Prototyping Products using Web-based AI Tools: Designing a Tangible Programming Environment with Children

Alpay Sabuncuoğlu
Koç University - Is Bank AI Center
Istanbul, Turkey
asabuncuoglu13@ku.edu.tr

T. Metin Sezgin
Koç University - Is Bank AI Center
Istanbul, Turkey
mtsezgin@ku.edu.tr

ABSTRACT

A wide variety of children's products such as mobile apps, toys, and assistant systems now have integrated smart features. Designing such AI-powered products with children, the users, is essential. Using high-fidelity prototypes can be a means to reveal children's needs and behaviors with AI-powered systems. Yet, a prototype that can show unpredictable features similar to the final AI-powered product can be expensive. A more manageable and inexpensive solution is using web-based AI prototyping tools such as Teachable Machine. In this work, we developed a Teachable Machine-powered game-development environment to inform our tangible programming environment's design decisions. Using this kind of an AI-powered high-fidelity prototype in the research process allowed us to observe children in a very similar setting to our final AI-powered product and extract design considerations. This paper reports our experience of prototyping AI-powered solutions with children and shares our design considerations for children's self-made tangible representations.

CCS CONCEPTS

• **Social and professional topics** → **K-12 education**; • **Applied computing** → *Education*;

KEYWORDS

co-design with children, tangible programming, co-creation with AI, child-AI interaction

ACM Reference Format:
Alpay Sabuncuoğlu and T. Metin Sezgin. 2022. Prototyping Products using Web-based AI Tools: Designing a Tangible Programming Environment with Children. In *6th FabLearn Europe / MakeEd Conference 2022 (FabLearn Europe / MakeEd 2022), May 30–31, 2022, Copenhagen, Denmark.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3535227.3535239

We released all the activity materials open-source at https://github.com/asabuncuoglu13/karton-games

1 INTRODUCTION

Designing products for children requires a detailed design process to support their needs [9]. Involving children as more decisive partners in the design process reveals their behaviors and enables them to voice their opinions in shaping designs [16]. Nevertheless, developing a setup for children's involvement in the design process is not an easy task. Especially when the product in question is powered by machine learning techniques; these products do not follow a linear workflow and can behave unpredictably. Thus, researchers adopt new ways of prototyping these smart features with children [3].

This paper presents a case study on our paper-programming environment in that we utilized a web-based AI development environment, Teachable Machine, to get insights for our final product. A paper programming environment uses code blocks that are printed on paper cards which can be processed using mobile cameras and shows the code output on a digital screen. In our final product, we aimed to improve the existing paper programming environments [6, 8, 14] by representing the code scopes using self-made tangible objects. To use these tangible representations in their program, children should take a few photos of these models to save them in program memory. However, designing this setup requires careful consideration of the children's behavior. The mobile interface design should consider the possible interactions when the AI model cannot recognize the object.

Following a design procedure like a wizard-of-oz (WoZ) prototyping [11] can be an affordable solution before developing a complete AI-powered mobile solution. WoZ setups proved their success in a variety of cases [13], yet mimicking an AI system via this setup could bring researcher bias and focus on cases that we would think of as a problem. To understand child-AI interaction and utilize a prototyping method that could bring minimum researcher bias, we developed a setup that children can use Teachable Machine [1] to create their own self-made tangibles to code the Flappy Bird [17] game. This prototype follows a similar flow to our final product, but it is much easier to develop and test and allows us to conduct our research more rapidly.

In addition, using an AI development tool to co-create a prototype encourages children to learn the underlying principles of working with data. Considering this additional benefit, we also developed some educational material for teachers and researchers to integrate these tools into their research easily. The rest of the paper describes the details of the design process, reports our experience using a high-fidelity AI prototype, and shares our curated design considerations. Researchers can use the design flow and considerations to support their decision-making in an affordable fashion in an AI-powered children's product development process.

2 THE TOOLBOX

We named this section "Toolbox" as we introduced the tools and techniques we utilized in the studies.

2.1 Web-based AI Prototyping Tools

Web-based AI prototyping/development tools *(e.g., Teachable Machine, Dialogflow, and Edge Impulse)* allow users to create near state-of-the-art results when they are trained with enough-diverse data on specific tasks such as image recognition. Users can easily introduce new data and share the model link only using web interfaces.

