LEGO Duplo bricks. Children can use normal LEGO bricks and LinkBricks together to build various of things, including interactive robots, architectures and devices (Figure 5).

C. Programming Application

With the proposed framework, we develop a GUI-based tablet application to program the physical kits. Two different interaction modes are provided in the application: Guided Mode (for novices with structured guidance), and Free Mode (provides more creativity for children). Considering children's attention shifts between the physical and virtual construction, and their ability to understand terms in the interfaces, we also designed a series of linkage hint animation in the application.

Guided Mode aims to teach the children how to use the programming application with bricks to create something that can move. Tasks of different topics are designed to familiarize them with the whole system, e.g., completing a

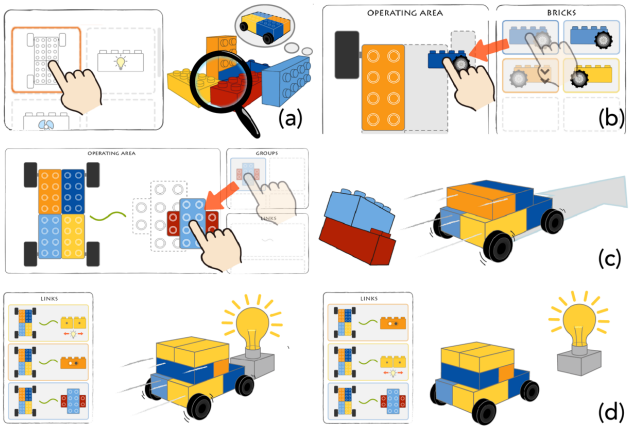

Fig. 6. Interaction process in Guided Mode

.

remote-control car step by step, or learning how to control a robot bee with a controller and a photosensitive brick. The programming application contains four major interactive areas: operating area, brick area, group area, and link area. All the physical bricks have their corresponding virtual versions in the brick area, which would appear automatically as the kit turns on. Children could follow the animation hint to drag a virtual brick to the operating area to compose a group or build a link. Animation effects are displayed to indicate the group and link operation, and icons would be updated in corresponding areas whenever certain association is made. Taking the remote-control car for an example, the interaction process could be described as below:

1) Assemble the physical bricks: The guided mode displays a car's 3D rendering model on the screen, and invite children to assemble the car using wheel-bricks. (Figure 6.a)

2) Make a virtual car group: After making the physical construction, children should find the wheel bricks' digital representation and then drag them together to make a car group (Figure 6.b). A template of the car is automatically provided as a hint.

3) Add a control link: Similarly, children are asked to make a crossed controller group, utilizing two incline sensor bricks in both physical and digital environments. Then, they are hinted to slide on the screen from the side of the controller to the car to generate a link. Thus, they can use the physical controller to control the car's movement (Figure 6.c).

4) Management of behavior: Children can create multiple links with different sensors. Each link represents a kind of action, and children can drag to change the link's priority (Figure 6.d). As discussed in the programming framework, this will lead to behavior changes. With simple drags and compares, children can gradually understand the rule between behavior changes and priority shifts in iterative attempts.

Free Mode includes all features of Guided Mode, meanwhile provides more openness and features for kids. In Free Mode, children do not have to follow a specific template. The application will show potential combinations templates

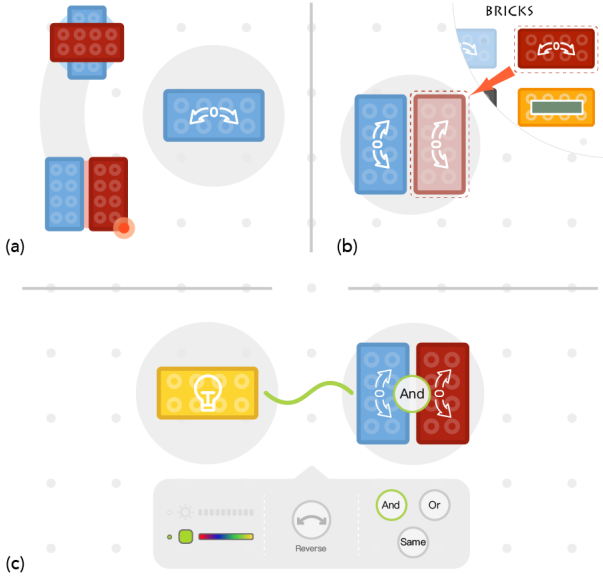

Fig. 7. Example of programming in Free Mode.

after a virtual brick being dragged to the operating area. In addition, more abstract group templates and links are available, e.g. a logic group (provides basic logic operation between sensors) and a control link (a mapping applicable between any sensors and actuators to define how they interact with each other).

