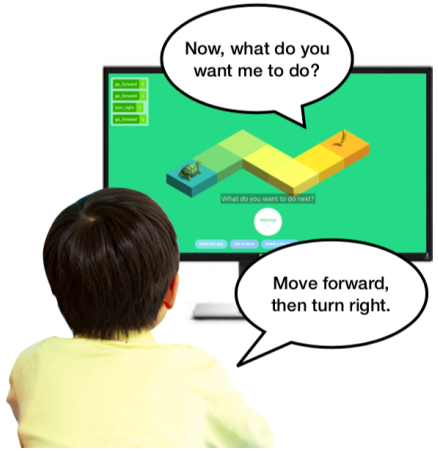


Figure 1: TurtleTalk, an educational programming game for children using VUI.

TurtleTalk: An Educational Programming Game for Children with Voice User Interface

Hyunhoon Jung
Naver Clova AI
hyunhoon.j@navercorp.com

Hee Jae Kim
Naver Clova AI
heejae.kim@navercorp.com

Seongeun So
Naver Clova AI
seongeun.so@navercorp.com

Jinjoong Kim
Naver Clova AI
jinjoong.kim@navercorp.com

Changhoon Oh
Seoul National University
yurial@snu.ac.kr

ABSTRACT

Interest in programming education for children is growing. This research explores the possibilities of utilizing voice user interface (VUI) in children's programming education. We designed an interactive educational programming game called TurtleTalk, which converts the various utterances of children into code using a neural network and displays the results on a screen (Figure 1). Through VUI, children can move the turtle, the voice agent of the game, to the target location and learn the basic programming concepts of "sequencing" and "iteration." We conducted a preliminary user study where eight children played the game and took part in a posthoc interview. The results showed that voice

CHI'19 Extended Abstracts, May 4–9, 2019, Glasgow, Scotland Uk

ACM ISBN 978-1-4503-5971-9/19/05.
https://doi.org/10.1145/3290607.3312773

interaction with TurtleTalk led children to be more immersed in the game and understand the elements of programming with ease and confidence.

CCS CONCEPTS

• **Human-centered computing → Human computer interaction (HCI)**.

KEYWORDS

Programming education; children; voice user interface

ACM Reference Format:
Hyunhoon Jung, Hee Jae Kim, Seongeun So, Jinjoong Kim, and Changhoon Oh. 2019. TurtleTalk: An Educational Programming Game for Children with Voice User Interface. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts (CHI'19 Extended Abstracts), May 4–9, 2019, Glasgow, Scotland Uk*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3290607.3312773

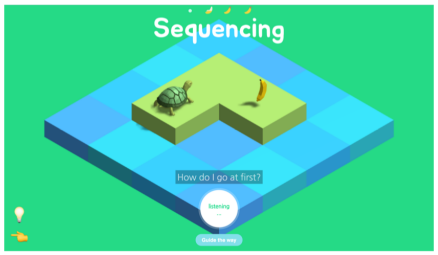
INTRODUCTION

As the importance of programming education for children has been widely recognized, many studies have been conducted to develop programming tools or environments that allow children to learn programming without being overwhelmed. With block-based visual effects, Scratch [8] and Blockly [3] help users understand the code intuitively [6]. Conversely, there have been attempts to make a natural-language-like programming environment, such as Clojure-Turtle [4], which uses a "logo" language in a Clojure context. Although these studies presented new perspectives on programming education for children, the former cases do not allow users to input the code quickly and the latter case still depends on the command line interface-based input method.

Meanwhile, the development of voice user interface (VUI), which has become popular recently, presents new possibilities for programming education for children. With the spread of voice user interfaces, such as Siri and Alexa, many users, including children, are accustomed to using voice to control user interfaces. Speech recognition, natural language processing, and voice synthesis, which form the basis of VUI, are being developed to the point that they can be applied to professional areas, including programming. Furthermore, using VUI makes it easier to overcome the drawbacks of the previous studies mentioned above and to take advantage of users' quick and natural input.