Teachable Machine is a web-based tool that enables the easy creation of machine learning models such as image, sound, and pose classification [1]. They can either upload their offline data or collect real-time data online. After training, this platform allows users to store their models on their servers. Users can use this link in their Javascript codes and easily combine it with p5.js and https://ml5js.org/ml5.js frameworks to create quick prototypes. Their training mechanism runs on-device, which makes it more safe and secure than cloud-based training platforms. In this paper, we presented a case study using Teachable Machine, but two other platforms are worth mentioning as they can benefit other tasks. *Dialogflow* is an AI-based dialog creation platform where researchers can easily create their own chatbots. In the training part, users can feed a variety of question-answer pairs and create a flow of possible dialogs to shape the intended conversation. Another tool is *Edge Impulse*, a data collection and training interface that can benefit researchers who use sensor data and need low-resource ML applications.

2.2 Designing Products with Children

Researchers use different methods from low- to high-fidelity prototyping techniques to test their ideas, gather children's input and desires before developing complete solutions. Sketching the frames, paper-prototyping, and similar low-budget prototyping techniques have become standard procedures for application development. If the development process is expensive and needs unconventional methods to prototype, researchers use Wizard of Oz (WoZ) Prototyping [11]. It is a method that allows researchers to simulate the behavior of the end result using affordable tools and technologies. In this system, a human "wizard" controls the system's intelligence and simulates the interaction using a mock interface. From sketching to WoZ, all these techniques help designers choose human-centered decisions, but they are still open to designer bias since the designer is the manager of the whole process. Using easy-to-use AI development tools together with children can improve the product design process by reducing the designer bias and help hearing children's voices, behavior, needs, and habits.

Children can adopt different roles in the design process. Druin defines four roles that children can partake in: user, tester, informant, and design partner. In the user case, children use the technology, while adults may observe, record, or test for developed skills [4]. Researchers use this role to understand the impact of existing technologies have on child users to shape future technologies and enhance usability before production. In the tester role, children test prototypes of technology that have not been released to the world. As a tester, children are again observed with the technology and asked for their direct comments concerning their experiences. These results change the development steps of future iterations of technology. In the informant role, children play a part in the design process at various stages, based on when researchers believe children can inform the design process. Finally, in the design partner role, children may be observed with existing technologies or asked for input on design sketches or low-tech prototypes. Once the technology is developed, children may again offer design ideas and feedback. With the role of design partner, children are considered equal stakeholders in the design of new technologies throughout the entire experience.

2.3 Paper Programming Environments

Paper programming environments use card-based tangible programming blocks to create a code. Mostly a separate scanner or a mobile device is utilized to interpret the recognized programming blocks [7, 14]. These environments show the educational advantages of physical computing while using inexpensive materials as programming blocks. Parallel with the benefits of using TUIs in learning, utilizing paper programming in educational activities keeps children mentally and physically active, supports natural group interactions, increases the visibility of outputs, naturally invites students to action, and increases accessibility by lowering the threshold for children's participation [10, 18]. Current paper programming environments [2, 6–8, 14] present a high-level tangible command-set that can be recognized using computer vision algorithms.

In this work, children are involved as informants throughout the design process using our high-fidelity Teachable-Machine-powered game development interface to inform the development of a paper-programming environment.

3 INTRODUCE CHILD-MADE TANGIBLES TO PROGRAMMING USING COMPUTER VISION

In the activity, participants created a new high-level programming interface with tangible blocks for the Flappy Bird Game. The study took one and a half hours. First, we introduced the study process and taught the basics of sequential program structure and using events in programming. Then, we decomposed the actions in the Flappy Bird game and introduced the ten programming commands that we can use to code the Flappy Bird game from scratch. For each command, they designed tangible representations using craft materials such as play-dough, wires, crayons, etc. Later, they used Teachable Machine to build the recognizer model. Finally, we used the model's shareable link in our game-development interface.

