Kart-ON: Affordable Early Programming Education with Shared Smartphones and Easy-to-Find Materials

Alpay Sabuncuoğlu
Koç University
asabuncuoglu13@ku.edu.tr

Metin Sezgin
Koç University
mtsezgin@ku.edu.tr

Figure 1: (1) Everyday materials can be used as programming objects. (2) These objects can be recognized by computer vision enabled smartphone application. (3) Recognized text and objects are translated to p5.js library commands for visual outputs.

ABSTRACT

Programming education has become an integral part of the primary school curriculum. However, most programming practices rely heavily on computers and electronics which causes inequalities across contexts with different socioeconomic levels. This demo introduces a new and convenient way of using tangibles for coding in classrooms. Our programming environment, Kart-ON, is designed as an affordable means to increase collaboration among students and decrease dependency on screen-based interfaces. Kart-ON is a tangible programming language that uses everyday objects such as paper, pen, fabrics as programming objects and employs a mobile phone as the compiler. Our preliminary studies with children ($n=16$, $\text{mage}=12$) show that Kart-ON boosts active and collaborative student participation in the tangible programming task, which is especially valuable in crowded classrooms with limited access to computational devices.

CCS CONCEPTS

• **Human-centered computing** → *Collaborative interaction; Interface design prototyping*; • **Social and professional topics** → *K-12 education*.

KEYWORDS

Affordable tangible programming, Collaborative classroom

ACM Reference Format:

Alpay Sabuncuoğlu and Metin Sezgin. 2020. Kart-ON: Affordable Early Programming Education with Shared Smartphones and Easy-to-Find Materials. In *25th International Conference on Intelligent User Interfaces Companion (IUI '20 Companion)*, March 17–20, 2020, Cagliari, Italy. ACM, New York, NY, USA, 2 pages. <https://doi.org/10.1145/3379336.3381472>

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

IUI '20 Companion, March 17–20, 2020, Cagliari, Italy

© 2020 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-7513-9/20/03.

<https://doi.org/10.1145/3379336.3381472>

1 EARLY PROGRAMMING EDUCATION

Programming education has been integrated into primary education in most developed countries and gradually accepted worldwide to support children in building computational and creative thinking skills. Computational thinking encompasses a set of problem solving techniques that have been used in computer science such as abstraction, decomposition, algorithmic thinking and pattern recognition. In recent years, block based visual languages gained popularity among teachers and students due to its error-free compilation and easy to use interface. Programming environments like Scratch¹ and Alice² have been developed to be visually programmed with drag and drop blocks without the need to learn the syntax. Additionally, programming environments like Tern³ or Google Bloks⁴ have been using tangible programming blocks in a graspable form. Research shows that these tangible platforms allow students to collaborate with each other and engages in active learning [1, 3].

However, even though tangible programming has many advantages, most of the tangible programming blocks are expensive due to cost of the electronic elements and the manufacturing materials. To that end, utilizing the recent developments in computer vision can reduce costs with the use of smart recognition of the blocks. This demo introduces an affordable system to complete all the tasks in a novice programming education with using everyday objects as tangible programming blocks, and a smartphone application to recognize these blocks.

2 PROGRAMMING WITH KART-ON

In a classroom setting with Kart-ON, groups of 3 or 4 students have physical cards to be used in solving the given task. The tasks can be drawing a snowman, moving an airplane or simulating the Earth's orbit. To do so, the children use the card blocks to design

¹<https://scratch.mit.edu>

²<https://www.alice.org/>

³<http://hci.cs.tufts.edu/tern/>

⁴<https://projectbloks.withgoogle.com/>

Figure 2: After creating a new function, students can associate any object to call this function. In this example, students use a tennis ball to save the ellipse-drawing program.

an algorithm. After they complete the algorithm for the task, they can request a smartphone or tablet to compile their program and see the digital output.

