Comparative Study on Programmable Robots as Programming Educational Tools

Shohei Yamazaki[1] **Kazunori Sakamoto**[2] **Kiyoshi Honda**[1]
Hironori Washizaki[1] **Yoshiaki Fukazawa**[1]

[1] Department of Computer Science and Engineering
Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo, Japan
Email: shohei-hamamatsu@moegi.waseda.jp, khonda@ruri.waseda.jp, washizaki@waseda.jp, fukazawa@waseda.jp

[2] Department of Computer Science and Engineering
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan
Email: exkazuu@nii.ac.jp

Abstract

Computational Thinking skills are basic and important to manipulate computers. Currently, several systems exist to provide an effective way to learn programming that use computers, smartphones, tablets, or programmable robots. Although studies have reported improved programming skills and motivation to learn programming using an on-screen application or a programmable robot, the benefits of these tools have not been directly compared.

To resolve this issue, especially with regard to motivation to learn programming and impression of programming, we conducted a large-scale comparative experiment involving 236 middle and high school students to evaluate the effects of a game-based educational application and programmable robots on learning programming. We then compared the effects of a game-based educational application with and without programmable robots on learning programming. We found that employing programmable robots on learning programming did not always give an improvement to all students.

Keywords: comparative study, programming education, programming environment, programmable robot, motivation, impression

1 Introduction

Computers have become commonplace. Because of this, Wing has suggested that people should learn Computational Thinking, which she defines as basic skills for manipulating computers (Wing 2006). Thus, we developed educational tools that teach computational thinking.

The motivation to learn and the impression of learning contents are very important not only when developing computational thinking, but learning in general. Several studies have focused on the importance of motivation to learn programming (DeClue 2003, Feldgen & Clua 2004, Kelleher et al. 2007, Jenkins 2001). Feldgen and Clua argued that instructors are critical in motivating students (Feldgen & Clua 2004). Jenkins argued that motivation is the product of expectation and value; thus, students must expect to succeed in learning and value their achievements (Jenkins 2001). These studies demonstrate the importance of providing learners with expectations and the value of being able to program.

Several educational tools have been developed to provide motivation to learn programming (Kölling & Henriksen 2005, Esper et al. 2013, Bezakova et al. 2013). For example, Scratch is a visual and block-based programming learning environment that allows learners to learn programming intuitively (Resnick et al. 2009). Several studies have investigated Scratch (Rizvi et al. 2011, Lewis 2010). Malan and Leitner as well as Maloney et al. have reported the effects of using Scratch as a programming educational environment on learning programming (Malan & Leitner 2007, Maloney et al. 2008). In addition, programmable robots have the potential to facilitate and inspire motivation to learn (Nourbakhsh et al. 2000, Lalonde et al. 2006). In fact, several studies have used robots as educational tools (Kumar & Meeden 1998, Billard et al. 2008). One such robot is LEGO®Mindstorms®. Those learning programming using LEGO Mindstorms create a robot by combining sensors and motors. Barnes reported a study in which Java was taught using Lego Mindstorms as a programming educational tool (Barnes 2002).

Although it is clear that introducing these learning environments and educational tools into learning programming is effective, the following remains unclear. Do these educational tools improve motivation to learn programming? Do these tools improve the impression of programming? How much is the actual improvement using these tools?

In this paper, we evaluate the effects of a game-based educational application and programmable robots on learning programming. We gathered 236 middle and high school students, most of whom were unfamiliar with programming, to participate in our experiment. Then we compared the effects of a game-based educational application with and without programmable robots on the motivation to learn programming and the impression of programming.

The contributions of this paper are:

- We conducted a large-scale comparative experiment where 236 students learned programming.
- We compared the effects of a game-based appli-

cation with and without programmable robots on the motivation to learn programming and the impression of programming using a questionnaire containing six items.

- We investigated the gender differences of the effects of programmable robots furthermore.

The rest of this paper is organized as follows. Section 2 details related works. Section 3 describes the game-based application, while two different programmable robots are described in Section 4. Section 5 details the comparative experiments. The results are evaluated in Section 6. Finally, our conclusion and future work are detailed in Section 7.

2 Related Work

Several studies have examined the effects of programming educational tools and environments on learning motivation. For example, there are several programming educational environments (Kelleher et al. 2007, Long 2007, Kölling & Henriksen 2005, Esper et al. 2013, Bezakova et al. 2013). Additionally, several studies have employed programmable robots as programming learning tools (Nourbakhsh et al. 2000, Lalonde et al. 2006, Fagin et al. 2001, Magnenat et al. 2012). Although they demonstrated the effects of teaching programming concepts to students without programming experience, the influence of game-based applications with and without programmable robots on learning were not compared.

McNally et al. investigated the motivation of two student groups at university (McNally et al. 2006). One group participated in LEGO Mindstorms activities, while the other took a traditional introductory programming course. The difference between our study and McNally et al. is that they discussed the motivation of undergraduates already familiar with programming. Our study investigates not only the motivation but also the impression of programming for middle and high school students, most of whom are unfamiliar with programming.

Scratch, which is aimed at novice programmers, was created by a group at the MIT Media Laboratory in collaboration with a group at UCLA (Resnick et al. 2009). Rizvi et al. investigated the effect of using Scratch to improve the retention and performance of at-risk computer science majors (Rizvi et al. 2011). The difference between these studies is that they targeted undergraduates majoring in computer science and investigated differences between students enrolled in CS0 and CS1, while we investigated the motivation to learn programming and the impression of programming of individuals unfamiliar with programming.

Lewins compared the effects, especially attitude and learning programming concepts, using either Logo or Scratch for sixth grade students learning programming (Lewis 2010). Although the Logo environment seemed to support students' confidence, interest in programming, and understanding of loop constructs, Scratch improved students' understanding of the construct conditions. These studies only treated on-screen applications, whereas our comparative study involves both an on-screen application and a programmable robot.

