Implementation of Input Methods with Natural Language for Pictogramming

Ishii Mikihiro, Aoyama Gakuin University, Japan
Ito Kazunari, Aoyama Gakuin University, Japan

The IAFOR International Conference on Education – Hawaii 2020
Official Conference Proceedings

Abstract
We previously developed and released "Pictogramming." This application enables users to learn basic programming concepts by creating pictogram content. Pictogramming adopts unique commands for the purpose of educational usage. We have also released several other applications based on Pictogramming to offer various input methods. 1) "Picthon" interprets the Python language. 2) Block Pictogramming constructs programs using visual blocks. 3) Mobile Pictogramming can be used on smartphones. These could be used in class lessons or workshops according to education level and purpose. This paper proposes "Natural Language Pictogramming," which enables users to create content using English sentence input. We conducted practice sessions with this application to evaluate its effectiveness, and herein report on and discuss that evaluation.

Keywords: Pictogramming, pictogram, contents creation, programming, natural language

1. Introduction

A pictogram is a graphic symbol that is used to represent a semantic concept based on the meaning of its shape (Ota, 1993). There are many studies of pictograms in fields such as affective engineering (Ishii & Ito, 2019), education (Ito, 2018b), intercultural communication (Mori, Takasaki, & Ishida, 2009), and semiotics (Hassan, 2017) due to globalization and the rapid worldwide increase in tourism. This is because pictograms are used internationally for symbolic expression.

A pictogram is a graphic sign that should convey meaning reliably. Therefore, the International Organization for Standardization (ISO) mainly deliberates and designs international standards for pictograms, and the appendix for ISO 3864 provides guidelines for the depiction of a human-shaped pictogram (i.e., “human pictogram”). Regardless of a pictogram’s standardization, many unique pictograms can be seen in cities. For that reason, the demand for generating pictograms has increased.

Our research group has been developing “Pictogramming” (“Pictogramming,” 2017), which is a pictogram authoring tool. Pictogramming is based on two words: “pictogram” and “programming,” and it is also used as a programming learning tool for beginners (Ito, 2018a).

Pictogramming has been used by many people from elementary school students to adults to learn programming and to create content in workshops and schools. Pictogramming uses a unique format for commands. We have also released several other applications based on Pictogramming to provide various input methods. 1) “Picthon” can interpret the Python language. 2) “Block Pictogramming” constructs a program with visual blocks. 3) “Mobile Pictogramming” can be used on smartphones. These can be used properly according to the school age and purpose when used in class lessons or workshops. In this paper, we call Pictogramming without an extension “standard Pictogramming.”

We propose "Natural Language Pictogramming" in this paper. It allows users to input Pictogramming's command format with English sentence input. We conducted a practice session with 35 Japanese high school students to confirm the effect of Natural Language Pictogramming. In this paper, we summarize our proposal and discuss the results of the practice session.

2. Pictogramming

2.1 Overview

Pictogramming is a web application that is implemented using HTML5, CSS, and JavaScript. Therefore, Pictogramming is not only easy to access using a Web browser, but also easy to extend.

2.1.1 Screen structure

Figure 1 is a screenshot of the Pictogramming when it is accessed through a PC browser.

Pictogramming is configured using three main areas. In Figure 1, area (A) is the human pictogram display panel, which displays a large human pictogram. Area (B) is the program code description area, and operations on the human pictogram are input and defined in this area. Area (C) is the list of buttons to input support commands, and the description of the program in the program code description area can be completed by clicking the buttons, which aims to reduce the burden of input. Pictogramming also has many buttons. Button (a) is used to run the program that is written in the program code description area. The program also runs when the return key is pressed if box (b) is unchecked. The program code can be saved when button (c) is pressed. The screen image of area (A) can be saved as a PNG image when button (d) is pressed. Animation content can be saved as GIF animation when button (e) is pressed. The saved filename is determined by the value in text field (f).

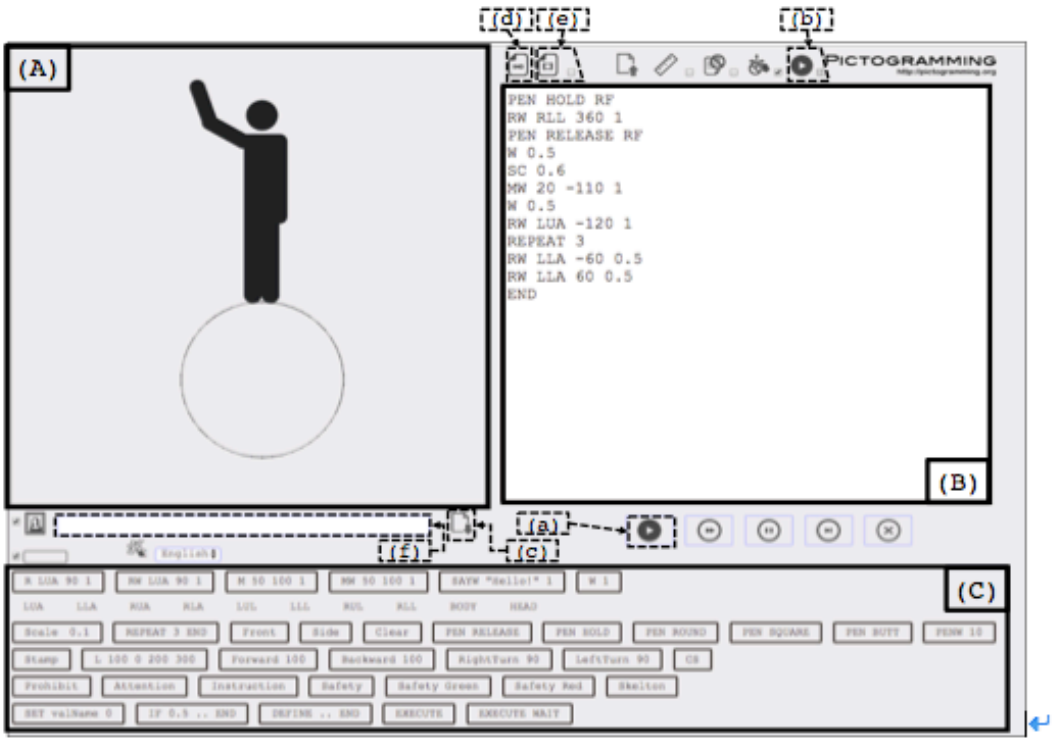

```
PEN HOLD RF
RW RLL 360 1
PEN RELEASE RF
W 0.5
SC 0.6
MW 20 -110 1
W 0.5
RW LUA -120 1
REPEAT 3
RW LLA -60 0.5
RW LLA 60 0.5
END
```