3.1 Participation Procedure

We obtained ethical approval for this study from our university's Committee of Human Research. We contacted the parents via the researcher's personal contacts and snowballing. The parents have informed the study details via Whatsapp and reported their consent via this channel. Due to restricted COVID-19 regulations, we could not conduct this activity collaboratively, but five children participated in these studies individually (one seven-year-old, three

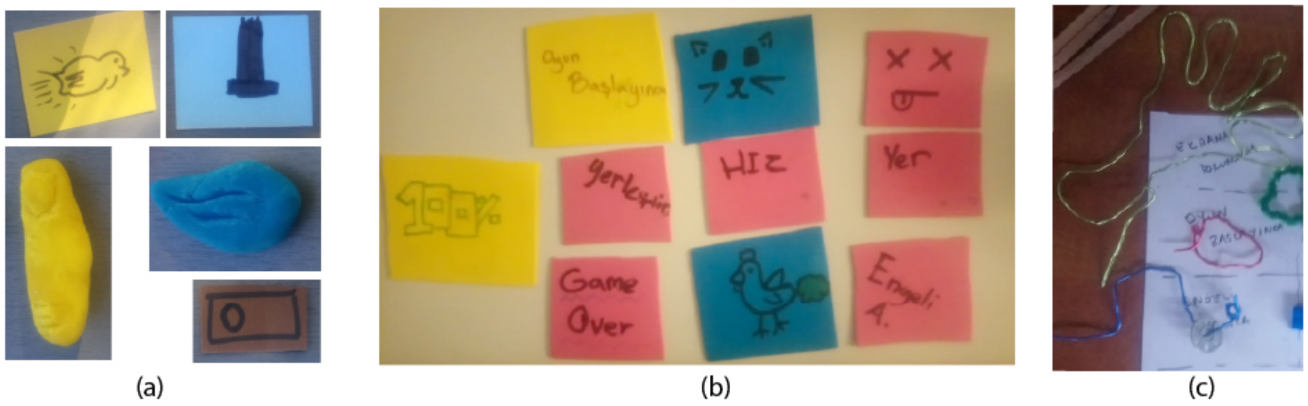


Figure 1: Example tangible representations of Flappy Bird game's programming commands. In the activity, they had fifteen minutes to build ten tangible representations. (a) This participant used cardboard and playdough, (b) She drew the actions/objects on different colored cardboard (the color choice was random) (c) He used metal wires and pipe cleaners to represent objects.

ten-year-olds, and one eleven-year-old). The seven-year-old participant was involved after his parent's special request, as he was curious about programming and would like to participate in the study. These students are from Antalya and Istanbul, major cities in Turkey. Two students are from private schools (one seven-year-old, one ten-year-old). Children participants had a mixed understanding of what AI can or cannot do. They use smartphones/tablets daily and exhibit digital literacy skills in our conversations.

3.2 Coding the Game with Tangible Representations

In our game coding interface, children show their self-made tangibles to add events and commands in the correct order to code the game. Ten pre-defined functions allow children to create a Flappy Bird game from scratch. These commands are "When Touch Screen," "When Game Starts," "When Pass Obstacle," "When Hit Obstacle," "When Hit Ground," "Flap," "Change Score," "Set Speed," "Place Obstacle," and "End Game." We asked them to build new tangible representations for these predefined event and action commands. We also handed out a probe with a grid table showing all ten predefined functions to help children track their creations. Children had only fifteen minutes in this session, which led them to use much simpler materials like paper, pen, and play-dough, as seen in Figure 1

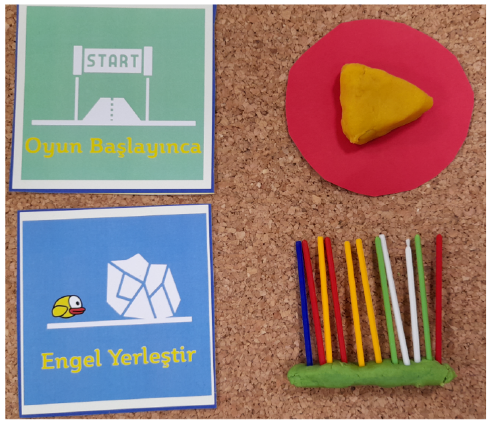
3.3 Using Teachable Machine

After creating the physical correspondences of these functions, children used the *Teachable Machine* as seen in Figure 2. Opening a new Image Project in Teachable Machine shows an intuitive interface where children encounter a traditional machine learning pipeline. The application interface serves a traditional machine learning pipeline with three steps: (1) Users collect data by either uploading images or capturing them in real-time via webcam, (2) Once the data is ready for all class labels, users can start training with pre-defined hyper-parameters. Experienced users can tweak the parameters to increase the speed or accuracy. (3) Finally, users can test the results in real-time. If the user is happy with the results, they can export the model using a single link. Alternatively, they can collect more samples or change the parameters and fine-tune their models to improve the classification accuracy.