Previous studies have not compared the effects of game-based educational applications with and without programmable robots on learning to program as long as we investigated. Thus, we conducted such a comparative study with an emphasis on the motivation to learn and the impressions of programming.

3 Game-based Educational Application

We developed an educational tool called ManekkoDance (Sakamoto et al. 2013). ManekkoDance is a programming educational tool that runs as an application on a smartphone or a tablet. There are two reasons why we developed an educational application for a smartphone or a tablet instead of a desktop or laptop computer. First, mobile applications can motivate students (Mahmoud 2008). Second, learning can occur anytime and anywhere using a smartphone or a tablet rather than a computer. ManekkoDance is a game where users move two yellow and orange baby chicks and answer problems by imitating the movements of two white and ocher chickens correctly as models by programming. For example, if the chickens raise their right wings, users have to raise the baby chicks' right wings. ManekkoDance shows whether the user program is correct (see Figure 1).

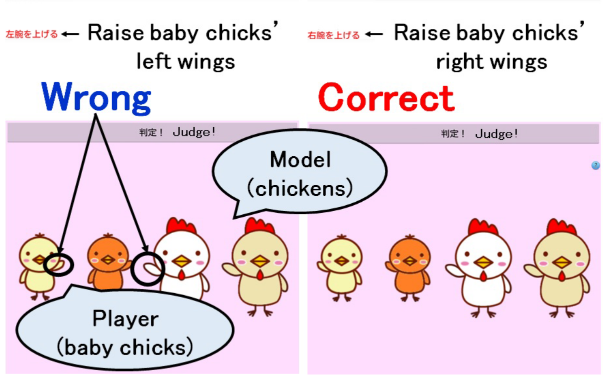

Figure 1: Screenshot of ManekkoDance (Left and right show an incorrect and correct program, respectively)

Users can play ManekkoDance, even if users connected programmable robots or did not connect programmable robots. Thus, we adopted ManekkoDance in this experiment.

To understand our experiments, here we briefly describe the features and learning contents of this application.

3.1 User Interface

A previous study reported that a good user interface can motivate learners (Cho et al. 2009). ManekkoDance has appealing interfaces such as the baby chick and chicken characters and icons which move baby chicks. Several students said, "The icons and characters are lovely or cute."

3.1.1 Icon-based Non-verbal Programming Language

Figure 2 shows that ManekkoDance interconverts between a verbal language and icon-based nonverbal programming language, allowing users to more easily write and intuitively understand a program.

Figure 3 shows sixteen icons that correspond to the baby chicks' actions. To play the game, users employ these sixteen icons and natural numbers. Users also have the option to use verbal language.

3.1.2 Characters

To prevent boredom while learning to program, we adopted appealing characters. For example, if the

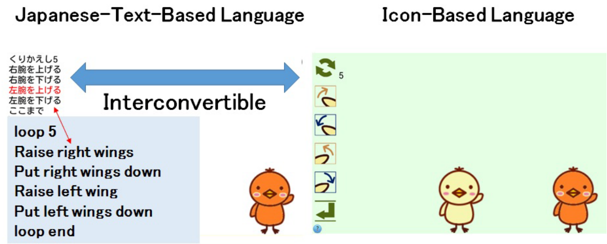

Figure 2: Same program written in a Japanese-text-basaed language (left) and icon based language (right)

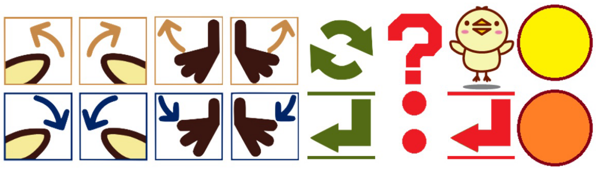

Figure 3: Sixteen icons

written program contains an error, instead of an error screen, the baby chicks fall down. Programming an unnatural motion gives rise to errors. For example, entering a icon to raise the baby chicks' right wings when their wings are already raised causes the baby chicks to fall down.

3.2 Learning Contents

We think that computational thinking is a common concept to various programming languages. We are referring to their idea about computational thinking (Brennan & Resnick 2012).

This game consists of stages so that users can learn gradually. The stages require users to combine the following four concepts in computational thinking. By playing the game, users can learn four concepts in computational thinking that are used in common in many programming languages:

- Sequences
- Concurrency
- Loops
- Conditionals

To view the flow of a sequence, the executed line is sequentially highlighted by a red letter in the execution screen. This allows users to comprehend sequences.

If a user enters plural icons in the same line, the program runs simultaneously. For example, if a user enters two icons in the same line to raise the right and left wings, the baby chicks simultaneously raise both wings. Therefore, users can learn concurrency intelligibly.

Most programs contain a loop function. Thus, in ManekkoDance, users can employ a loop function if they want the chicks to repeat a motion. Figure 4 shows the example program of a loop function in this game.

For example, if a user would like to repeat a chicks' motion, a program is inserted between a loop command, which consists of the starting symbol and a natural number to indicate the number of times to repeat the motion, and a green ending symbol. One

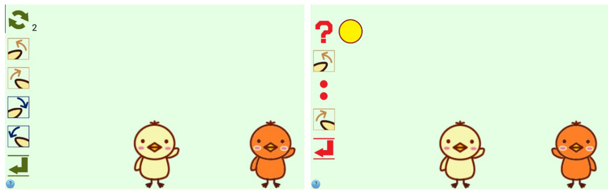

Figure 4: Example programs of loop functions (left) and conditionals (right)

stage requires that a user writes a program so that the baby chicks repeat the motions to raise their left wing, their right wing, put their left wing down, and put their right wing down. This repeated sequences teaches the convenience of the loop function.

Conditionals are important concepts that are used frequently in programming. Users can learn the conditional concept by choosing to move only one of the baby chicks. Figure 4 shows the example program of conditionals in this game. The conditional command consists of the following rules. A user must enter a red question mark, which means "if", and yellow or orange circle which means yellow or orange baby chick in the same line. A red colon means "else". Conditionals end at a red symbol. For example, conditionals make the yellow chick raise its right wing while the orange chick raise its left wing (see Figure 4).