Figure 1: Screenshot of Pictogramming

2.1.2 Command format and content example

Command input in the program code description area follows the format in Figure 2. "Opcode" means operation code. The opcode and arguments are separated by a space.

```
opcode arg1 arg2 ……
```

Figure 2: Command format

Table 1 shows the commands that are mainly used when creating content. By combining the commands in Table 1, pictogram animation content, pictogram graphics content, and combined animation and graphics content as shown in Figure 3 can be created. Pictogram animation content are works that express change in a human pictogram's movement using mainly `R`, `RW`, `M`, and `MW` commands, and pictogram graphic content are works that show the movement path of a human pictogram by mainly using the PEN, and L command. Figure 4 shows sample code that can create content with combined animation and graphics, shown on the right side of Figure 3. Also, a safety sign can be created using the six commands shown from line 16 (P command) to the last line (SR command) in Table 1. Figure 5 shows examples for each mode.

Table 1: Main commands used to create content

Command format	Process
R arg1 arg2 (arg3)	Rotate arg1, a part of the body, arg2 degrees counterclockwise over arg3 seconds. If arg3 is omitted, arg3 is treated as "0".
RW arg1 arg2 (arg3)	Rotate arg1, a part of the body, arg2 degrees counterclockwise over arg3 seconds. The next command is not executed until the movement is complete. If arg3 is omitted, arg3 is treated as "0".
M arg1 arg2 (arg3)	Move arg1 pixels in an X-axis positive direction and arg2 pixels in a Y-axis positive direction with linear uninform motion over arg3 seconds. If arg3 is omitted, arg3 is treated as "0".
MW arg1 arg2 (arg3)	Move arg1 pixels in an X-axis positive direction and arg2 pixels in a Y-axis positive direction with linear uninform motion over arg3 seconds. The next command is not executed until the movement is complete. If arg3 is omitted, arg3 is treated as "0".
W arg1	Wait arg1 seconds. The next command is not executed until the wait is complete.
PEN arg1 (arg2)	Release pen if arg1 is "RELEASE." Hold pen if arg1 is "HOLD." A part of body can be set as arg2 to set the location of releasing or holding the pen only if arg1 is "RELEASE" or "HOLD". If arg2 is omitted, it is considered to be set to "BODY".
PENW arg1	Set width of pen to arg1. (Initial state is "1.")
L arg1 arg2 arg3 arg4	Draw line from coordinate (arg1, arg2) to coordinate (arg3, arg4).
REPEAT arg1	Execute until END command arg1 times.
END	End of REPEAT statement block.
SC arg1	Change scale to arg1 time(s) from current size.
FD	Switch the human pictogram to the front direction. (Initial state is front direction.)
SD	Switch the human pictogram to the side direction.
SK	Change to skeleton mode if current mode is not Skeleton mode. Change to normal mode if current mode is Skeleton mode.
P	Change to prohibit mode.
A	Change to attention mode.
I	Change to instruction mode.
S	Change to safety mode.
SG	Change to safety green mode.
SR	Change to safety red mode.

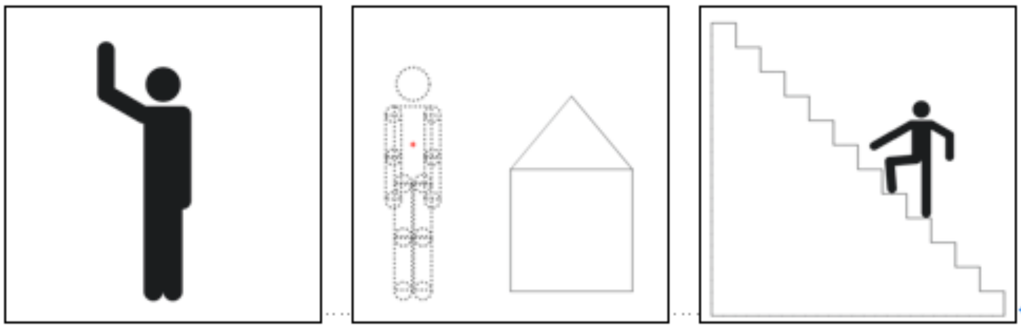


Figure 3: Pictogram animation content (left side), pictogram graphics content (center), and combined animation and graphics content (right side)

```
SK
MW -300 -250 0.5
PEN HOLD
MW 0 600 0.5
MW 600 0 0.5
REPEAT 12
MW 0 -50 0.2
MW -50 0 0.2
END
PEN RELEASE
SC 0.5
MW 430 240 0.5
SK
R LC -85 0.6
R LK 90 0.6
R LS -60 0.6
R RS 60 0.6
R RE -60 0.6
```

Figure 4: Code for the right side of Figure 3

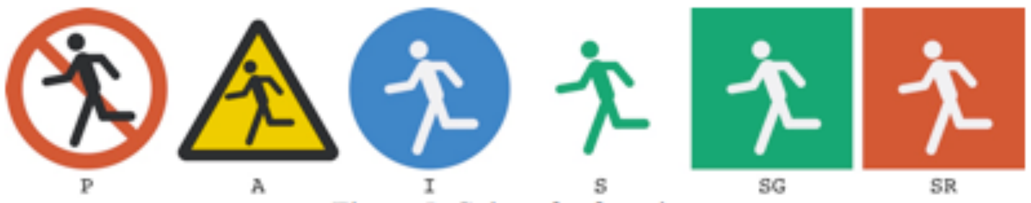


Figure 5: Color of safety sign

Next, the names of the parts that are used in the commands for the human pictogram are described. The movement decisions and the status of the human pictogram are determined by designating body parts or joints. The left and right sides of Figure 6 show the names of the body parts for a human pictogram facing front. The left and right sides of Figure 7 show the names of body parts for a human pictogram facing to the side.