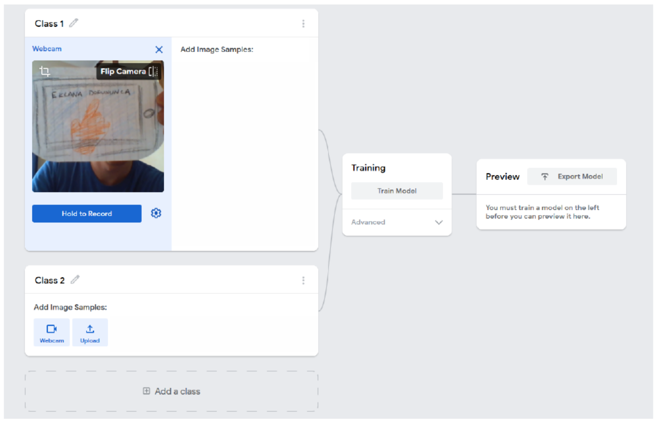
In all our studies, children either needed to update their model by adding new samples to the dataset or changing our tangible objects' design. Figure 1 shows sample tangible programming blocks that issue different challenges. (a) One child used yellow cardboard to sketch figures. When he created three visually similar tangibles, we asked him to test the model. As expected, the confidence score of test results was not reliable. Before explaining why the test resulted in this, he already made the deduction. Then, he changed the tangibles with different materials like play dough and gave them different shapes. (b) In this set, the child sketched or wrote some text on the cardboard. With an analogy to application buttons, the pink cardboards have some text on them. Teachable Machine can recognize different text when the training data is rich and sample images are close to the camera. When the model is trained this way, we always needed to show these cards very close to the camera and shoot many images from different angles, but it was not a problem for her. (c) This child picked metal wires in the tangible building phases. For example, the "When Touch Screen" command is represented by a wire hand. The problem is that metal wire exposes all the background, and the AI model mostly learns the background image features. Additionally, metal wires' visibility quickly changes in different light conditions. After seeing the confidence scores of the AI model, this child chose a different material.

When the AI model was not working as expected, we stopped and asked what can be the model's problem. They neither blame the system nor assume the system is broken. Some common guesses and suggestions were increasing the number of samples, using better lighting while taking photos, and using diverse samples that the system can easily differentiate. These suggestions indicate that children can use and improve AI systems with sufficient guidance.

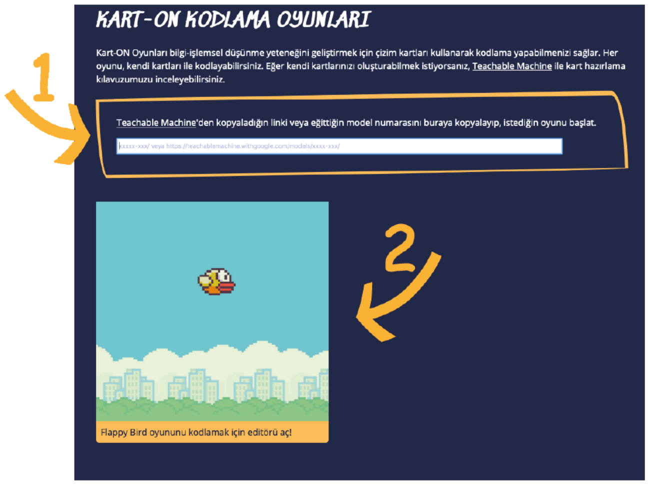
After the discussion, we guided them by adding more samples or updating the tangibles to separate visually similar objects better. Utilizing Teachable Machine in our design process allowed us to



(a)



(b)