Under this background, this study introduces TurtleTalk, an educational game for kids where children can learn programming easily and enjoyably while talking with the turtle, the voice agent of the game. Child users can move the turtle on the screen to reach the target point by engaging in voice interaction (conversation) with it and receiving feedback from it. Through this process, users can both experience and learn about the programming concepts of "sequencing" and "iteration." Specifically, to match the various utterances spoken by children in a natural situation to the code controlling the

(a) sequencing



(b) iteration

Figure 2: Content users learn with system. (a) Sequencing ("go forward" > "turn right" > "go forward") and (b) iteration ("go forward" > "go forward six times").

movement of the turtle, TurtleTalk artificially generates natural language datasets paired with Turtle Graphic Language (TGL) (described in detail in the next section) and trains the model with it using a neural network-based model.

To explore the possibilities of TurtleTalk in programming education for children, we conducted a user study with eight children. The results demonstrated that voice interaction with TurtleTalk made children more engaged in the game and helped them understand the elements of programming with ease and confidence.

TURTLETALK: PROTOTYPE DESIGN

Here, we give an overview of the design of TurtleTalk. Built on a web-based system, it operates on a computing device equipped with a screen and a speaker, and it is capable of voice recognition so that child users can perform programming tasks on the screen through VUI. Naver Clova API was used for the speech recognition and synthesis (https://www.ncloud.com/product/aiService). The following process was used to design and implement this system.

Designing the voice agent "Turtle"

As the first step, we designed the voice agent of the system. We set a friendly turtle as the persona of the agent [2], who would guide children to play the game with voice. Turtle also appears on the screen and moves on the blocks according to the user's voice commands. By interacting with Turtle, children learn programming.

Designing content: What children will learn

As the content users would learn with the system, we chose "sequencing" and "iteration," both of which are fundamental programming elements. In the case of the former, child users can learn that the program runs in a sequence structure, where an action, or event, leads to the next ordered action that cannot be skipped. The child must move the turtle to the target point in a certain order. In the case of the latter, children can learn that programs often contain loops of instructions that are executed repeatedly. The child learns to move the turtle several blocks at a time rather than moving it one block at a time. We designed four different game stages based on the contents so that children could fully experience each element (see Figure 2).

Designing the conversation flow

We designed the overall flow of conversation between child users and the turtle. We designed the conversation in a turn-taking way, because it is useful for controlling user conversations and letting users focus on tasks [7]. After greeting the user, the turtle explains how to play the game and the programming elements. Then, the user starts the main task consisting of a loop of the turtle's guidance,

1. TGL For the turtle graphics, vector graphics generated by a turtle that moves according to relative coordinates on a Cartesian plane, we specifically designed the TGL for TurtleTalk. It supports the functions of controlling and executing the movements of the character (the turtle) as well as debugging. We made a list of possible TGL commands that control the movement of the turtle of the game. Based on the commands on the list, we designed the TGL generation rule, whose form and grammar are very similar to those of natural languages. For the context-free grammar [9], we divided natural sentences into subject, verb, and intent and then defined the rules by which these elements were combined.
2. Data generation Utilizing the TGL generation rule, we created a large number of natural language sentences and TGL pairs based on probabilities.
3. Model training We designed the seq2seq-based neural network model [11] and trained the model with the generated data. This model allowed the various utterances of child users to be converted to the optimal code.

Sidebar 1: Process of building model that converts child user's utterances to code

the user's action request (command), and again the turtle's confirmation and action, which is repeated until the user successfully moves the turtle to the target location. After all four stages are complete, the conversation ends with the turtle saying goodbye.

Building model that converts child user's utterances to code

In addition to designing the basic conversation flow, we also needed to convert the children's utterances during the conversation into appropriate programming code. We designed the TGL, generated the data, and trained the model (see details in Sidebar 1).

Designing system's GUI

We designed the graphical user interface (GUI) of TurtleTalk to provide users with visual feedback on their programming (see details in Figure 3).