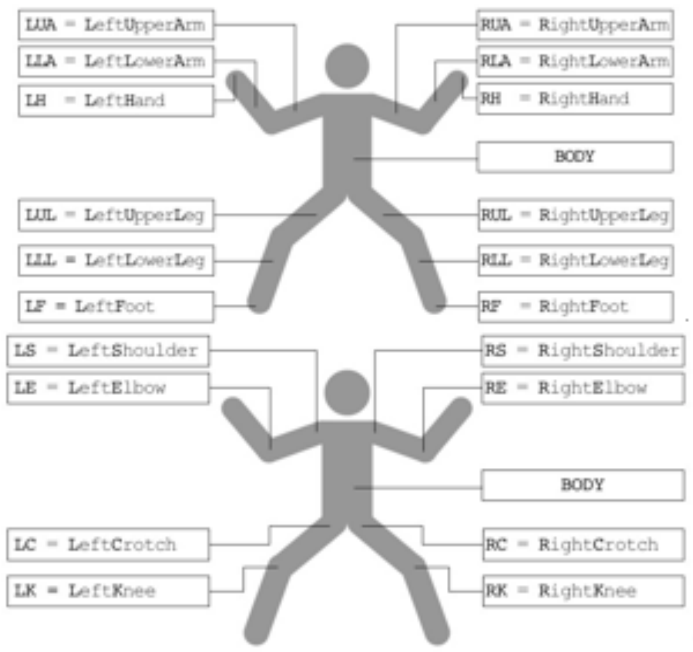


Figure 6: Names of body parts and joints (front direction)

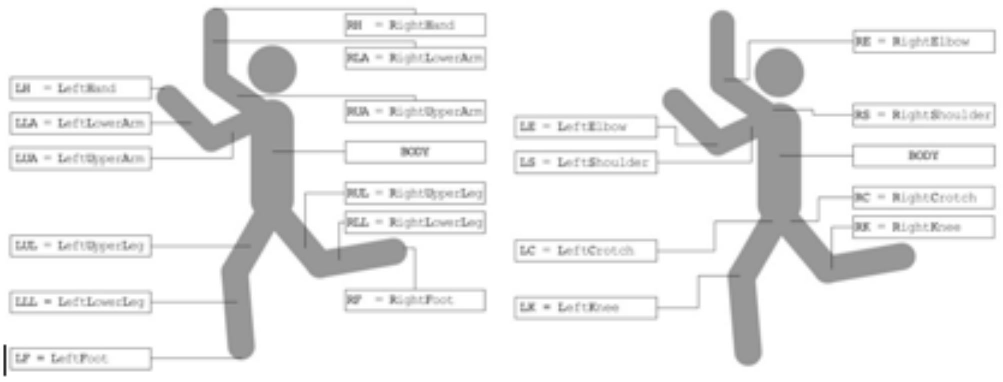


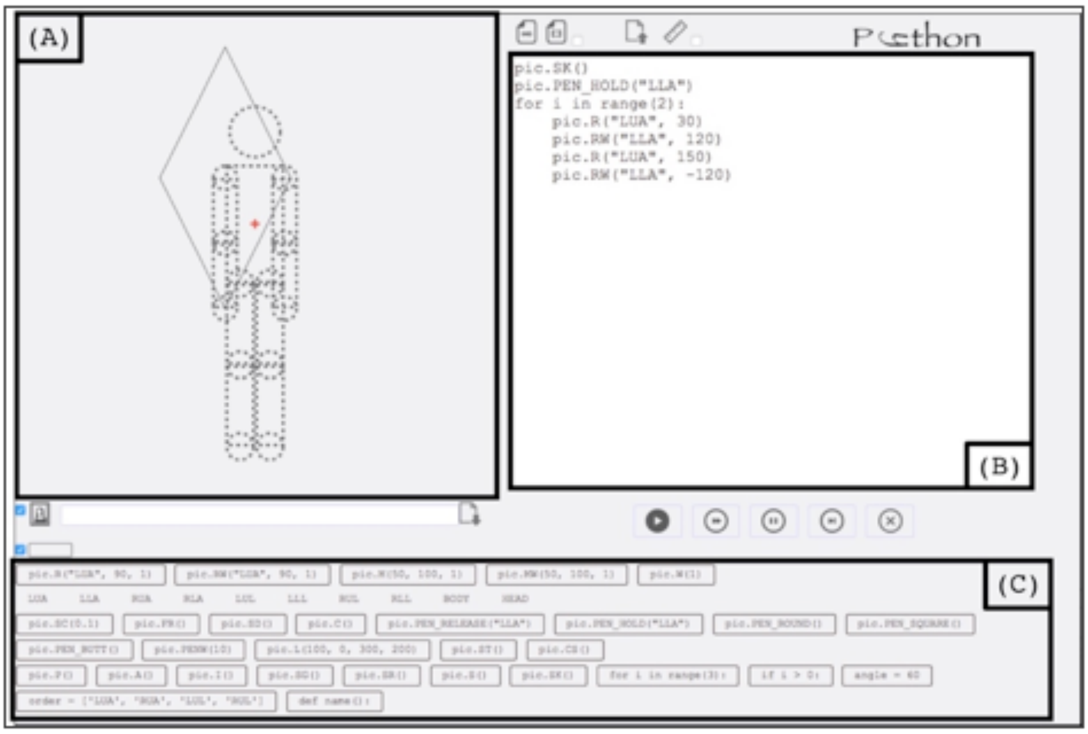
Figure 7: Names of body parts and joints (side direction)

2.2 Pictogramming series

We have developed extensive applications. To distinguish between applications, we named this application “standard Pictogramming,” as mentioned in Section 2.1.