KART-ON KODLAMA OYUNLARI

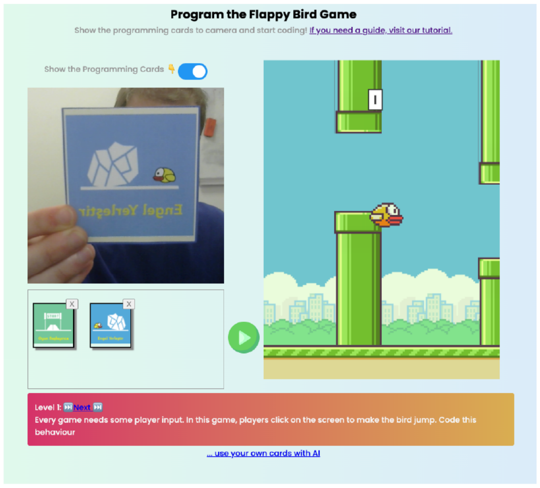
Kart-ON Oyunları bilgi-işlemsel düşünme yeteneğini geliştirmek için çizim kartları kullanarak kodlama yapabilmenizi sağlar. Her oyunu, kendi kartları ile kodlayabilirsiniz. Eğer kendi kartlarınızı oluşturabilmek istiyorsanız, Teachable Machine ile kart hazırlama kılavuzumuzu inceleyebilirsiniz.

Teachable Machine'den kopyaladığın linki veya eğittiğin model numarasını buraya kopyalayıp, istediğin oyunu başlat.

xxxxx-xxx/ veya https://teachablemachine.withgoogle.com/models/xxxx-xxx/

Flappy Bird oyununu kodlamak için editörü aç!

(c)


Program the Flappy Bird Game

Show the programming cards to camera and start coding! If you need a guide, visit our tutorial.

Level 1: ⏭Next ⏭
Every game needs some player input. In this game, players click on the screen to make the bird jump. Code this behaviour

... use your own cards with AI

(d)

Figure 2: (a) Children created tangible representations of pre-defined Flappy Bird game programming commands. (b) Then, they used the Teachable Machine Image Project Training Interface to create the computer vision model. Children collect data via webcam, hit training when the data collection is completed, and test the recognition accuracy. (b) After they complete the training process, they copy and paste the public link of the AI model. (c) Finally, they used their tangible programming blocks in our web coding interface. First, they show the tangible representation to the webcam. If the model recognizes the representation, it appears in the coding area. Then, they click the "play" button, which runs the game with the coded logic. For example, this image shows the output of two programming cards: "When the game Starts" - "Put obstacles," which results in a game interface with obstacles.

see the tangible programming objects in action, and children felt their impact on our design process.

4 DESIGN CONSIDERATIONS FOR SELF-MADE TANGIBLES

After conducting five sessions, our observations and analysis of tangible outputs led us to curate design considerations for building self-made tangibles. Although testing the setup with a small-scale group with participants in different age groups restricted the generalizability, our observations and analysis led us to develop early design considerations.

4.1 Observations and Analysis

In the analysis, we used our notes and tangible output photos. We thematically categorized our observations into two categories: improving the vision model and interaction with the AI system.

- **Improving the vision model:** Children created objects from wires and other transparent materials. The computer

vision model could not differentiate these objects as using these materials shows a larger portion of the background. So, this type of material selection that increases the transparency reduces the accuracy of the vision model.
If children use the vision model in different backgrounds, or if the light conditions of the setup change rapidly, taking photos in different backgrounds or adding data augmentation by applying a segmentation beforehand is required.
Considering different environmental conditions is important as environment lighting in the case of computer vision, and noise in speech systems determines the accuracy.

- **Interaction with AI system:** Children above the age of seven (N=4) quickly learned the relation between model accuracy and the number of data samples. They easily grasped if the model needs more data to improve the model, yet they need an intuitive method of adding more data to the right label.
Conducting the activity with a seven-year-old participant allowed us to get a glimpse into younger age groups' use of the Teachable Machine for prototyping. During the study, he needed constant support from the main researcher, but eventually was able to introduce new tangible representations for a new class.