4 Programmable Robots

As mentioned in Section 2, several programming educational tools such as programmable robots have been developed. The processing result of the program written by a learner is not only reflected in the software but also in the robot (e.g., LEGO Mindstorms), which a learner can see and touch. To evaluate the effects between game-based educational applications (on screen) and programmable robots on the ability to learn programming, we conducted a comparative experiment with an emphasis on motivation to learn programming and impression of programming.

By connecting Manekko Dance and two robots, a user can operate the two robots from ManekkoDance. For example, if a user writes a program to move the baby chicks' right wing, the two robots raise their right hands as well (see Figure 5). Because a student may dislike a particular robot, we used two different programmable robots. That is, we avoided things that could decrease motivation to learn or negatively impact impression of programming.

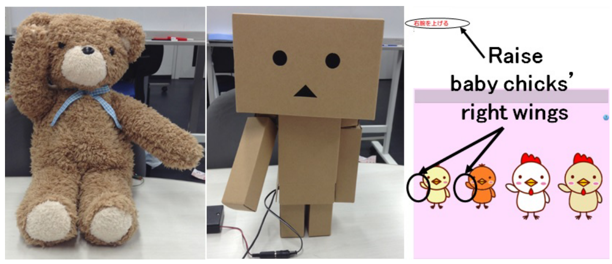

Figure 5: Two Robots interlocked with ManekkoDance (Stuffed Teddy Bear Robot, Cardboard Robot and screenshot of ManekkoDance on left, center and right sides, respectively)

4.1 Stuffed Teddy Bear Robot

We used a Stuffed Teddy Bear Robot (STBR) (Takase et al. 2013) which can move its head and hands as well as roll its head.

STBR has two features: an appealing appearance and a soft texture. This robot is a cuddly teddy bear with fluffy fur. Takase et al. argued that the fluffiness is a factor of loveliness (Takase et al. 2013). Additionally, STBR is so soft that a user can strongly grasp it. Its moving parts consist of fabrics such as cloth, thread, and cotton. The fluffy fur is a factor that makes STBR soft to the touch.

Figure 6 shows the connection of STBR and ManekkoDance, which uses a Wireless Fidelity (Wi-Fi) and a Web application. STBR, a personal computer (PC), and a smartphone or tablet are connected through Wi-Fi. The PC functions as a Web server. The application on the smartphone or tablet sends the signal to move STBR to the PC, which then sends the signal to STBR.

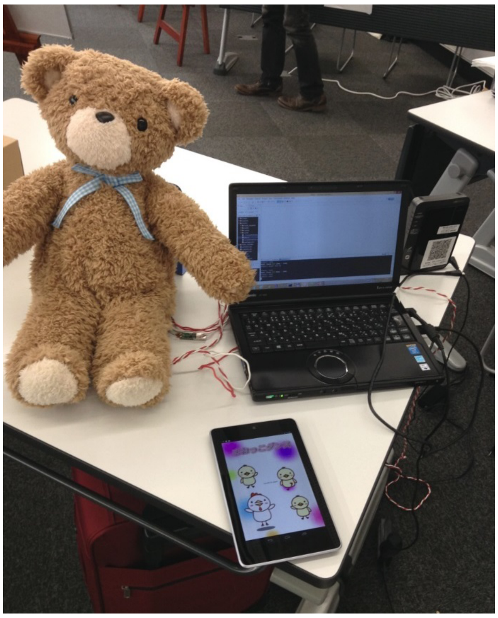

Figure 6: STBR connected with ManekkoDance

4.2 Cardboard Robot

We also used a Cardboard Robot called DANBOARD™, which is a popular character that appearing in Japanese comics. The Cardboard Robot can move its hands differently from STBR. The Cardboard Robot has two main features: a pretty appearance that is not a typical robot and a form that is familiar to users.

Figure 7 shows the connection of Cardboard Robot and ManekkoDance. Moving the servomotor attached to this robot's arms via a pulse wave allows its arms to be raised and lowered. The Cardboard Robot is connected to a smartphone or tablet through the earphone jack.

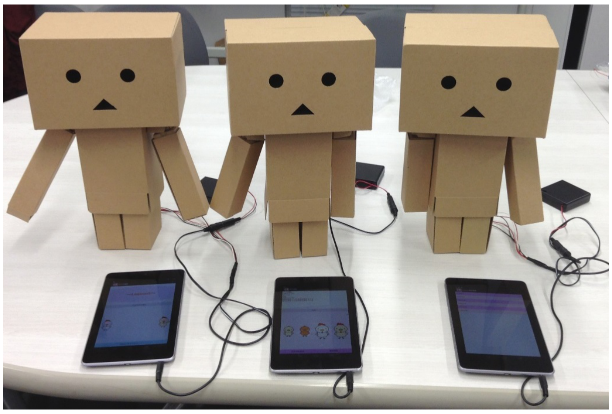

Figure 7: Cardboard Robot connected with ManekkoDance

5 Experiment

We conducted a large-scale comparative experiment involving 236 middle and high school students who were inexperienced programmers attending an open campus event at our university on August 2 and 3, 2014. Open campus is an event in which an individual can participate freely in Japan. We asked students about programming experience by the before questionnaire.

Some students used one STBR connected to ManekkoDance, others used one of the three Cardboard Robots connected to ManekkoDance and the others used ManekkoDance alone as educational tools. To evaluate the effects of a game-based educational application and programmable robots on learning programming, we randomly divided the students into three groups by distributing numbered tickets. Students were divided into three groups according to the numbered tickets (Table 1):

Group A: Each student who learned programming using only ManekkoDance.

Group B: Each student who learned programming using STBR connected to ManekkoDance as a programmable robot.

Group C: Each student who learned programming using a Cardboard Robot connected to ManekkoDance as a programmable robot.

