International Journal of Control and Automation
Vol. 10, No. 1 (2017), pp.421-430
http//dx.doi.org/10.14257/ijca.2017.10.01.38

Analysis of Mutual Language Intelligibility of the Programming Tool Used in Robot Programming Learning

Se-min Kim[1], Chang-su Ryu[2] and Choong Ho Lee[3*]

[1]Graduate School of Information Communication Engineering,
Hanbat National University
[2]Department of Cartoon & Animation, Yewon Arts University
[3]Graduate School of Information Communication Engineering, Hanbat National University
[1]imsim303@hotmail.co.kr, [2]twin4me@hotmail.com, [3]chlee@hanbat.ac.kr

Abstract

In the Republic of Korea, several policies and studies are underway to organize a software curriculum into a regular course. In this study, we compared and analyzed several programming tools available in the robot programming learning activities of the software curriculum. Programming tools utilized in this study were MSRDS VPL and App Inventor and these were performed in a class by utilizing robots. As students can easily learn the foreign language similar to their mother tongue, this study examined how easily students were able to learn another programming language with a similar interface.

Keywords: *Software Education, MSRDS, App Inventor, Block Type Programming*

1. Introduction

The 21st century we live has entered a knowledge-oriented society due to the development of ICT skills. The developed various information-based technologies and application technologies have applied to many fields [1], and the development of the information communication technology such as wireless internet, mobile internet devices and the Internet of Things has given an impact on an educational environment. Such a society pursues the human who is not standardized by monolithic education but thinking creatively about what is happening in the world and able to actively respond to changes in a given environment. Therefore, new education which can cultivate a creative problem-solving ability on the basis of creativity is required, not the former education doing such as rote learning and cramming education [3]. In the Republic of Korea, the information curriculum of elementary and secondary education was renamed as the software curriculum, and the curriculum was revised to teach software based on Computational Thinking. Universities are seeking changes for carrying out software education for all students. In particular, Ministry of Science, ICT and Future Planning is working for training software-talented individuals by supporting 11 billion won for six years as a supporting business of software-oriented university at the national level [4]. In this study, we compared MSRDS VPL and App Inventor with C language in class and we used robots as educational materials. This study tried to find out the mutual language intelligibility among languages.

2. Related Research

2.1. Preceding Research

C. J. Park (2015) analyzed the relationship among the abstract thinking, language intimacy, and study achievement about the understanding of C and Scratch program. Park

ISSN: 2005-4297 IJCA

showed that C and Scratch were main languages which the students of the study had experienced before, and the language intimacy of Java or Visual Basic affected students' abstract thinking skills positively.

S. W. Seo (2010) compared the text-based programming languages with the visual-based programming languages. Seo concluded that the text-based programming languages affected the improving thinking ability of science of information more meaningfully than the visual-based programming languages [6].

2.2. Programming Tools

This study has investigated the keyword, mutual language intelligibility among several programming tools. It is easy to learn a foreign language which is very similar mutually. MSRDS (MicroSoft Robotics Developer Studio) is developing tools and an environment that help to facilitate the development of various robotic applications if professional workers or ordinary people have general knowledge of robot programming. Figure 1 and Figure 2 shows a programming result as if there were robot equipment [7].