4.2 Design Considerations

The prototyping process allowed us to learn the challenges before developing our computer vision model quickly. Based on user patterns and concerns, we can share the following considerations for educators and researchers:

(1) **Combine materials:** Combination of materials enhances the interpretability of programming objects from the child's perspective. Using combinations of different textures and colors also increases the accuracy of the AI model's recognition.
(2) **Add details with 'simple' shapes:** Adding details helped children to memorize the function of designed tangible and enhanced the AI model accuracy. Yet, we emphasized adding details with 'simple shapes' to use children's time more efficiently. In the tangible creation process, they tended to add details with 'fancy' materials and zigzagged scissor moves which distracted them from the tasks.
(3) **Use the materials that fit the environment:** Considering the materials for different locations is an important aspect of the quality of programming time. If the children plan to use the objects more than once, play-dough is not encouraged since it cracks after drying. Also, light conditions affect the model's accuracy significantly. Using only wires to represent programming objects result in a high dependency on the foreground-background clarity that are challenging in outdoor settings (e.g., many elements like vegetation). Therefore the environment should be considered when choosing the materials.

5 DISCUSSION AND CONCLUSION

This paper demonstrated an example of co-designing an interface with children using high-fidelity prototyping that shows similar behavior to the final product. Using the web-based AI training tool *Teachable Machine* allowed children to create their physical products in real-life scenarios. In traditional co-design activities, children or adult users mostly build paper prototypes or other low-fidelity design outcomes. Our studies demonstrated that using a functional development environment with novice users on an unfamiliar task increased the involvement of students in the design process and supported their iterative learning process.

During the study, we were children's more experienced activity partners rather than product owners. When children grasped the fundamental mechanisms behind the system, they became better at training the AI model. We encouraged children to test their model with different dataset sample sizes, lighting conditions, and using distinct shapes. Teachable Machine's step-by-step interface naturally guides children to collect data and test the results in a structured way. During the studies, we discussed why some test samples are recognized quickly, why some are recognized slow, and why the system could not understand some objects. Introducing the AI model's basic mechanism with this flow helped children grasp the system capabilities quickly and reason when they encountered misclassification and low test accuracy cases. These learning outcomes are important aspects of K-12 AI Education [15], and prototyping new AI-powered products can be an engaging process for AI education. Prototyping a product using an easy-to-use AI development platform can help educators to discuss the development process, ethics, and possible consequences of integrating AI into our daily lives [5]. We open-sourced all the activity guides and materials to encourage teachers and researchers to develop new application prototypes in the classrooms and share their processes. Educators can adapt our workflow in their curriculum to develop an AI system together with children.

Tools like Teachable Machine and other no-code platforms can accelerate co-design activities where designers and end-users can update the product in the design process with active participation. Similar creative development activities can foster awareness of new technology, make the co-design more fruitful, motivate children to explore possibilities, and encourage reflection in collaborative work [12]. We hope this work will open up the discussion on using AI tools to encourage designers to get children's voices and creativity in their product design.

ACKNOWLEDGMENTS

The authors gratefully acknowledge that this work was supported by TUBITAK [Grant Number 218K436] and Koç University-Is Bank AI Center.

REFERENCES

[1] Michelle Carney, Barron Webster, Irene Alvarado, Kyle Phillips, Noura Howell, Jordan Griffith, Jonas Jongejan, Amit Pitaru, and Alexander Chen. 2020. Teachable Machine: Approachable Web-Based Tool for Exploring Machine Learning Classification. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3334480.3382839
[2] Xiaozhou Deng, Danli Wang, Qiao Jin, and Fang Sun. 2019. ARCat: A Tangible Programming Tool for DFS Algorithm Teaching. In *Proceedings of the 18th ACM International Conference on Interaction Design and Children* (Boise, ID, USA) *(IDC '19)*. Association for Computing Machinery, New York, NY, USA, 533–537. https://doi.org/10.1145/3311927.3325308
[3] Stefania Druga. 2019. Co-designing inclusive and equitable intelligent systems with and for kids around the world. *Journal of Design and Science* (5 5 2019). https://jods.mitpress.mit.edu/pub/3yi7jnz9 https://jods.mitpress.mit.edu/pub/3yi7jnz9.