Group	Boys	Girls	Total
A	76	35	111
B	38	23	61
C	41	23	64
B&C	79	46	125
A&B&C	155	81	236

Table 1: Numbers of people participating in this experiment

Each student completed a questionnaire before and after participating in the experiment. For each student, we compared the responses of these two questionnaires and analyzed the effects of a game-based educational application with or without programmable robots on learning from two viewpoints: the motivation to learn programming and the impression of programming.

The experimental procedure was the same for all groups. First, students completed the before questionnaire. Then they learned programming using the tools based on group assignment. Finally they completed a survey after the experiment. The experiment lasted 30 minutes per student. The questionnaire contained six questions. In addition, we classified the

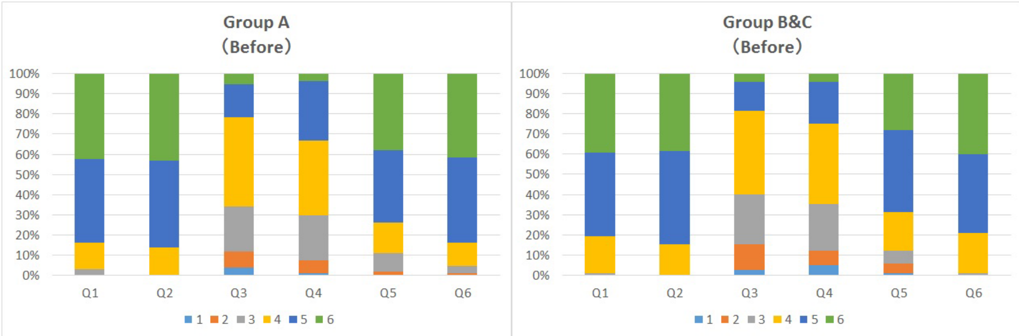

Figure 8: Bar graph of the results of Group A and Groups B&C prior to the experiment. Color scales denote a rating of 1(strongly disagree)6(strongly agree), respectively. Q1 (motivation), Q2 (impression), Q3 (self-confidence), Q4 (liberal arts), Q5 (gender) and Q6 (usefulness)

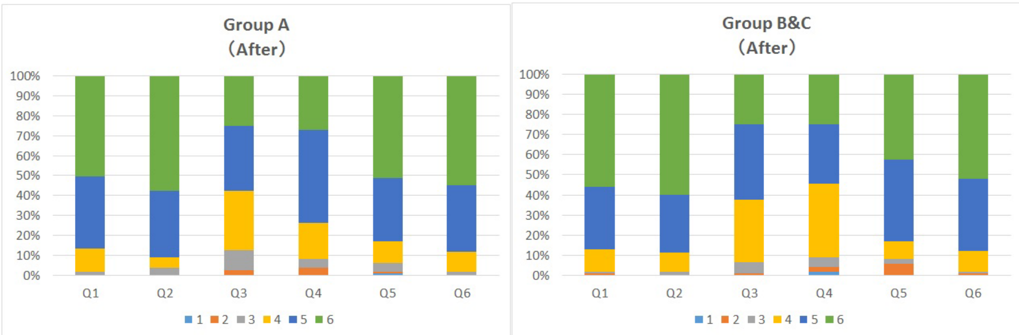

Figure 9: Bar graph of the results of Group A and Groups B&C after the experiment. Color scales denote a rating of 1(strongly disagree)6(strongly agree), respectively. Q1 (motivation), Q2 (impression), Q3 (self-confidence), Q4 (liberal arts), Q5 (gender) and Q6 (usefulness)

motivation to learn and the impression of programming into six question items more finely as follows:

Q1: I want to learn programming. (motivation)

Q2: I feel that programming is fun. (impression)

Q3: I think that I can program. (self-confidence)

Q4: I think that liberal arts students can do programming. (liberal arts)

Q5: I think that being good at programming are related to gender. (gender)

Q6: I think that programming skills are useful. (usefulness)

6 Evaluation

We evaluate the results of our experiment and answer following RQs:

RQ1: Does using a game-based application and a programmable robot result in a difference in motivation and impression of learning programming?

RQ2: Compared to a game-based application, does using a programmable robot increase the rate of positive responses to Q1 (motivation), Q2 (impression), Q3 (self-confidence), Q4 (liberal arts), Q5 (gender) and Q6 (usefulness) in the survey?

6.1 Results

We evaluated the before and after questionnaires to compare the effects of a game-based application with and without programmable robots on the motivation to learn programming and the impression of programming.

	Before		After		After − Before	
	Q1B	Q2B	Q1A	Q2A	Q1A−Q1B	Q2A−Q2B
a_1	4	5	6	6	2	1
a_2	3	4	6	5	3	1
Average					2.5	1

Table 2: Example of the subtraction method

Group	Q1	Q2	Q3	Q4	Q5	Q6
A	0.117	0.153	0.901	0.901	0.261	0.216
B&C	0.216	0.240	1.152	0.880	0.336	0.192
Change Rate (B&C/A)	1.844	1.279	1.567	0.977	1.286	0.888

Table 3: Average of the subtraction results

For the comparison, the responses from Groups B and C were combined and compared to the responses from Group A for the six items described in the previous section (Q1 Q6). All of the students replied to the questionnaires on a six-point scale where a six indicated strongly agree and a one indicated strongly disagree.

Figure 8 shows the ratings prior to the experiment, while Figure 9 shows the ratings after the experiment. The figures employ color scales where aqua, orange, gray, yellow, blue, and green denote a rating of 1 6, respectively.

Because directly comparing the raw data (Figures 8 and 9) did not clearly demonstrate differences between answers regarding motivation and impression of programming, we employed a different analysis approach. For each question, we subtracted the value before from the value after the experiment for each person. Table 2 shows an example using Q1 (Q2) where Q1B (Q2B) and Q1A (Q2A) denote before and after the experiment, respectively, while a_n denotes individual responses. For example, if a_1 answered 4 to Q1 before the experiment and 6 after the experiment, the net value is 2. Then the average difference was determined using all the responses for Group A and Groups B&C.