2.2.1 Picthon

Picthon (Ito Lab., 2019a) is a version of Pictogramming, in which content using a human pictogram can be created in Python. Figure 8 shows a screenshot of Picthon.



```
pic.SK()
pic.PEN_HOLD("LLA")
for i in range(2):
    pic.R("LUA", 30)
    pic.RW("LLA", 120)
    pic.R("LUA", 150)
    pic.RW("LLA", -120)
```

Figure 8: Screenshot of Picthon

The interface is almost the same as the standard Pictogramming, and the placement of the three main areas (Figure 8 (A), (B), and (C)) is the same as in the standard Pictogramming. The only difference between Picthon and the standard Pictogramming is that the codes written in the program description area (Figure 8 (B)) are written in Python. Naturally, the text string that is input by clicking on the buttons for supporting command input is a Python program string. Figure 9 shows an example of a program in Picthon.

```
pic.SK()
pic.PEN_HOLD("LLA")
for i in range(2):
    pic.R("LUA", 30)
    pic.RW("LLA", 120)
    pic.R("LUA", 150)
    pic.RW("LLA", -120)
```

Figure 9: Sample code for Picthon

2.2.2 Block Pictogramming

Block Pictogramming ("Block Pictogramming," 2019b) is a block-type visual programming version for Pictogramming. Figure 10 is a screenshot of Block Pictogramming. The interface is almost the same as standard Pictogramming, and two out of the three main areas (Figure 10 (A) and (B)) shown are the same as in standard Pictogramming. In addition, Area (B) is a drawing by Google Blockly. Google Blockly is a library to help develop block typed programming language (Google Developers, n.d.). It is open-source product, therefore lots of research projects which uses Blockly has been reported (Marron, Weiss, & Wiener, 2012). There is no Area (C)–the list of buttons for supporting command input—because the blocks play the same role as the buttons, and the commands are input by selecting a block from Area (B'), which is the list of blocks. The difference from standard Pictogramming is that the program is constructed using blocks. Figure 11 shows an example of a program string in Block Pictogramming.

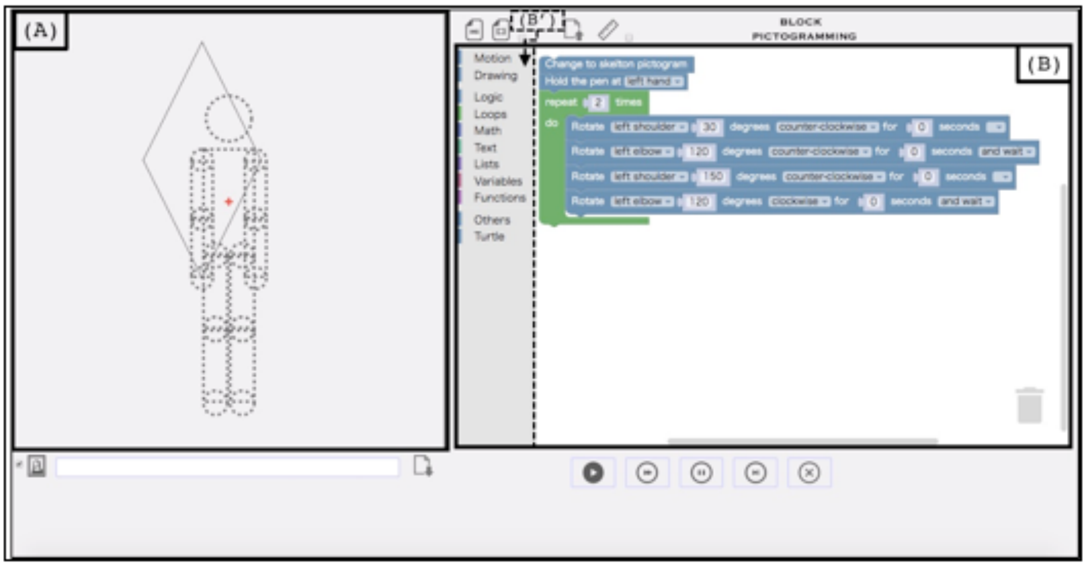
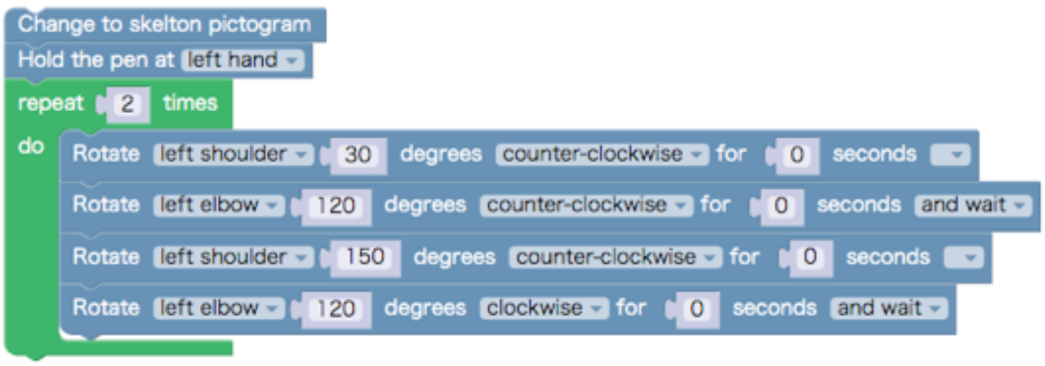