[4] Allison Druin. 2002. The role of children in the design of new technology. *Behaviour & Information Technology* 21, 1 (jan 2002), 1–25. https://doi.org/10.1080/01449290110108659

[5] d.school. 2019. Using Teachable Machine in the d.school classroom | by Michelle Carney | Medium. https://medium.com/@michellecarney/using-teachable-machine-in-the-d-school-classroom-96be1ba6a4f9

[6] Anna Fuste and Chris Schmandt. 2019. HyperCubes: A Playful Introduction to Computational Thinking in Augmented Reality. In *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts* (Barcelona, Spain) *(CHI PLAY '19 Extended Abstracts)*. Association for Computing Machinery, New York, NY, USA, 379–387. https://doi.org/10.1145/3341215.3356264

[7] Michael S Horn and Robert JK Jacob. 2007. Tangible programming in the classroom with tern. In *CHI'07 extended abstracts on Human factors in computing systems*. 1965–1970.

[8] Felix Hu, Ariel Zekelman, Michael Horn, and Frances Judd. 2015. Strawbies: Explorations in Tangible Programming. In *Proceedings of the 14th International Conference on Interaction Design and Children* (Boston, Massachusetts) *(IDC '15)*. Association for Computing Machinery, New York, NY, USA, 410–413. https://doi.org/10.1145/2771839.2771866

[9] Florence Kristin Lehnert, Jasmin Niess, Carine Lallemand, Panos Markopoulos, Antoine Fischbach, and Vincent Koenig. 2021. Child–Computer Interaction: From a systematic review towards an integrated understanding of interaction design methods for children. *International Journal of Child-Computer Interaction* xxxx (2021), 100398. https://doi.org/10.1016/j.ijcci.2021.100398

[10] Paul Marshall. 2007. Do Tangible Interfaces Enhance Learning?. 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, 163–170. https://doi.org/10.1145/1226969.1227004

[11] David Maulsby, Saul Greenberg, and Richard Mander. 1993. Prototyping an intelligent agent through Wizard of Oz. In *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*. 277–284.

[12] S. Papavlasopoulou, M. N. Giannakos, and L. Jaccheri. 2017. Reviewing the affordances of tangible programming languages: Implications for design and practice. In *2017 IEEE Global Engineering Education Conference (EDUCON)*. 1811–1816. https://doi.org/10.1109/EDUCON.2017.7943096

[13] Laurel D. Riek. 2012. Wizard of Oz Studies in HRI: A Systematic Review and New Reporting Guidelines. *J. Hum.-Robot Interact.* 1, 1 (jul 2012), 119–136. https://doi.org/10.5898/JHRI.1.1.Riek

[14] Alpay Sabuncuoğlu and Metin Sezgin. 2020. Kart-ON: Affordable Early Programming Education with Shared Smartphones and Easy-to-Find Materials. In *Proceedings of the 25th International Conference on Intelligent User Interfaces Companion*. 116–117.

[15] David Touretzky, Christina Gardner-McCune, Cynthia Breazeal, Fred Martin, and Deborah Seehorn. 2019. A Year in K-12 AI Education. *AI Magazine* 40, 4 (Dec. 2019), 88–90. https://doi.org/10.1609/aimag.v40i4.5289

[16] Maarten Van Mechelen, Mathieu Gielen, Vero vanden Abeele, Ann Laenen, and Bieke Zaman. 2014. Exploring Challenging Group Dynamics in Participatory Design with Children. In *Proceedings of the 2014 Conference on Interaction Design and Children* (Aarhus, Denmark) *(IDC '14)*. Association for Computing Machinery, New York, NY, USA, 269–272. https://doi.org/10.1145/2593968.2610469

[17] R Williams. 2014. What is flappy bird? the game taking the app store by storm. *The Daily Telegraph,[Online]. Available: https://www. telegraph. co. uk/technology/news/10604366/What-is-Flappy-Bird-The-game-taking-the-App-Store-by-storm. html.[Accessed 3 March 2019]* (2014).

[18] Oren Zuckerman, Saeed Arida, and Mitchel Resnick. 2005. Extending Tangible Interfaces for Education: Digital Montessori-inspired Manipulatives. *CHI'05 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2005), 859–868. https://doi.org/10.1145/1054972.1055093