Table 3 and Figure 10 show the average values of the subtraction method for all six questions. In Figure 10, blue and orange indicate Group A and Groups B&C, respectively.

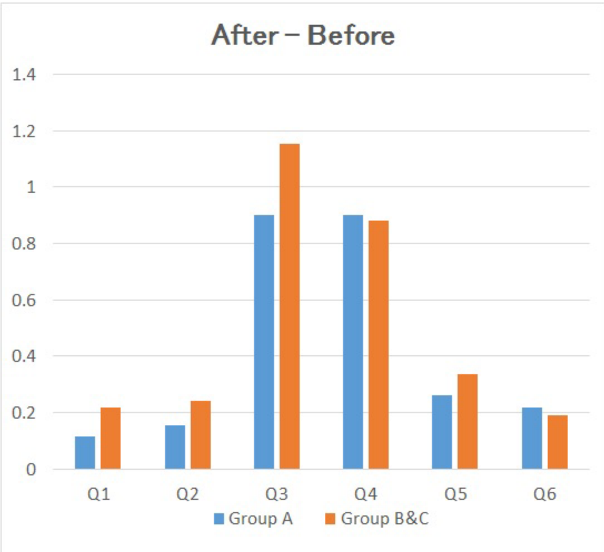

Figure 10: Bar graph of the average of the subtraction value. Blue and orange indicate Group A and Groups B&C, respectively. Q1 (motivation), Q2 (impression), Q3 (self-confidence), Q4 (liberal arts), Q5 (gender) and Q6 (usefulness)

6.2 Discussion

In Table 3 and Figure 10, RQs can be answered.

RQ1: Differences clearly exist between using a game-based application with and without a programmable robot.

RQ2: Q1) Employing programmable robots increases the positive responses to Q1 (motivation) 1.844 times more compared to a game-based application alone. Programmable robots may motivate students to learn programming compared to a game-based application alone.

Q2) Employing programmable robots increases the positive response to Q2 (impression) 1.279 times more compared to a game-based application alone.

Q3) Employing programmable robots increases the positive response to Q3 (self-confidence) 1.567 times more compared to a game-based application alone. Moving programmable robots connected to a game-based application may provide students with self-confidence compared to a game-based application alone.

Q4) Employing programmable robots slightly decreases the positive response to Q4 (liberal arts) (0.977 times) compared to a game-based application alone. Liberal arts is almost changeless when programmable robots are compared to a game-based application alone. We discuss the result about liberal arts later.

Q5) Employing programmable robots increases the positive response to Q5 (gender) 1.286 times more compared to a game-based application alone. We discuss the result about gender later.

Q6) Employing programmable robots decreases the positive response to Q6 (usefulness) (0.888 times) compared to a game-based application alone. Q6 (usefulness) may be ineffective because programmable robots can act only simple things. For example, programmable robots can move only both hands.

Liberal Arts: Andersen et al. reported that fewer liberal art students are interested in programming compared to science students (Andersen et al. 2003). Although the average value with regard to Q4 (liberal arts) decreases when using a programmable robot, most of the students participating in the experiment have not settled on a major. Thus, Q4 (liberal arts) may be ineffective for the participants. Because the students participating in the experiment have not settled on a major, we cannot go into detail about the differences between liberal arts majors.

Group	Gender	Q1	Q2	Q3	Q4	Q5	Q6
A	Boys	0.118	0.197	0.947	0.987	0.184	0.211
B&C	Boys	0.316	0.266	1.076	0.848	0.329	0.228
A	Girls	0.114	0.057	0.800	0.714	0.429	0.229
B&C	Girls	0.043	0.196	1.283	0.835	0.345	0.130
Change Rate (B&C)/A	Boys	2.672	1.347	1.136	0.859	1.787	1.082
	Girls	0.380	3.424	1.603	1.309	0.812	0.571

Table 4: Average subtraction values by gender

Gender: The less number of girl students who, major in computer science has become a problem (Olivieri 2005). Thus, we considered that girl students would not be interested in programming compared to boy students. However, Q5 (gender) in Table 3 and Figure 10 shows that the programmable robots have a positive result on the average change. To investigate the gender difference, we divide the results of the before and after questionnaires by gender. Table 4 and Figure 13, 11 and 12 show the results.

For Q2 (impression of programming) and Q3 (self-confidence) the average change when using a programmable robot increases for both genders. Additionally, for Q2 (impression of programming) and Q3 (self-confidence), it is more effective for girl students to employ programmable robots than for boy students. Especially, for Q2 (impression of programming), while the boys' average change is 1.347, the girl' is 3.424. It is more effective for girl students to employ programmable robots compared to boy students because the girls' average change is 2.54 times of boys'.

For Q1 (motivation), Q5 (gender) and Q6 (usefulness), the boys' responses increase, while the girls' decrease. For Q1, while the boys' average change is 2.672, the girls' is 0.380. It is more ineffective for girl students to employ programmable robots compared to boy students because the boys' average change is 7.031 times of girls'. For Q5 (gender), in Table 3, employing programmable robots increases the positive response to Q5 (gender) 1.286 times more compared to a game-based application alone was obtained. In detail, while the girls' average change was 0.812, the boys' was 1.787. For Q6 (usefulness), while the boys'

