ASM Sc. J., 13, Special Issue 3, 2020 for CIIDT2018, 6-10

# Tangible Programming Environment by using Augmented Reality (AR) Markers

Momoko Irie[1*], Akiyoshi Miyatake[2] and Keiichi Shiraishi[3]

[1]*Advanced Course in Electronics, Information and Communication Engineering, National Institute of Technology, Kagawa College, Japan*

[2]*Department of Information Engineering, National Institute of Technology, Kagawa College, Japan*

[3]*Department of Communication Network Engineering, National Institute of Technology, Kagawa, Japan*

Visual programming environments are often used for programming education. These make programming education easy to understand and fun. The current programming education has some problems related to deterioration in eyesight, group work and teacher's burden. The purpose of this re-search is to develop a programming environment that solves the above problems. We thought that an environment using TUI (tangible user interface) would be effective. We use Augmented Reality (AR) markers printed on paper as tangible blocks. The program to be created is based on Turtle Graphics and draws a figure by moving the turtle. To reduce the teacher's burden, we prepare exercises. In this research, we propose new AR marker specifications that do not use existing AR library for identification. The identification accuracy of the proposed AR marker is 99% or more. The false detection rate was as high as about 25%. In experiment targeting two adults, the learners solved questions while speaking and choosing AR marker together. We showed that this system using TUI is effective for solving the problems. But this has some problems in usability. We aim for a system in which programming education and other education help each other.

**Keywords:** AR Marker; Tangible Programming; Education

## I. INTRODUCTION

In recent years, programming education at a young age has become popular. In Japan, programming education will become a required course in elementary school from 2020. Currently, they often use visual programming environment like Scratch for programming education. In that environment, we use command blocks on PC (personal computer) screen for programming. A program is created by combining the blocks by a mouse operation. Such environments have the advantage that they can be programmed with a simple GUI (graphical user interface) operation, and the effect is shown visually. This makes programming education easy to understand and fun. On the other hand, there are also problems related to deterioration in eyesight, group work and teacher's burden.

We thought that environment using TUI is effective for solving above problems. In the environment using TUI, the command blocks are realized by tangible blocks, not on the PC screen. Tangible blocks can increase the movement of the eye and prevent us from continuing to see nearby screen. In addition, since the operation of arranging the blocks can be performed by multiple learners, the group-work becomes easy. Horn *et al.* confirmed that users were more interested and actively involved in environment using TUI than that using GUI (Horn, M.S. *et al.*, 2009).

At present, programming environments using TUI include AlgoBlock (Suzuki, Kato & Ito, 1993), Google Project Bloks, Yashiro's research (Yashiro, 2013), Tada's research (Tada, 2015). Since AlgoBlock and Project Bloks use electronic blocks, introduction cost is high. Yashiro proposes environments that uses acrylic blocks. He created three environments with different command block identification

*Corresponding author's e-mail: a17505_irie@yahoo.co.jp

methods (camera, scanner, electric circuit) and examined the requirements of TUI environment. From the results of the experiment, he shows the following six points as a requirement (Yashiro, 2015).

- Blocks must be at least as big as the fingertip (one-two $cm^2$).
- Information that blocks represent can be understood at a glance.
- Programming exercises are complicated to the degree that users are actively discussing each other.
- There are outputs that make it easier to explain by physical action and illustration.
- Changes in programs appear as changes in movements that humans can recognize.
- To have a function to help recognize change of movement.

In our system as well, we need to satisfy these. Tada proposes an environment using AR markers printed on paper. They created a function unique to paper that the program changes by writing to the AR marker. With this function, the learner felt more friendly to this environment. On the other hand, there is a problem that the operation of erasing after writing is laborious. In our research, we use the AR marker printed on paper as tangible blocks as in Tada's research. Yashiro and Tada use existing AR libraries to identify AR markers. In contrast, we propose a new specification of AR markers and create our own identification program. Existing libraries have two problems. The first is that they cannot recognize the same AR markers together. The second is that recognition accuracy becomes worse when using fine patterns. Our program solves these problems.

On the other hand, only by using TUI is insufficient to reduce the burden on teachers. Therefore, we make practice questions to reduce the burden of teachers' thinking about how to teach. Moreover, it will help learner learning other than programming education.