Figure 10: Screenshot for Block Pictogramming

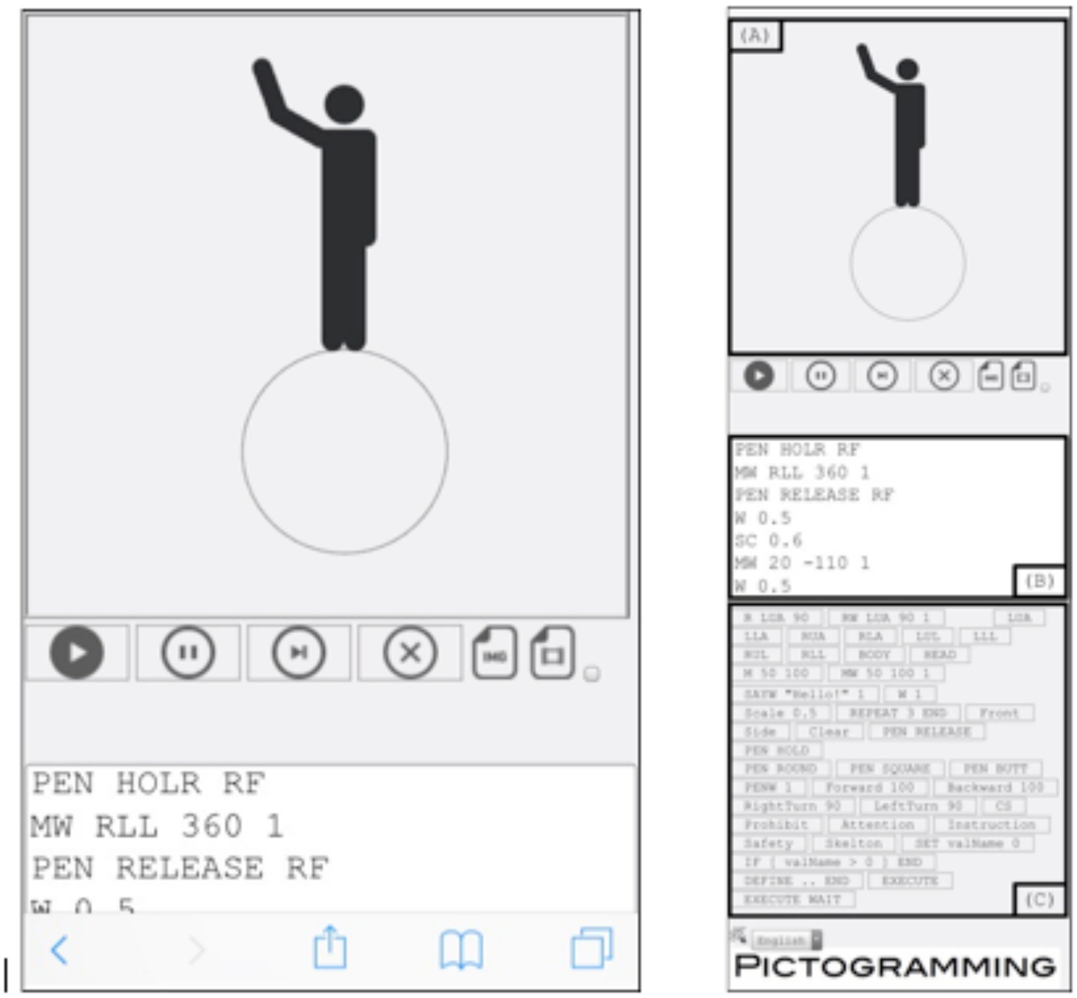


```
Change to skelton pictogram
Hold the pen at left hand
repeat 2 times
do  Rotate left shoulder 30 degrees counter-clockwise for 0 seconds
    Rotate left elbow 120 degrees counter-clockwise for 0 seconds and wait
    Rotate left shoulder 150 degrees counter-clockwise for 0 seconds
    Rotate left elbow 120 degrees clockwise for 0 seconds and wait
```

Figure 11: Sample code for Block Pictogramming

2.2.3 Mobile Pictogramming

Mobile Pictogramming ("Mobile Pictogramming," 2018) is smartphone version of Pictogramming. It is easy to adapt to a smartphone because it is web application implemented with HTML5, CSS, and JavaScript (Ito, 2019). On the left side of Figure 12 is a screenshot of the Mobile Pictogramming application from a browser on an iOS device. However, the three main areas ((A): the human pictogram display panel; (B): the program code description area; and (C): the list of buttons for supporting commands input) are arranged vertically due to the differences in the interface and display area from a PC (see the right side of Figure 12).



```
PEN HOLR RF
MW RLL 360 1
PEN RELEASE RF
W 0.5
```

```
PEN HOLR RF
MW RLL 360 1
PEN RELEASE RF
W 0.5
SC 0.6
MW 20 -110 1
W 0.5
```