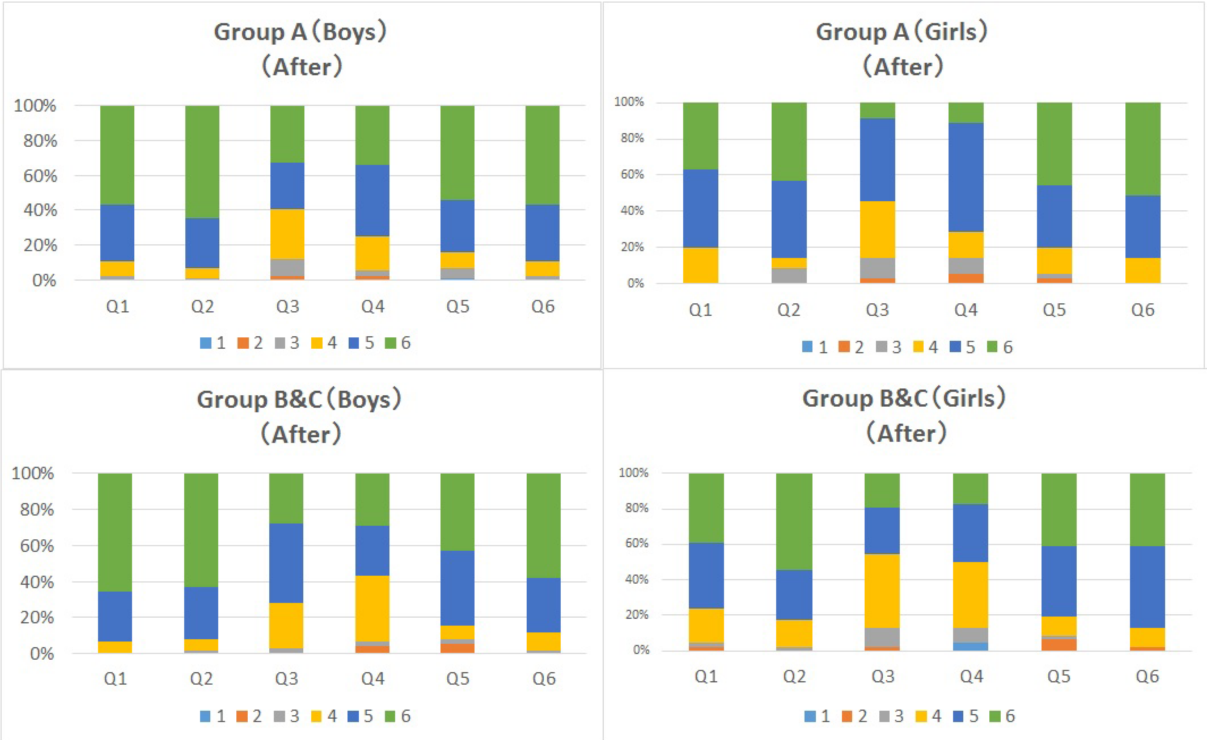

Figure 11: Bar graph of the results of Group A, Groups B&C after experiment according to gender. Color scales denote a rating of 1(strongly disagree)6(strongly agree), respectively. Q1 (motivation), Q2 (impression), Q3 (self-confidence), Q4 (liberal arts), Q5 (gender) and Q6 (usefulness)

average change was 1.082, the girls' was 0.571. It is more ineffective for girl students to employ programmable robots than boy students.

For Q4 (liberal arts) the boys' responses decrease, but the girls' responses increase. As we stated previously, we cannot go into detail about the differences between science and liberal arts majors.

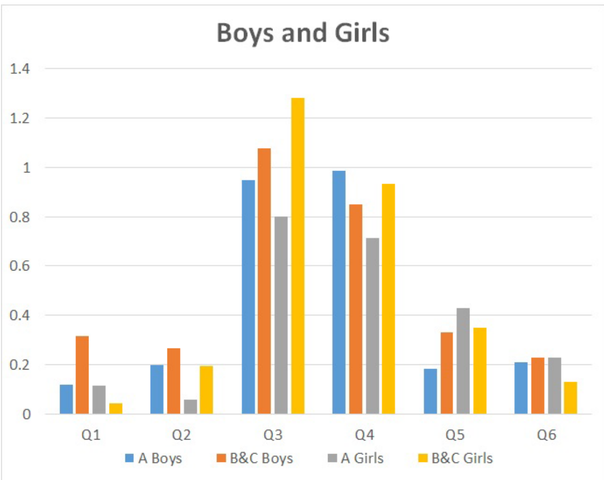

Figure 12: Bar graph of the average of subtraction value. Blue and orange indicate boy students of Group A and Groups B&C, respectively. Gray and yellow indicate girl students of Group A and Groups B&C, respectively.

6.3 Limitation

We analyzed the values of the subtractions using Wilcoxon rank sum test. The results are shown in Table 5.

Question	W	p-value
Q1 (motivation)	6247	0.1309
Q2 (impression)	6377.5	0.2354
Q3 (self-confidence)	6119	0.1031
Q4 (liberal arts)	6994.5	0.9089
Q5 (gender)	6937.5	1
Q6 (usefulness)	6885	0.9082

Table 5: The result of Wilcoxon rank sum test

The p-values of Q1, Q2, Q3, Q4, Q5 and Q6 are 0.1309, 0.2354, 0.1031, 0.9089, 1 and 0.9082, respectively. All of these p-values are larger than 0.05 ($p > 0.05$). There are no statically significant differences in this experiment. However, we do not change our opinions in this research. We think that because there were few scales in this experiment, there are no statically significant differences.

6.4 Threats to Validity

We considered four factors that may influence our findings.

Because we employed questionnaires, the feeling expressed by an adverb such as strongly vs. somewhat in the rating system may vary by individual. Thus, the responses may not be reliable, and our analysis of the motivation to learn programming and the impression of programming may be impacted.

Our experiment only involved middle and high school students. The results may differ if individuals in other age groups participated. Thus, the age of the participants may influence the results.