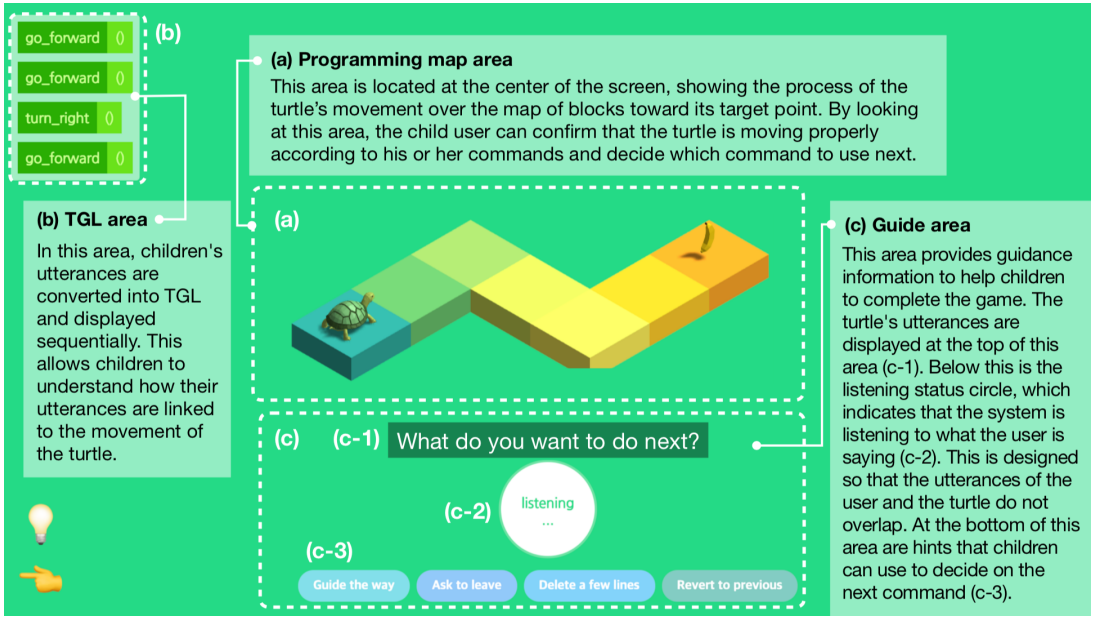


Figure 3: GUI of TurtleTalk.

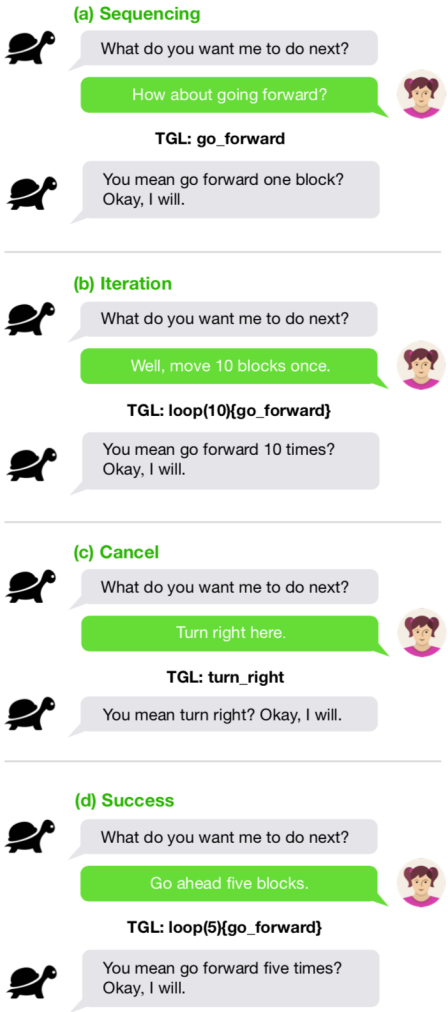


Figure 4: Example of conversation between user and Turtle ((a) Sequencing > (b) Iteration > (c) Cancel > (d) Success).).

USER STUDY

To understand how users interact with TurtleTalk and explore the future possibilities of the system, we designed and conducted a user study of the system.