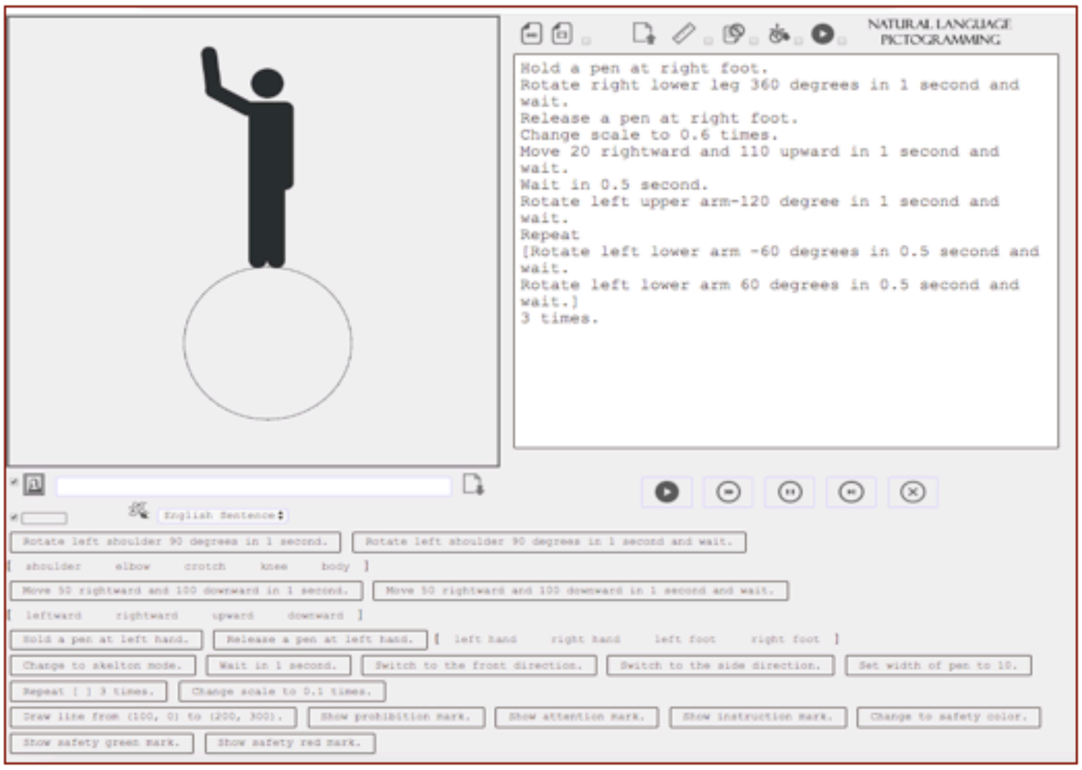
Figure 12: Screenshot of Mobile Pictogramming

The screen layout is optimized for a smartphone, but all commands that are shown in Table 1 of Section 2.1.2 can be used. Of course, the content can be saved as text, PNG image, and as GIF animation in the same way as with standard Pictogramming.

3. Natural Language Pictogramming

3.1 Screen structure

Figure 13 is a screenshot of Natural Language Pictogramming using a PC browser. The interface is almost the same as standard Pictogramming.


NATURAL LANGUAGE PICTOGRAMMING
English Sentence
Rotate left shoulder 90 degrees in 1 second.
Rotate left shoulder 90 degrees in 1 second and wait.
[shoulder elbow crotch knee body]
Move 50 rightward and 100 downward in 1 second.
Move 50 rightward and 100 downward in 1 second and wait.
[leftward rightward upward downward]
Hold a pen at left hand.
Release a pen at left hand.
[left hand right hand left foot right foot]
Change to skelton mode.
Wait in 1 second.
Switch to the front direction.
Switch to the side direction.
Set width of pen to 10.
Repeat [ ] 3 times.
Change scale to 0.1 times.
Draw line from (100, 0) to (200, 300).
Show prohibition mark.
Show attention mark.
Show instruction mark.
Change to safety color.
Show safety green mark.
Show safety red mark.

```
Hold a pen at right foot.
Rotate right lower leg 360 degrees in 1 second and
wait.
Release a pen at right foot.
Change scale to 0.6 times.
Move 20 rightward and 110 upward in 1 second and
wait.
Wait in 0.5 second.
Rotate left upper arm-120 degree in 1 second and
wait.
Repeat
[Rotate left lower arm -60 degrees in 0.5 second and
wait.
Rotate left lower arm 60 degrees in 0.5 second and
wait.]
3 times.
```

Figure 13: Screenshot of Natural Language Pictogramming

3.2 Example of command input

Natural Language Pictogramming supports input using natural sentences for all the commands in Table 1 in Section 2.1.2. Table 2 shows the command format in Natural Language Pictogramming. The buttons for the supporting command input show an example sentence using the command format in Table 2. Figure 14 shows the display with the buttons for supporting command input. The end of a command is expressed by adding a "." (period) on the termination of the command, as shown in Table 2, or by inserting “return” on the termination of the command. Both "." and the “return” are available.

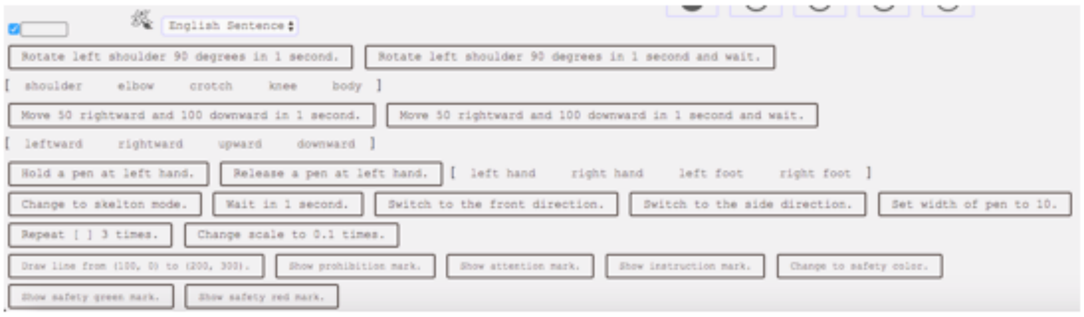

English Sentence
Rotate left shoulder 90 degrees in 1 second.
Rotate left shoulder 90 degrees in 1 second and wait.
[shoulder elbow crotch knee body]
Move 50 rightward and 100 downward in 1 second.
Move 50 rightward and 100 downward in 1 second and wait.
[leftward rightward upward downward]
Hold a pen at left hand.
Release a pen at left hand.
[left hand right hand left foot right foot]
Change to skelton mode.
Wait in 1 second.
Switch to the front direction.
Switch to the side direction.
Set width of pen to 10.
Repeat [ ] 3 times.
Change scale to 0.1 times.
Draw line from (100, 0) to (200, 300).
Show prohibition mark.
Show attention mark.
Show instruction mark.
Change to safety color.
Show safety green mark.
Show safety red mark.