Assuming a kid has successfully controlled a LED by an incline sensor, and then s/he wants to play together with her/his friend. An example in Figure 7 shows how they can control a LED brick by using two incline sensor bricks. First, possibilities of building a group with an incline sensor brick are presented after double-clicking (Figure 7.a). Two options are graphically instructed: a cross-controller group, and a group of separated sensors. Second, corresponding bricks (red incline sensor brick, and other sensor bricks if any) will flicker in the brick area as a visual cue, and then the kid can notice and drag the new brick to the existing brick (Figure 7.b) to create a new group. Third, a control link is created by the kid, and can be adjusted in a few ways (Figure 7.c). The kid can define the logic of the controller between "And", "Or" and "Same". For example, when "Same" logic is chosen the LED brick will only be activated if both incline sensor bricks are turned to the same direction. The kid can also choose to control either the brightness or the color of the LED. The "Reverse" button helps kid to easily switch the direction of control logic, e.g., turn left/right to control the LED on/off.

Overall, Free Mode provides less structured and more scaffolding instructions and functions. Children are free to try combining different virtual bricks, or adjusting link functions and link order. Whenever children make any changes, real-time feedback can be seen on the physical bricks. This provides heuristics to involve children in the creative activities.

Fig. 8. Children in the user study.

III. User Study

In this study, the independent variables (IV) were *LinkBricks*. The dependent variables (DV) were selected to examine our design goal of bringing easy-to-use kits for children to involve in the constructive activities and knowledge learning. The DVs were divided into three aspects, i.e. (1) Usability and engagement of *LinkBricks*, (2) Attitudes towards programming and engineering.

A. Study Procedure

We have invited twenty children aged from 4-7 to take the experiment separately. The study was conducted in a classroom of the kindergarten. We set up two video recorders to capture children's behaviors and voice. The study included three stages: pre-survey and teaching stage, test stage, and interview stage.

The first stage: Pre-survey and teaching. The child first makes the self-reporting about his attitudes toward playful toolkits. Then s/he is provided with four wheel bricks and two incline bricks, and required to complete the example task step-by-step with additional help from our researchers.

The second stage: test. After finishing the car combination and controlling its movements in Guided Mode, we introduce the child into Free Mode and ask s/he to complete a construct task to light a LED brick using a button brick. We record the task time and allow the children to make the interactive combination with their own expression.

The third stage: interview. After completing the whole construction process, the child is asked to complete a simplified version of usability questionnaire[27] and post-reporting of attitudes. Then we conduct an interview to get meaningful comments from the kids.

B. Results

1) Usability and engagement of LinkBricks: The simplified questionnaire mainly assesses the measurement of usability and experience with *LinkBricks*. The questionnaire is based on a 5-point semantic differential scale with several positive and negative performance of the *LinkBricks*. (i.e., -2: complicated, +2:simple).As Figure 9 shows, Q1-Q3 evaluated practical qualities (average score: 1.33 STD= 1.03) which revealed that the kids saw *LinkBricks* as simple, clear and easy-to-learn. Moreover, we recorded the time duration of Guide Mode (The average time: 5.34min STD= 1.29) and LED tasks in Free Mode(The average time: 4.35min STD= 1.31). There has been a decrease of time after initial learning,