Study design and analysis methods

We recruited eight participants 6 to 9 years of age (M: 7.62, SD: 1.40). The participants were instructed on how to move the turtle in TurtleTalk using their own spoken words. They performed the task of moving the turtle to the target point. There was no time limit, and they were able to learn these concepts repeatedly by completing the four different stages. After the participants finished their game play, we conducted a semi-structured interview, which was based on the measurement items on programming self-belief of previous studies [5, 10]. All of the game play was video-recorded, and the interviews were audio-recorded and transcribed. The qualitative data from the user study was analyzed using thematic analysis [1].

Results

Through the user study, we found three main issues on the interaction of users with the VUI-based programming environment for children. (The quotes were originally written in Korean, but they have been translated into English) (see Figure 4).

Children can learn programming easily, enjoyably, and confidently. Participants commonly responded that the system was easy and fun and they felt confident in the learning process (P01-P06). P01 said, "I didn't know what programming was, so at first, I was a bit worried. But after playing the turtle game, I found it very easy and fun." P03 said, "It was not difficult to move the turtle on the screen. It was nice that the turtle seemed to understand what I was saying." P02 even said, "I can play a much harder level of the game. The turtle will understand what I say if I just tell it what I am thinking."

Children can understand the elements of programming through games. Participants replied that they were able to understand the "sequencing" and "iteration" of programming through the game (P01-P06). P02 said, "I soon realized that the turtle moved in order. So I told the turtle to move forward or back." P01 said, "When the turtle went the wrong way, I made it come back to the origin." P05 said, "I did not have to say the same thing every time I moved the turtle one space at a time."

Voice interaction allows children to become more immersed in the game. Finally, participants showed more immersion in the game through voice interaction (P01-P06). While talking to the turtle, they tended to personify it, and it motivated their programming activity. P07 said, "Turtles in the real world cannot speak, right? But this turtle can. She really understands me by listening to me. So kind and cute." P06 said, "It was as if I was advising the turtle on what to do."

CONCLUSION AND FUTURE WORK

This study makes two contributions to the human-computer interaction (HCI) community: (1) We have designed a system based on voice interaction for teaching children coding and empirically showed its possibilities by conducting a preliminary user study. (2) We have introduced a neural network-based learning model and translated various children's utterances into appropriate codes to facilitate the integration of VUI into programming education for children.

However, there are limitations to this study. Although we considered various factors in the design of the system, we only conducted the initial study, and we did not verify the performance of each element. In a future study, we plan to evaluate whether children actually learned the elements of programming with more quantitative methods. The performance of the proposed neural network-based model will be also evaluated and improved. Finally, we will also look at the differences between VUI and other user interfaces in a programming environment and seek ways to improve them.

REFERENCES

[1] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.

[2] Stefania Druga, Randi Williams, Cynthia Breazeal, and Mitchel Resnick. 2017. Hey Google is it OK if I eat you?: Initial Explorations in Child-Agent Interaction. In *Proceedings of the 2017 Conference on Interaction Design and Children*. ACM, 595–600.

[3] N Fraser et al. 2013. Blockly: A visual programming editor. *URL: https://code. google. com/p/blockly* (2013).

[4] Google. 2017. Clojure turtle. Git hub. https://github.com/google/clojure-turtle.

[5] Edward F Melcer and Katherine Isbister. 2018. Bots & (Main) Frames: Exploring the Impact of Tangible Blocks and Collaborative Play in an Educational Programming Game. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 266.

[6] Lauren R Milne and Richard E Ladner. 2018. Blocks4All: Overcoming Accessibility Barriers to Blocks Programming for Children with Visual Impairments. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 69.

[7] Cathy Pearl. 2016. *Designing Voice User Interfaces: Principles of Conversational Experiences*. " O'Reilly Media, Inc.".

[8] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, et al. 2009. Scratch: programming for all. *Commun. ACM* 52, 11 (2009), 60–67.

[9] Stephen Scheinberg. 1960. Note on the Boolean properties of context free languages. *Information and Control* 3, 4 (1960), 372–375.

[10] Michael James Scott and Gheorghita Ghinea. 2014. Measuring enrichment: the assembly and validation of an instrument to assess student self-beliefs in CS1. In *Proceedings of the tenth annual conference on International computing education research*. ACM, 123–130.

[11] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.