Figure 14: Buttons for supporting command input in Natural Language Pictogramming

Table 2: Command input format using natural sentences

Commands format in standard Pictogramming	Input way in Natural Language Pictogramming
`R arg1 arg2 (arg3)`	`Rotate arg1 arg2 degrees in arg3 seconds.`
`RW arg1 arg2 (arg3)`	`Rotate arg1 arg2 degrees in arg3 seconds and wait.`
`M arg1 arg2 (arg3)`	`Move arg1 leftward and arg2 downward in arg3 second.`
`MW arg1 arg2 (arg3)`	`Move arg1 leftward and arg2 downward in arg3 seconds and wait.`
`W arg1`	`Wait in arg1 seconds.`
`PEN arg1 (arg2)`	`arg1 a pen at arg2.`
`PENW arg1`	`Change scale to arg1 times.`
`L arg1 arg2 arg3 arg4`	`Draw line from (arg1, arg2) to (arg3, arg4).`
`REPEAT arg1` *The command repeated* `END`	`Repeat` [*The command repeated*] `arg1 times.`
`SC arg1`	`Change scale to arg1 times.`
`FD`	`Switch to the front direction.`
`SD`	`Switch to the side direction.`
`SK`	`Change to skeleton mode.`
`P`	`Show prohibition mark.`
`A`	`Show attention mark.`
`I`	`Show instruction mark.`
`S`	`Change to safety color.`
`SG`	`Show safety green mark.`
`SR`	`Show safety red mark.`

Various movements in human pictograms can be constructed using the buttons for supporting command input shown in Figure 14 because the buttons in Natural Language Pictogramming show the unit such as second and degree. For instance, when a user who cannot understand the command format in standard Pictogramming thinks "I want to make human pictogram move 50 pixels leftward and 10 pixels upward in 1 second," the person can input command by referring the button "`Move 50 leftward and 100 downward in 1 second.`"

Natural Language Pictogramming also supports an input format other than the display of the buttons for supporting command input. For instance, word order, and another expression that is different from those of the buttons for supporting command input are available as input formats. Table 3 shows an example of some input formats that is different from the buttons for supporting command input, so other word orders and expressions are also available.

Table 3: Input examples using natural sentences

The display buttons for support command input	Input examples other that the display buttons for supporting command input
`Rotate left upper arm 90 degrees in 1 second.`	・`Rotate left shoulder 90 degrees to clockwise in 1 second.` ・`Rotate left upper arm -90 degrees to anticlockwise in 1 second.`
`Rotate left upper arm 90 degrees in 1 second and wait.`	・`Rotate left shoulder 90 degrees to clockwise in 1 second and wait.` ・`Rotate left upper arm -90 degrees to anticlockwise in 1 second.`
`Move 50 leftward and 100 downward in 1 second.`	・`Move 100 downward and 50 leftward in 1 second.` ・`Move -50 rightward and -100 upward in 1 second.`
`Move 50 leftward and 100 downward in 1 second and wait.`	・`Move 100 downward and 50 leftward in 1 second and wait.` ・`Move -50 rightward and -100 upward in 1 second and wait.`
`Wait in 1 second.`	・`Wait for 1 second.` ・`Wait 1 second.`

3.3 The processing procedure

The input sentences are converted to the command format in Table 1 of Section 2.1.2 and are executed. The summary of the conversion procedure is presented from «Step 1» to «Step 5».

«Step 1» The text strings that were input are separated by periods to divide the text strings into sentences. The conversion is done per sentence.
«Step 2» The words for operation code, unit (degree, second, and so on), and digit (value of second, value od degree, and so on) are extracted from the sentences that were input. The words for unit, number, and their locations are also identified.
«Step 3» The sentence that was input is converted to the command format of standard Pictogramming based on the results of «Step 2».
«Step 4» In case of the `REPEAT` command, the conversion in «Step 3» is not done correctly because `REPEAT` commands are input using "`[ ]`" (brackets). Therefore, the locations of the brackets are identified. Based on the locations, the word order could be switched.
«Step 5» The commands are run based on the results until «Step 4» is reached.

3.4 Overview of the practice

3.4.1 Purpose

The practice was carried out to survey whether it was easy to use the input in Natural Language Pictogramming. Participants created real content by using the implementation that was released. Then, we asked them to answer the questionnaire.

3.4.2 The participants

The practice involved 35 fourth-grade students from Kobe University Secondary School (equivalent to the first grade of senior high school). The participants had learned standard Pictogramming in the year 2018 and learned Picthon in November 2019. The practice was done on November 25 (Monday) in a classroom. All lessons were 50 minutes long.

In addition, the practice only allowed the participants to use basic commands for pictogram content creation because we wanted to evaluate the ease of use solely for sentence input. Therefore, the practice did not allow the participants to use the `L` command or to allow them to create complex content.

3.4.3 Procedure

Table 4 shows a summary of the practice procedure. In the "Content" column in Table 4, the contents which was done in the class is ordered in time. In the practice, the handout was used. The handout gave three exercises to the participants. The code as first exercise is shown in Figure 15, and second exercise is shown in Figure 16. The participants make free contents as third exercise.

Table 4: Timetable of the lesson

Content	Time (minutes)
・ Greeting	1
・ Startup computers, and guide to web page	3
・ Explain the summary for Natural Language Pictogramming	2
・ Distribute handouts	2
・ Explain application overview by using the buttons for supporting command input	2
・ Practice to make programs shown in the handout	30
・ Answer the questionnaire	10

```
Rotate left upper arm -120 degrees in 0.8 second and wait.
Repeat
[Rotate left lower arm -60 degrees in 0.4 second and wait.
Rotate left lower arm 60 degrees in 0.4 second and wait.]
3 times.
```

Figure 15: Code as first exercise

```
Move 40 leftward and 100 upward in 1 second and wait.
Change to skeleton mode.
Hold a pen at right foot.
Rotate right upper leg 360 degrees in 1 second.
```

Figure 16: Code as second exercise

3.5 Practice results and consideration

This section presents the results from an analysis of the practice. The analysis was based on a questionnaire evaluation, observing the participants while they were creating, and data that were logged while participants were creating.