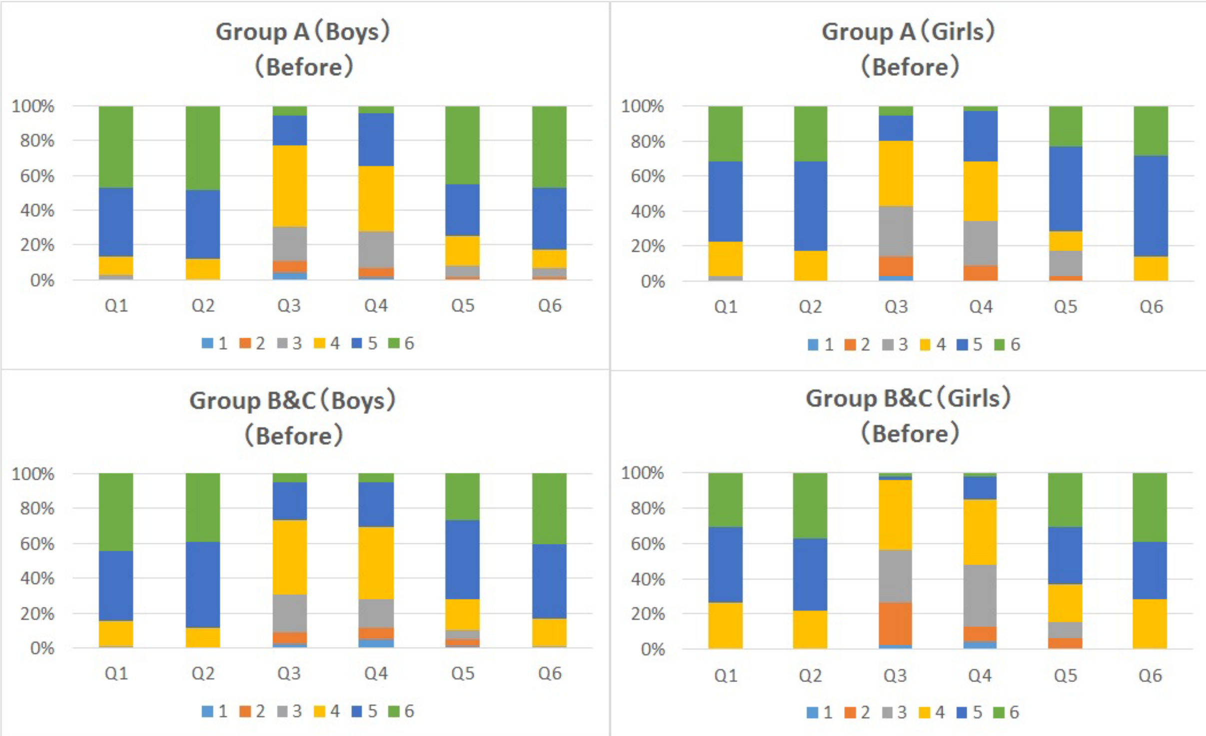

Figure 13: Bar graph of the results of Group A, Groups B&C before experiment according to gender. Color scales denote a rating of 1(strongly disagree)6(strongly agree), respectively. Q1 (motivation), Q2 (impression), Q3 (self-confidence), Q4 (liberal arts), Q5 (gender) and Q6 (usefulness)

Although 236 middle and high school students participated in the experiment, there were only four instructors. Thus, the number of instructors, especially if the student to teacher ratio is one to one, may affect the results.

We randomly divided the 236 students into three groups. Thus, the two scenarios (game-based vs. programmable robot) were not compared using the same student. Thus, a difference in a population may affect the results.

7 Conclusion and Future Work

The contributions of the paper are a large-scale comparative experiment using students learning to program via a game-based application with and without programmable robots. Employing either a game-based application with a programmable robot or without a programmable robot affects the motivation to learn and impression of programming. Additionally, there are gender differences. We answer the following RQs:

RQ1: Does using a game-based application and a programmable robot result in a difference in motivation and impression of learning programming?

RQ2: Compared to a game-based application, does using a programmable robot increase the rate of positive responses to Q1 (motivation), Q2 (impression), Q3 (self-confidence), Q4 (liberal arts), Q5 (gender) and Q6 (usefulness) in the survey?

The answer of RQ1 is that differences exist between using a game-based application with and without a programmable robot. The answer of RQ2 is explained in the following: Using a six items questionnaire, the rates of positive responses to the questions about "motivation" to learn programming, "impression" of programming, "self-confidence" when programming, and ability to program by "gender" increase more when using a game-based application with a programmable robot than when using a game-based application alone. However, the increment in positive responses for questions related to liberal art majors and usefulness is larger for a game-based application alone than a game-based application with a programmable robot. We found that employing programmable robots on learning programming did not always give an improvement to all students. In addition, the rate of positive responses to the questions regarding impression of programming and self-confidence when programming increase for boys, but decrease for girls, while the responses to questions related to programming usefulness and type of major show the opposite trend. It is effective for both boys and girls to employ programmable robots on learning programming for impression and self-confidence only.

Thus, we propose that if you employ programmable robots on learning programming, you can give a good impression and self-confidence of programming, and as for motivation, liberal arts, gender and usefulness, you should take account of the effects depends on students' elements, for example gender.

In the future, we will not only show the effects, especially the motivation to learn and the impressions of programming, but also improve the skills of programming by introducing programmable robots to learn programming. Although we dealt with the problems of a standard difficulty in this experiment, we would like to change the difficulties of the problems to deal with in next experiments. As we mentioned in Section 6.3, we think that because there were few scales in this experiment, there are no statically significant differences. To find statistically significant results, we plan to improve the fineness of the scale and conduct further experiments. In addition, we plan to expand the topics related to learning programming via programmable robots.

Acknowledgments

We would like to thank associate professor Hasegawa of the Tokyo Institute of Technology, Ms. Nishino, Ms. Li, and all of students of the Hasegawa Laboratory for their guidance and assistance in developing the Stuffed Teddy Bear Robot. We would also like to thank Kazuki Otomori, Machiko Hattori, and Yuma Matsuura, who belong to Fukazawa Laboratory and Washizaki Laboratory in Waseda University for their help with our research and the teaching assistants of the open campus at Waseda University. This work was supported by JSPS KAKENHI Grant Number 14504538. This work was also supported by Benesse Corporation.