In this paper, we first describe learning-contents and specifications of AR markers. Learning-contents are created based on learning at elementary school. Detailed patterns are drawn on the proposed AR markers, so we are easy to acquire information. Next, we explain the processing necessary to realize the system. Using the system, we conducted an accuracy check of AR marker and an experiment targeted two adults.

## II. OVERVIEW OF THIS SYSTEM

In this system, we use AR marker, web camera and PC (personal computer). Figure 1 shows the system configuration. The learner first arranges the AR markers. Next, shoot the AR markers with a web camera. The system automatically creates a program from the shot image. The learner executes the pro-gram and confirms that the program is correct.

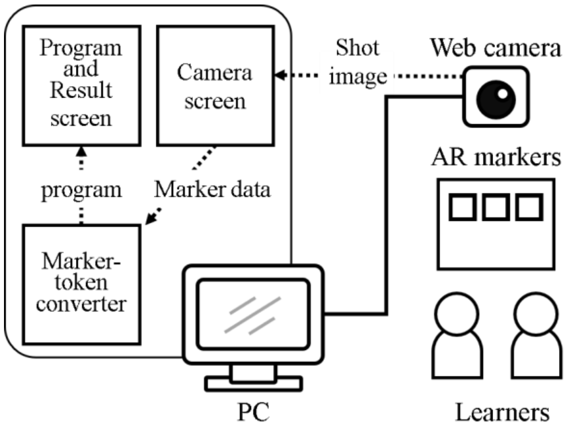



Figure 1. The system configuration

The program to be created is based on Turtle Graphics and draws a figure by turtle movement. Turtle Graphics is often used for introductory programming because the result is clearly visible. The output of Turtle Graphics satisfies the requirement concerning recognition of output by Yashiro.

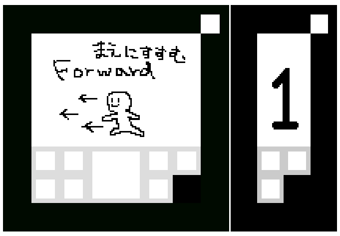
### *A. Learning Contents*

Elementary school and junior high school mainly use visual programming environment. It is aimed at nurturing logical thinking, not mastering of a specific programming language. They often make programs that animate pictures and let robots escape the maze. The advantage of these learnings is that the purpose is clear. However, it is not directly related to learning in class.

In this research, we create practice questions targeted at elementary school fourth grade students and over. The figure to be drawn is a straight line, a square, a triangle, and the like. We anticipate discussions similar to class will occur among learners. This learning-contents satisfies the conditions concerning the programming exercises mentioned by Yashiro.

### *B. AR Markers*

To move the turtle, we need to order like "move forward X steps" or "turn to the right by X degrees". In other words, we need to specify command and numeric value. We prepared two types of AR markers: a command marker and a numeric marker. As examples of AR marker, Fig. 2a shows a command marker of "move forward", and Fig. 2b shows a numeric marker of "1".




(a) Command marker of "move forward" (b) Numeric marker of "1"

Figure 2. Examples of AR marker

The size of numeric marker is equal to a half size of the command marker. To identify by the same processing as command markers, use two numeric markers as a set. The AR marker consists of three important parts. Figure 3 shows the role of each part.

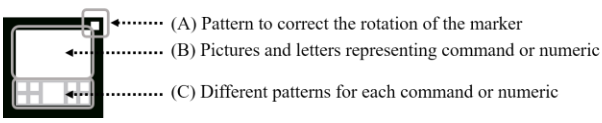



Figure 3. The role of each part

Different black and white patterns for each command or numeric in (C) are used to identify AR markers. Since (B) has no relation to identification, we can use detailed pictures and letters. This specification satisfies the requirement stated by Yashiro "Information that blocks represent can be understood at a glance".

Tada proposed a method of writing to the AR marker. He described AR markers that do not write in are inferior in familiarity to those that write in. Also, that in our research has no opera-tion unique to paper. However, since the subject of our system includes elementary school students, we need to reduce factors that impede learning. We focused on concentration of learners and proposed a specification that enables programming only by placement.