The questionnaire asked the participants about four things that are shown in Table 5. From question 1 to question 3, six options were offered. The options were: 6-very easy to use; 5-easy to use; 4-if anything, easy to use; 3-if anything, hard to use; 2-hard to use; and 1-very hard to use. In question 4, the format of the answer was open ended.

Table 5: Content of the questionnaire

#	Content
1	Please evaluate the usability of input by natural sentences, which was implemented this time.
2	Please evaluate the usability of input by Picthon, which you used this month.
3	Please evaluate the usability of input by words with a separator, which you used in the previous school year.
4	Open ended: Input by natural sentences is available this time. Please describe your opinions and feelings about this function.

First, Table 6 shows the results of Q.1 to Q.3. The mean value for Natural Language Pictogramming is more than the mean value for standard Pictogramming, but it is less than the mean value for Picthon. Also, a T-test was done based on the null hypothesis that there was no difference between Natural Language Pictogramming and standard Pictogramming, but no significant difference was found (p = 0.3789).

Table 6: Results from Q.1 to Q.3

	1	2	3	4	5	6	mean	SD
Q.1 (for Natural Language Pictogramming)	0	0	12	10	7	6	4.200	1.106
Q.2 (for Picthon)	0	0	3	16	10	6	4.543	0.886
Q.3 (for standard Pictogramming)	0	4	7	9	11	4	4.114	1.143

Second, Figure 17 shows excerpts of answers to Q.4.

The results of the analysis based on the questionnaire evaluation, participant observation, and the data log is described below.

There were answers similar to A), B), and C), saying that they felt the input by English sentences was easy to understand because it did not use abbreviations, and everyone could create content even if they did not know the command format or the abbreviations for the command name. Then, even if the movement of the human pictogram differed from their thinking, there were many events that could be solved by discussion in the participants. The participants confirmed whether there is no error about the program grammar each other. The program grammar is English sentence, so confirmation was done easily. Therefore, it was thought that content creation was easier using input in natural sentences to a certain degree.

There were answers such as D), saying that they felt that learning a programming language was important using natural sentence input. Therefore, using Natural Language Pictogramming before standard Pictogramming and Picthon may generate learning effectiveness.

On the other hand, there were answers such as E), F), and G), saying that Natural Language Pictogramming was troublesome because it required input in long sentences. In practice, there were many participants who were not used to typing English, so it seemed difficult for them to input using English.

A) Thank you for teaching about official names. I did not know the meaning of the abbreviations. Natural Language Pictogramming was fun. It took a long time to read the content of the buttons, but it was easier to understand the meanings in Natural Language Pictogramming than in Picthon or standard Pictogramming. I was also able to learn English vocabulary.
B) Natural Language Pictogramming is good because I can easily imagine the movement of a human pictogram. If I could get used to inputting English words, I think I could use it more easily.
C) Natural Language Pictogramming is very easy to use. When I used Picthon before, I had to know about programming grammar. In Natural Language Pictogramming, I can input commands using only knowledge of English. In addition, I have the opportunity to learn English by using English sentences.
D) When I used standard Pictogramming and Picthon, I had no idea of the significance of learning a programming language. However, it was difficult to input English, so I was able to learn about the importance of a programming language through Natural Language Pictogramming.
E) The input method when I used Picthon was easier than this software.
F) This lesson was difficult for me.
G) The input using Picthon was easier than using English because English sentences are long and complicated. However, I thought programming in English was better for learning English words.

Figure 17: Excerpts of answers to Q.4

4. Conclusion

In this study, we proposed Natural Language Pictogramming, which constructs codes from natural language for pictogram content creation. We reported on the evaluation of a practice session but analyzed it only from the viewpoint of usability. Therefore, we plan to conduct further practice sessions to compare Natural Language Pictogramming with standard Pictogramming and Picthon from the viewpoint of learning effects.

References

Google Developer. (n.d.). Retrieved from https://developers.google.com/blockly

Hassan, E. M. M. (2017). The semiotics of pictogram in the signage systems. International Design Journal, 5.

Ishii, M. & Ito, K. (2019). Presence of motion lines in human pictograms: Analyses and evaluations. The International Association of Societies of Design Research (IASDR) 2019.

Ito, K. (2018a). Pictogramming – Programming learning environment using human pictograms. IEEE Global Engineering Education Conference (EDUCON 2018). 134–141.

Ito, K. (2018b). Pictogramming: Learning environment using human pictograms based on constructionism. Constructionism 2018. 592–599.

Ito, K. (2019). Mobile Pictogramming. International conference on ubiquitous information management and communication (IMCOM). 547–553.

Pictogramming (2017). Retrieved from https://pictogramming.org/editor/

Mobile Pictogramming (2018). Retrieved from https://pictogramming.org/editor/sp.html

Pichon (2019a). Retrieved from https://pictogramming.org/editor/picthon.html

Block Pictogramming (2019b). Retrieved from https://pictogramming.org/editor/block_en.html

Marron, A., Weiss, G., & Wiener, G. (2012). A decentralized approach for programming interactive applications with JavaScript and blockly. Proceedings of the 2nd edition on Programming systems, languages and applications based on actors, agents, and decentralized control abstractions (AGERE! 2012). 59–70.

Mori, Y., Takasaki, T., & Ishida, T. (2009). Patterns in pictogram communication. Proceedings of the 2009 International Workshop on Intercultural Collaboration (ACM). 277–280.

Ota, Y. (1993). Pictogram design. Tokyo: Kashiwa Bijutsu Shuppan.

Contact email: kaz@si.aoyama.ac.jp
c8119001@aoyama.jp