References

Andersen, P. B., Bennedsen, J., Brandorff, S., Caspersen, M. E. & Mosegaard, J. (2003), 'Teaching programming to liberal arts students: A narrative media approach', *SIGCSE Bull.* **35**(3), 109–113.

Barnes, D. J. (2002), Teaching introductory java through lego mindstorms models, *in* 'Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education', SIGCSE '02, ACM, New York, NY, USA, pp. 147–151.

Bezakova, I., Heliotis, J. E. & Strout, S. P. (2013), Board game strategies in introductory computer science, *in* 'Proceeding of the 44th ACM Technical Symposium on Computer Science Education', SIGCSE '13, ACM, New York, NY, USA, pp. 17–22.

Billard, A., Calinon, S., Dillmann, R. & Schaal, S. (2008), Robot programming by demonstration, *in* B. Siciliano & O. Khatib, eds, 'Springer Handbook of Robotics', Springer Berlin Heidelberg, pp. 1371–1394.

Brennan, K. & Resnick, M. (2012), New frameworks for studying and assessing the development of computational thinking, *in* 'Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada', Citeseer.

Cho, V., Cheng, T. & Lai, W. (2009), 'The role of perceived user-interface design in continued usage intention of self-paced e-learning tools', *Computers & Education* **53**(2), 216–227.

DeClue, T. H. (2003), 'Pair programming and pair trading: Effects on learning and motivation in a cs2 course', *J. Comput. Sci. Coll.* **18**(5), 49–56.

Esper, S., Foster, S. R. & Griswold, W. G. (2013), On the nature of fires and how to spark them when you're not there, *in* 'Proceeding of the 44th ACM Technical Symposium on Computer Science Education', SIGCSE '13, ACM, New York, NY, USA, pp. 305–310.

Fagin, B. S., Merkle, L. D. & Eggers, T. W. (2001), Teaching computer science with robotics using ada/mindstorms 2.0, *in* 'Proceedings of the 2001 Annual ACM SIGAda International Conference on Ada', SIGAda '01, ACM, New York, NY, USA, pp. 73–78.

Feldgen, M. & Clua, O. (2004), Games as a motivation for freshman students learn programming, *in* 'Frontiers in Education, 2004. FIE 2004. 34th Annual', pp. S1H/11–S1H/16 Vol. 3.

Jenkins, T. (2001), The motivation of students of programming, *in* 'Proceedings of the 6th Annual Conference on Innovation and Technology in Computer Science Education', ITiCSE '01, ACM, New York, NY, USA, pp. 53–56.

Kelleher, C., Pausch, R. & Kiesler, S. (2007), Storytelling alice motivates middle school girls to learn computer programming, *in* 'Proceedings of the SIGCHI Conference on Human Factors in Computing Systems', CHI '07, ACM, New York, NY, USA, pp. 1455–1464.

Kölling, M. & Henriksen, P. (2005), 'Game programming in introductory courses with direct state manipulation', *SIGCSE Bull.* **37**(3), 59–63.

Kumar, D. & Meeden, L. (1998), 'A robot laboratory for teaching artificial intelligence', *SIGCSE Bull.* **30**(1), 341–344.

Lalonde, J.-F., Hartley, C. & Nourbakhsh, I. (2006), Mobile robot programming in education, *in* 'Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on', pp. 345–350.

Lewis, C. M. (2010), How programming environment shapes perception, learning and goals: Logo vs. scratch, *in* 'Proceedings of the 41st ACM Technical Symposium on Computer Science Education', SIGCSE '10, ACM, New York, NY, USA, pp. 346–350.

Long, J. (2007), 'Just for fun: Using programming games in software programming training and education', *Journal of Information Technology Education: Research* **6**(1), 279–290.

Magnenat, S., Riedo, F., Bonani, M. & Mondada, F. (2012), A programming workshop using the robot " thymio ii " : The effect on the understanding by children, *in* 'Advanced Robotics and its Social Impacts (ARSO), 2012 IEEE Workshop on', IEEE, pp. 24–29.

Mahmoud, Q. H. (2008), Integrating mobile devices into the computer science curriculum, *in* 'Frontiers in Education Conference, 2008. FIE 2008. 38th Annual', pp. S3E–17–S3E–22.

Malan, D. J. & Leitner, H. H. (2007), 'Scratch for budding computer scientists', *SIGCSE Bull.* **39**(1), 223–227.

Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M. & Rusk, N. (2008), 'Programming by choice: Urban youth learning programming with scratch', *SIGCSE Bull.* **40**(1), 367–371.

McNally, M., Goldweber, M., Fagin, B. & Klassner, F. (2006), Do lego mindstorms robots have a future in cs education?, *in* 'Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education', SIGCSE '06, ACM, New York, NY, USA, pp. 61–62.

Nourbakhsh, I. R., Mobile, T., Lab, R. P. & Robots, T. T. (2000), 'Robots and education in the classroom and in the museum: On the study of robots, and robots for study'.

Olivieri, L. M. (2005), 'High school environments and girls' interest in computer science', *SIGCSE Bull.* **37**(2), 85–88.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B. & Kafai, Y. (2009), 'Scratch: Programming for all', *Commun. ACM* **52**(11), 60–67.

Rizvi, M., Humphries, T., Major, D., Jones, M. & Lauzun, H. (2011), 'A cs0 course using scratch', *J. Comput. Sci. Coll.* **26**(3), 19–27.

Sakamoto, K., Takano, K., Washizaki, H. & Fukazawa, Y. (2013), Learning system for computational thinking using appealing user interface with icon-based programming language on smartphones, *in* 'Proceedings of the 21st International Conference on Computers in Education (ICCE)'.

Takase, Y., Mitake, H., Yamashita, Y. & Hasegawa, S. (2013), Motion generation for the stuffed-toy robot, *in* 'SICE Annual Conference (SICE), 2013 Proceedings of', pp. 213–217.

Wing, J. M. (2006), 'Computational thinking', *Commun. ACM* **49**(3), 33–35.