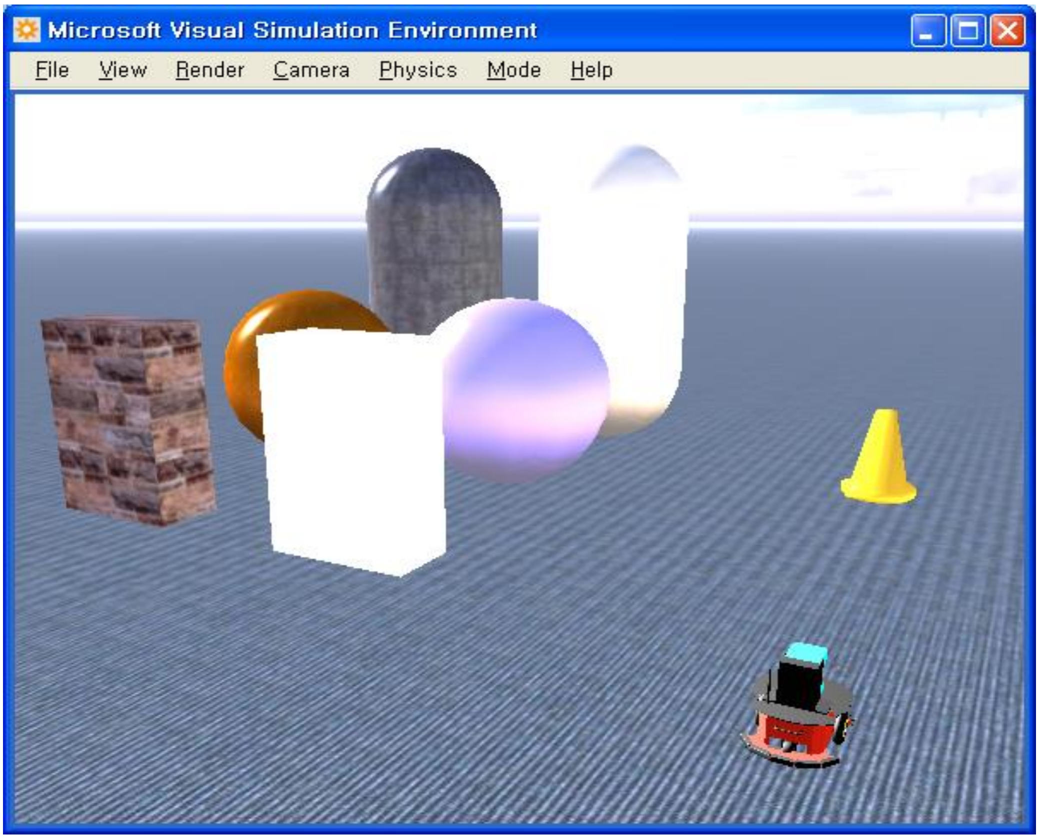


Figure 1. Robotics Simulation (Keyboard Control)

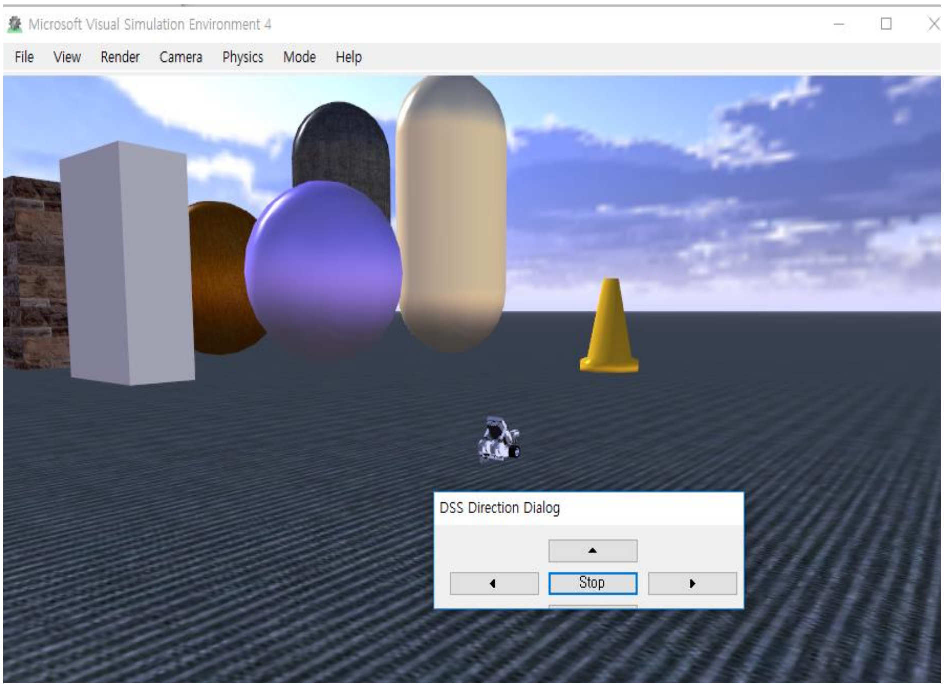

Microsoft Visual Simulation Environment 4
File View Render Camera Physics Mode Help
DSS Direction Dialog
Stop

Figure 2. Robotics Simulation (Direction Dialog Control)

VPL (Visual Programming Language) helps people do programming by connecting lines between icon-based shapes named Activity as shown in Figure 3 and Figure 4. Also, VPL helps easily to understand the basic concepts of programming and very usefully to grasp the principles of the programming.