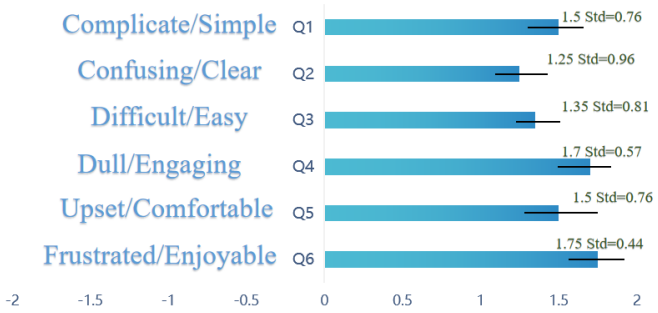

Fig. 9. The assessment results of usability and experience. Q1-Q3 related to the practical qualities and Q4-Q6 to affective qualities

which also reveals that LinkBricks is usable by learning-by-doing process. Q4-Q6 evaluated affective qualities (average score :1.41 STD = 1.01) which reflected children's overall engagement, comfortable and enjoyment during the construction process. The whole construction process is filled with positive emotion, especially when the children finish the tasks and see physical feedback. Many kids expressed their excitement and surprise when they get the car moving, i.e., *'Wow, It was so cool!'*, *'Oh,I make it!'*. During the car building process, we noticed that one girl would rather decorate her car with iced-cream shaped bricks instead of trying to make it move. After the researcher described a scene of selling ice cream, she started constructing the controller and completed the program smoothly.

2) Attitudes towards programming and engineering: When examining pre-investigation and post-survey for kids who report their attitudes towards physical programming, we saw increased learning interest and technological self-efficacy. In post-survey, 90% of participants said they want to play again and learn more about how *LinkBricks* works. Before they use our system, only 35% of participants agreed that they enjoy making the construction kits on their own. Some kids said that they have played physical LEGO bricks before, but it was the first time they can make them alive from ordinary toys. Such computational enhanced experience is so impressed for kids that really inspires them to continue to explore.

IV. Discussion and Conclusion

A. Discussion

The "ice-creamed" car inspired us to set more possibilities for children to reach this kit, e.g., like LEGO Friends series which attracts a lot of girls. Diversified scenes and themes would encourage more children to use engineering and computing knowledge as a tool to explore.

Based on participants' behavior, we found that the ability of conversion between 3D and 2D is developing in the age group 4-6, which means the interface design should be more explicit for target users and also points out that we should consider different cognitive levels in our design principles.

We also observed the artifacts made in the Free Mode in the second stage of the experiment. Exploration through repeated combination operations, we find many children realize the functionality of sensors and actuators as 9 of

them made a association between a button and a LED brick. Encouraging children's imagination and creativity is the most important aspect of creative learning while helping them learn the STEM-related knowledge is also a profound concern. In the future, we will develop more *LinkBricks* modules and templates to expand its construction and programming possibilities. For the assessment of knowledge acquisition, we will conduct a systematic artifacts analysis method [27] to build a coding scheme for the interactive robots using the key concepts from *Computing* and *Engineering*.

B. Conclusion

In this paper we propose a novel construction kit, *LinkBricks*, which provides LEGO-compatible computational blocks for physical construction and an easy-to-use programming application to low the barrier for preschool children. We conducted preliminary evaluation to assess the usability and engagement of our toolkit, and the results show that *LinkBricks* is easy-to-use and children can create different robots while learn STEM-related knowledge.

ACKNOWLEDGEMENT

This work is supported by National Key Research and Development Plan of China under Grant No. 2016YFB1001402, Tsinghua Shuimu Scholarship, and Tsinghua University Initiative Scientific Research Program (20197010003).

REFERENCES

[1] M. G. Petersen, M. K. Rasmussen, and K. B. Jakobsen, "Framing open-ended and constructive play with emerging interactive materials," in *the 14th International Conference*, 2015.

[2] H. Senaratne, P. Gunatilaka, U. Gunaratna, Y. Vithana, C. de Silva, and P. Fernando, "Sifeb–a simple, interactive and extensible robot playmate for kids," in *2014 4th International Conference on Artificial Intelligence with Applications in Engineering and Technology*. IEEE, 2014, pp. 143–148.

[3] A. Sipitakiat and N. Nusen, "Robo-blocks: designing debugging abilities in a tangible programming system for early primary school children," in *Proceedings of the 11th International Conference on Interaction Design and Children*, 2012, pp. 98–105.

[4] K. B. Jakobsen, J. Stougaard, and M. G. Petersen, "Expressivity in open-ended constructive play: Building and playing musical lego instruments," in *International Conference*, 2016.

[5] M. Przybylla and R. Romeike, "Physical computing and its scope - towards a constructionist computer science curriculum with physical computing," *Informatics in Education*, vol. 13, no. 2, pp. 241–254, 2014.