Kart-ON uses *p5.js*, a JavaScript library for creative coding, to draw shapes and animations. For example, students can animate a simple moving circle, with the following program text:

```
|create variable: c init: 0|- |background r: 255, g: 255, b: 255|-  
|ellipse x: c y: 20 w: 10 h: 10|- |increase the value of: c by: 1|
```

In this program "create variable" command executes once in the setup. As program runs in every frame, the rest of the program executes forever: The value of *c* will increase, background will be repainted again and the circle will move toward *x* axis by one pixel. We modified the commands and the structure based on the student needs and best practices from the field [2]. We developed the language in an iterative way informed through regularly meeting with K4-6 grade students.

New variables and objects can be defined in two possible ways as seen in Figure 2: (1) They can assign the function to an object (a LEGO brick, a toy or a sketch of the function) or (2) use text in cards to start and end the function definition. We used Tensorflow to learn new objects with transfer learning. With this approach, our system returns %80+ accuracy when 20+ images given for one object class.

To recognize the text on pre-defined cards, we include Firebase ML Vision to our app, which allows the app to recognize in real-time with good results. After understanding the text, we used a fuzzy-search library to understand the code when the text recognizer returns a text with smaller errors. Additionally, students can use play-dough, fabrics or any other object in their program to represent colors if they are not familiar with the RGB or HSL representation.

3 APPLICATIONS AND OBSERVATIONS

Using self-designed tangibles that can be recognized from a shared-smartphone is not only useful for drawing simple shapes and animations, but we can develop functional programs to solve data science problems. For example, Figure 3 shows an example application to map a range of football players data on a geographic map in Kart-ON language. The application uses the same techniques to understand cards and text and JavaScript libraries like *d3*, *c3* and *Leaflet*. We choose to define programming blocks in a higher level of abstraction, to make the programming easier and focus on teaching data-science topics [2].

We met with different student groups on four occasions to understand their needs, explore their way of thinking and observe their

Figure 3: An example data science program with this approach. It downloads the football player data, gets the *nationality* column, filters the values, adds filtered countries on a map

collaboration amongst themselves. In these two hour meetings, we also discussed applications of coding, drew different shapes with Kart-ON and learned to represent colors in digital. These meetings allowed us to know them better and get feedback about the product. We further interviewed teachers to get feedback on Kart-ON. From our meetings we obtained three main advantages of using this interaction from an educational perspective:

- (1) It reduced the screen time and increased the active discussion time. Students could hold and rearrange the cards while thinking about the task, and they are able to express their ideas with cards.
- (2) Students tended to think about their algorithm thoroughly before calling their teacher to execute the task via the phone. This prevented the trial and error behaviour to complete the task, commonly seen in screen-based coding interfaces.
- (3) After the session, students became more aware of the fact that everyday materials can be used to design programming tools and were eager to use their materials in an activity.

Our meetings in two different schools with four different groups and the teacher feedback encourage us to develop more applications with these cards and use this simple approach to support equal education. Further work will include a framework to create easy programs with this technique and apply the approach to several applications.

4 CONCLUSION

This demo shows a convenient approach to work with mobile phones and tangibles for programming education. We demonstrated developing an affordable programming environment that encompasses using tangibles and different computer vision applications such as understanding text and objects, and used them as programming elements. Our programming environment allows students and teachers to discuss the computational thinking related problems with only one smartphone in a crowded classroom. (<https://karton.ku.edu.tr/home>)

REFERENCES

- [1] A. N. Antle and A. F. Wise. 2013. Getting Down to Details: Using Theories of Cognition and Learning to Inform Tangible User Interface Design. *Interacting with Computers* 25, 1 (Jan 2013), 1–20. <https://doi.org/10.1093/iwc/iws007>
- [2] Mitchel Resnick and Brian Silverman. 2005. Some reflections on designing construction kits for kids. *Proceeding of the 2005 conference on Interaction design and children - IDC '05* (2005), 117–122. <https://doi.org/10.1145/1109540.1109556>
- [3] Oren Zuckerman, Saeed Arida, and Mitchel Resnick. 2005. Extending tangible interfaces for education. June 2014 (2005), 859. <https://doi.org/10.1145/1054972.1055093>