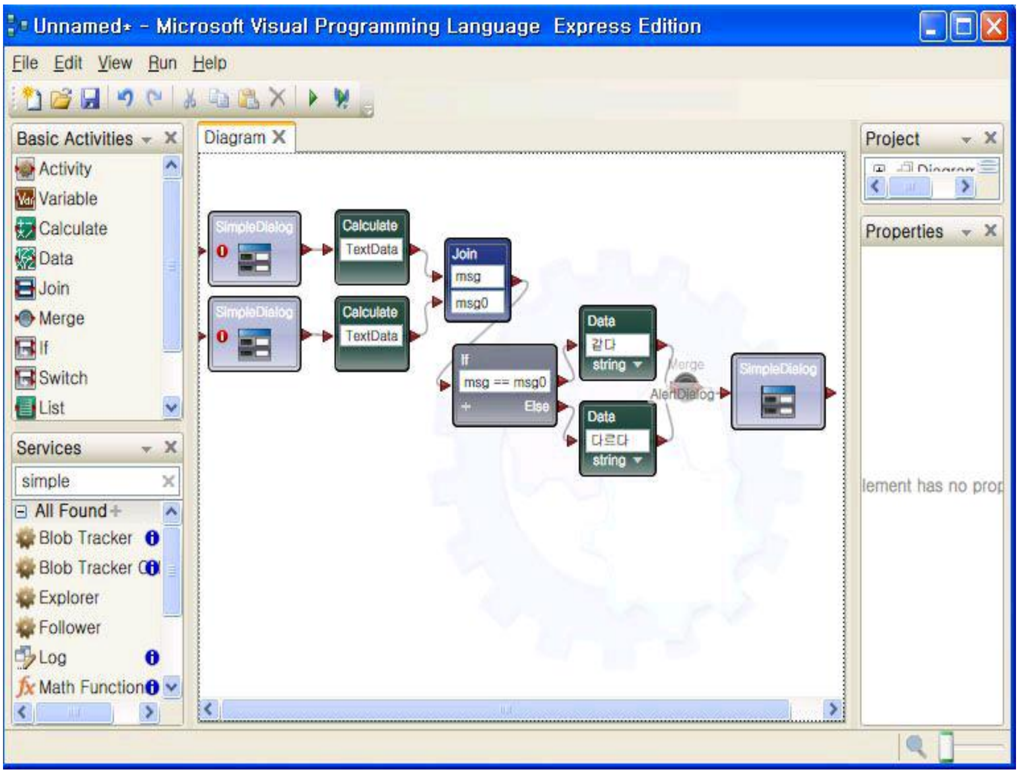


Figure 3. Source Code Programming by VPL (Level Low)

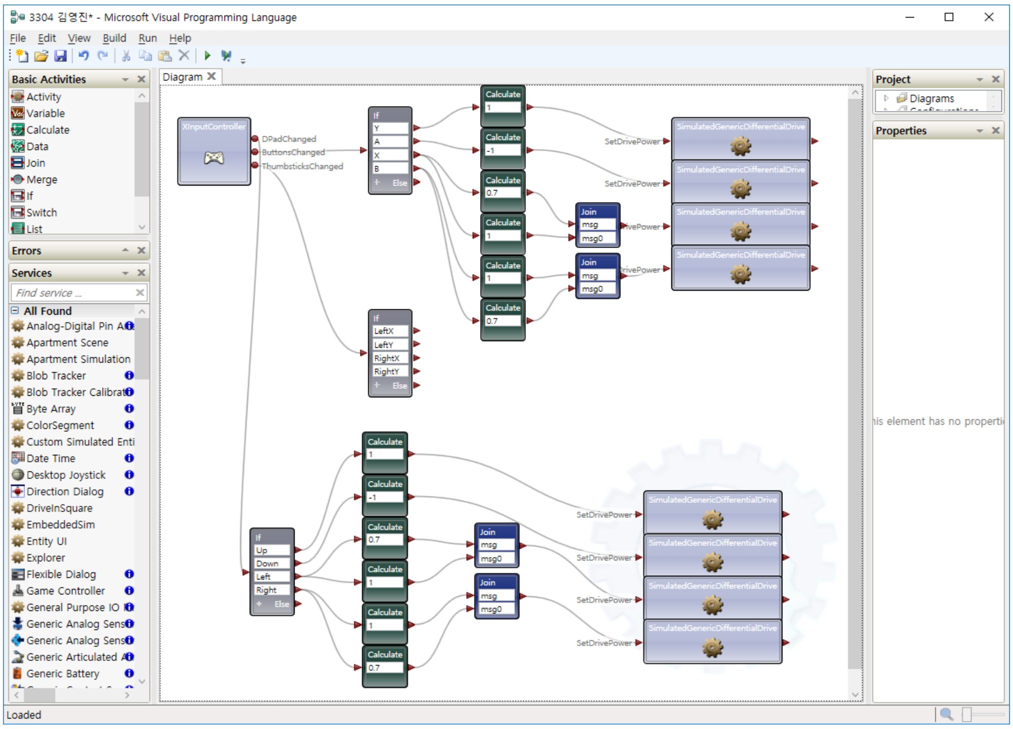


Figure 4. Source Code Programming by VPL (Level High)