[6] C. Vandevelde, J. Saldien, C. Ciocci, and B. Vanderborght, "Overview of technologies for building robots in the classroom," pp. 122–130, 2013.

[7] M. S. Horn, E. T. Solovey, and R. J. K. Jacob, "Tangible programming and informal science learning: making tuis work for museums," in *International Conference on Interaction Design Children*, 2008.

[8] K. Brennan, M. Chung, and J. Hawson, "Creative computing: A design-based introduction to computational thinking," *Retrieved May*, vol. 9, p. 2012, 2011.

[9] M. Eisenberg, A. Eisenberg, M. Gross, K. Kaowthumrong, N. Lee, and W. Lovett, "Computationally-enhanced construction kits for children: Prototype and principles," in *Proceedings of the Fifth International Conference of the Learning Sciences*, 2002, pp. 23–26.

[10] T. Booth, S. Stumpf, J. Bird, and S. Jones, "Crossed wires: Investigating the problems of end-user developers in a physical computing task," in *the 2016 CHI Conference*, 2016.

[11] S. A. Papert, *Mindstorms: Children, Computers, And Powerful Ideas*. Basic Books, 1993.

[12] S. H. Yoon, A. Verma, K. Peppler, and K. Ramani, "Handimate: exploring a modular robotics kit for animating crafted toys." in *International Conference on Interaction Design Children*, 2015.

[13] P. Blikstein, "Gears of our childhood: Constructionist toolkits, robotics, and physical computing, past and future," in *International Conference on Interaction Design Children*, 2013.

[14] M. Resnick, F. Martin, R. Sargent, and B. Silverman, "Programmable bricks: Toys to think with," *Ibm Systems Journal*, vol. 35, no. 3, pp. P.443–452, 1996.

[15] A. Sipitakiat, P. Blikstein, and D. P. Cavallo, "Gogo board: augmenting programmable bricks for economically challenged audiences," pp. 481–488, 2004.

[16] S. GREENBERG, "Phidgets : Easy development of physical interfaces through physical widgets," in *Proc of Acm Symposium on User Interface Software Technology*, 2001.

[17] H. S. Raffle, A. J. Parkes, and H. Ishii, "Topobo: a constructive assembly system with kinetic memory," in *Sigchi Conference on Human Factors in Computing Systems*, 2004.

[18] M. D. Gross, "roblocks: a robotic construction kit for mathematics and science education," in *International Conference on Multimodal Interfaces*, 2006.

[19] M. Resnick and B. Silverman, "Some reflections on designing construction kits for kids," pp. 117–122, 2005.

[20] E. Eastmond, "The scratch programming language and environment," *Acm Transactions on Computing Education*, vol. 10, no. 4, pp. 1–15.

[21] G. Revelle, O. Zuckerman, A. Druin, and M. Bolas, "Tangible user interfaces for children," in *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 2051–2052. [Online]. Available: https://doi.org/10.1145/1056808.1057095

[22] S. Fleck, C. Baraudon, J. Frey, T. Lainé, and M. Hachet, ""teegi's so cute!": assessing the pedagogical potential of an interactive tangible interface for schoolchildren," in *the 17th IDC ACM Conference*, 2018.

[23] F. Güldenpfennig, D. Dudo, and P. Purgathofer, "Toward thingy oriented programming: Recording marcos with tangibles," in *Tei 16: Tenth International Conference*, 2016.

[24] A. Bdeir and T. Ullrich, "Electronics as material: littlebits," in *International Conference on Tangible*, 2010.

[25] J. Gao, M. Wang, and Y. Du, "Intelligent building block construction and programming system based on progressive layered structure," *Jisuanji Fuzhu Sheji Yu Tuxingxue Xuebao/Journal of Computer-Aided Design and Computer Graphics*, vol. 32, no. 7, pp. 1171–1176, 2020.

[26] R. Brooks, "A robust layered control system for a mobile robot," *IEEE journal on robotics and automation*, vol. 2, no. 1, pp. 14–23, 1986.

[27] S. Fleck, C. Baraudon, J. Frey, T. Lainé, and M. Hachet, "" teegi's so cute!" assessing the pedagogical potential of an interactive tangible interface for schoolchildren," in *Proceedings of the 17th ACM Conference on Interaction Design and Children*, 2018, pp. 143–156.