## III. PROCESSING

### *A. Detection and identification of AR marker*

We use OpenCvSharp which is the OpenCV wrapper library. First, the system detects rectangular and convex contours from the shot image. Whether the contour is rectangle can be known from the number of vertices. Transform each contour to a square. Next, rotate the image so that the pattern shown in Fig. 3 (A) comes to the upper right. If the pattern does not come to the upper right corner even if the image is rotated, it is judged not an AR marker. Then check the pattern shown in Fig. 3 (C). Figure 4 shows the order to check the pattern.

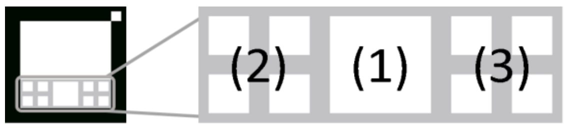



Figure 4. Order to check the pattern

First, it examines whether the part of (1) is white or black. If it is white, it is a command marker. If it is black, it is a numeric marker. The part of (2) and (3) are divided into four places and it is examined whether it is white or black. Record 0 if it is white, 1 if it is black. For each of (2) and (3), convert the recorded numeric string of 0 and 1 to decimal number. We refer to these decimal numbers as (2)' and (3)'. Multiply (2)' by 10 and add (3)'. The calculation result is marker number for command marker or numerical value for numeric marker. The result of calculation is any one of 0 to 159.

### *B. Convert Marker Numbers Identified from AR Markers to Tokens*

Compare the obtained marker number with the marker-token conversion table. The table records the marker number, its corresponding token and whether a numeric value is required.

### *C. Execute the Program*

We use the library Microsoft.CodeAnalysis.Csharp. Scripting. This is a library for script execution of C# code on C# program. Unique command for realizing Turtle Graphics were implemented with reference to the published sample program (Iwanaga, 2016). The program can be executed by passing it to the library in string type.

## IV. EXPERIMENTS

### *A. Accuracy Check of AR Marker*

The illuminance of the room was 400 lux. The distance between the web camera and the AR marker was set to 40 cm. At the school's classroom the minimum illumination is 300 lux and 500 to 700 lux is recommended. In the inspection, arrange that there is no gap between one set of numerical markers and part of AR markers does not become a shadow. When there is a gap or when it becomes a shadow, detection leakage or identification error occurs.

The identification rate was 99% or more. However, the false detection rate was as high as 25%. Here, false detection is to detect those that are not AR markers as AR markers. False detection occurred when something rectangular was reflected on the screen or when the shadow became a rectangle.

### *B. Use Experiments*

We conducted an experiment targeted two adults. The questions were printed on paper. Figure 5 shows the questions used in experiment. Table 1. shows the answers when the initial position is point B (lower left of the triangle).

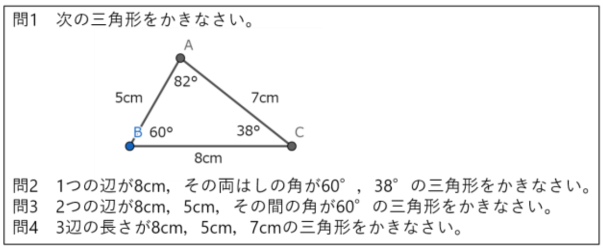



Figure 5. The questions used in experiment

Table 1. Answer to question 1 to 3

| **Question 1** | **Question 2** | **Question 3** |
|---|---|---|
| right(30) | right(30) | right(30) |
| forward(5) | forward(8) | forward(5) |
| right(98) | backward(8) | xPos(11) |
| forward(7) | right(60) | yPos(9) |
| right(142) | forward(8) | setPos() |
| forward(8) | left(142) | left(120) |
| | forward(8) | forward(8) |

The following tendency was seen in the experiment.