App Inventor was a project to provide a developing environment to replace the existing programming languages in order to make it easy to develop applications that run on Android operating system [8]. As Figure 5 and Figure 6, App Inventor helps designing like an environment similar to smart phones, can take advantage of the sensor of smart phones, and can program with blocks as shown in Figure 7 and Figure 8.

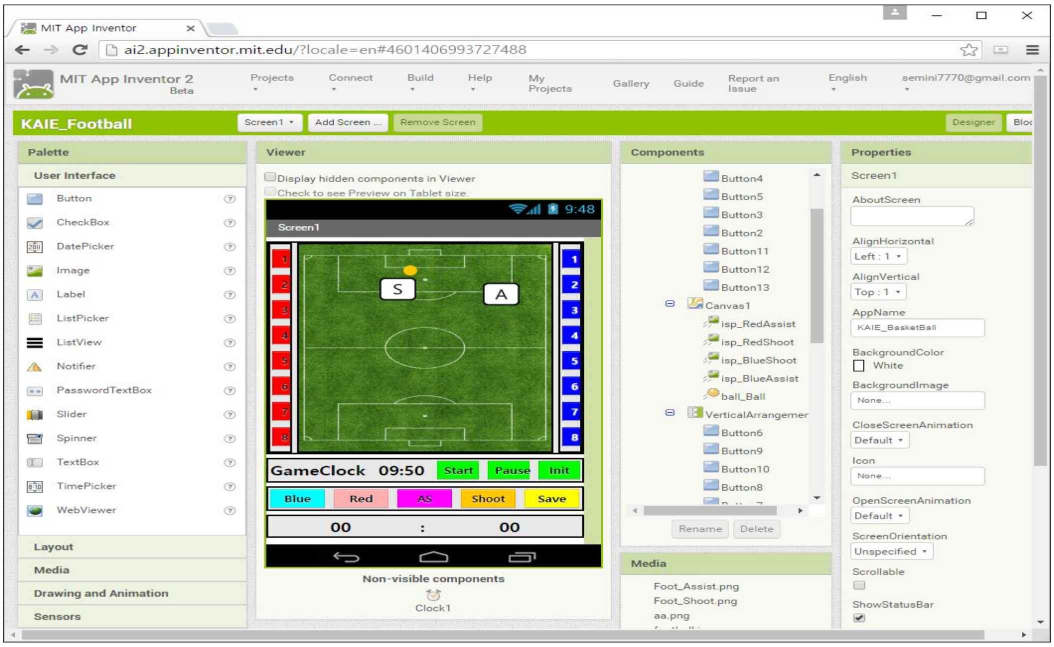
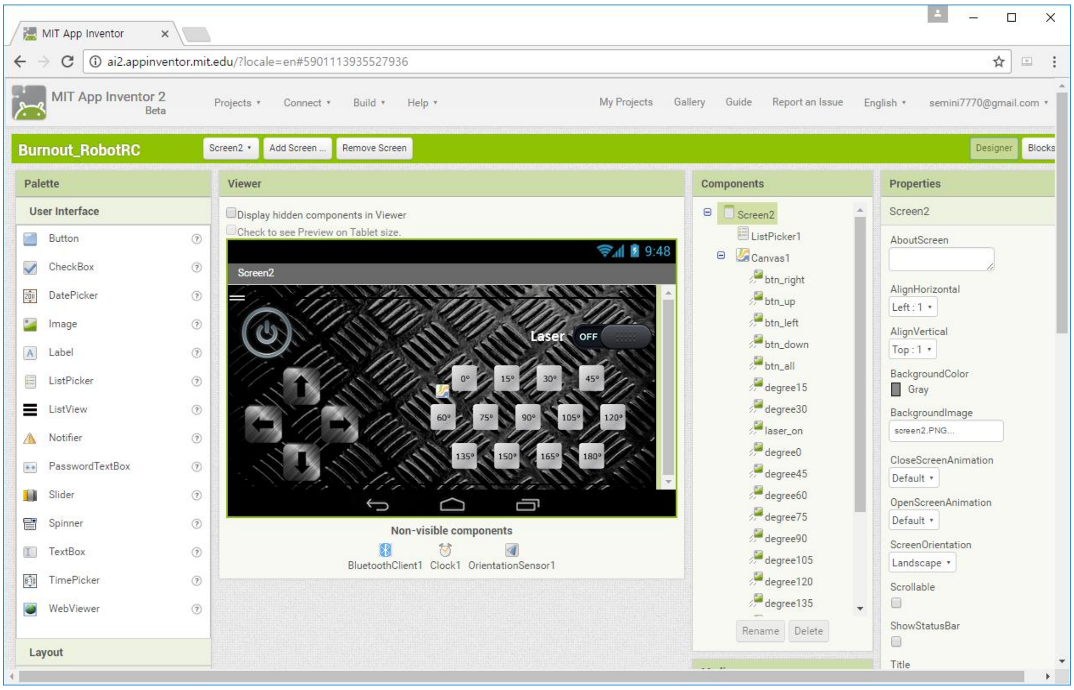