- WHILE LISTENING TO THE EXPLANATION, THEY WERE GETTING INFORMATION BY LOOKING AT THE COMMAND MARKER.
- WHEN SOLVING QUESTIONS, THEY WERE MAINLY LOOKING AT PRINTED QUESTIONS AND AR MARKERS ALTERNATELY.
- IT SEEMED DIFFICULT TO HANDLE BECAUSE THE AR MARKER WAS THIN.
- IT TOOK A LONG TIME TO FIND THE DESIRED AR MARKER.
- THERE WERE MANY DETECTION LEAKAGES DUE TO GAPS BETWEEN NUMERIC MARKERS.
- THERE WAS NO MISIDENTIFICATION OF THE AR MARKER.
- THE OPERATION TO SELECT THE AR MARKER WAS PERFORMED BY TWO PERSONS, BUT THE OPERATION TO ARRANGE CAREFULLY WAS DONE BY ONE PERSON.
- THEY MISTOOK THE TURTLE'S ROTATION ANGLE FOR THE FIRST TIME.
- THEY WERE THINKING WHILE POINTING AT A PICTURE OF QUESTIONS, DRAWING INFORMATION AND TALKING.

Looking at the result of operation, they uttered such as "different from what I thought", "why", "I see", and so on. Also, we got feedback such as "It is brainwork." and "Is this programming?".

## V. DISCUSSION

In the questions and their answers, we confirmed that we can use this system to solve the questions of drawing figures used in class. However, there are questions that cannot be solved. In class, basically the format like question 1 (given information on three sides and three angles) is not given, and the form of question 2 to 4 are often used. To draw a triangle from the information of question 4 to be used, we need to use compasses. We think that it can be solved in this system if there is a circular drawing command. Compasses are used not only in triangles but also in many questions. In addition, we need to create a function to add a graphic image of questions. Also, in elementary school learning, to learn speed is particularly hard. We think about a way to help such learning. From the results of the accuracy check and use experiment, the following can be seen. The identification accuracy of the detected AR marker is enough. However, countermeasures against detection leakages and false detection are not enough. We anticipate that detection leakages due to gaps can be dealt with by closing images.

From the result of the experiment, we found it easy to obtain information from the proposed AR marker. They were touching the AR marker while talking, without keeping seeing the screen while answering. From this, we consider that this system is effective for solving the problem of programming education mentioned. On the other hand, we found out that AR markers just printed on paper are difficult to handle and need politeness. Also, with the monochrome AR markers, we are hard to distinguish each marker until we see the pattern. We aim to solve these problems by printing on thick paper and color coding. Practice questions were difficult enough for even adult experiment participants to give a wrong answer. We conduct experiments on low age and consider difficulty level.

## VI. CONCLUSION

In this research, for solving the problem of programming education, we developed a tangible programming environment by using AR markers. Sequential structural programs can be de-scribed in this system. We proposed a new specification of AR marker and created questions based on learning guidance. Experiments showed that this system is effective for solving problems. Detection of AR marker, improvement of ease of use and addition of necessary functions are future tasks. We will also experiment with low age learners and improve the questions. We aim for a system in which programming education and other education help each other.

## VII. REFERENCES


Horn, MS, Solovey ET, Crouser, RJ & Jacob, RJK 2009, ‘Comparing the Use of Tangible and Graphical Programming Languages for Informal Science Education’, *Proceedings of the 27th International Conference on Human Factors in Computing Systems - CHI 09*, pp. 975-984.

Iwanaga, N 2016, C# scripting sample - Turtle Graphics, viewed 20 August 2018, <https://github.com/ufcpp/UfcppSample/tree/master/Chapters/Scripting/TurtleGraphics>.

Suzuki, H, Kato, H & Ito, Y 1993, ‘A tangible programming language "AlgoBlock" - a tool for thought and collaboration’, *Proc. 46th National Convention of IPSJ*, 1, 1.9-1.10.

Tada, K & Tanaka, J 2015, ‘Tangible programming environment using paper cards as command objects’, *Procedia Manufacturing* vol. 3, pp. 5482-5489.

Thomas, S 1997, *Guide to personal efficiency*, Adelaide University, viewed 6 January 2004, <http://library.adelaide.edu.au/~sthomas/papers/perseff.html>.

Yashiro, T 2015, ‘Design guidelines in tangible programming environments for programming learning’, Master thesis Future University Hakodate.

Yashiro, T & Mukaiyama, K 2013, Material programming - A visual programming development environment with material -. ITE Technical Report, vol. 37, no. 17, pp. 223-226.