Figure 5. Design Screen of App Inventor (Level Low)



AboutScreen

AlignHorizontal: Left : 1

AlignVertical: Top : 1

BackgroundColor: Gray

BackgroundImage: screen2.PNG...

CloseScreenAnimation: Default

OpenScreenAnimation: Default

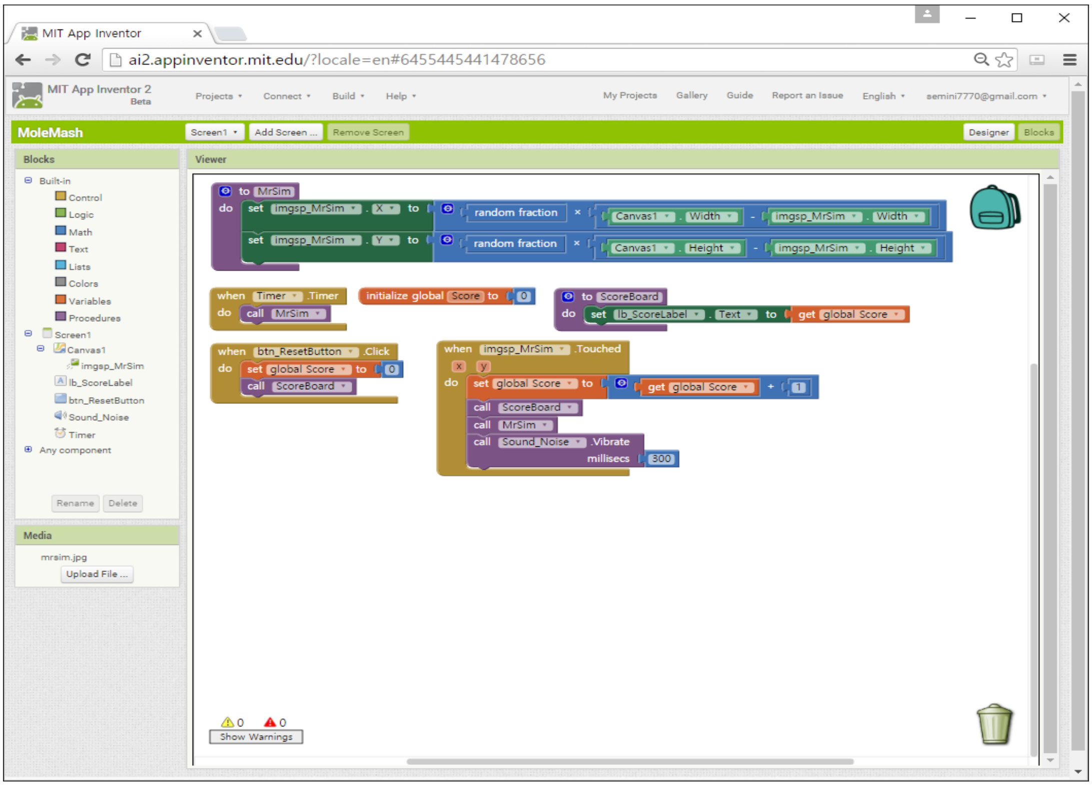
ScreenOrientation: Landscape

Scrollable: □

ShowStatusBar: □

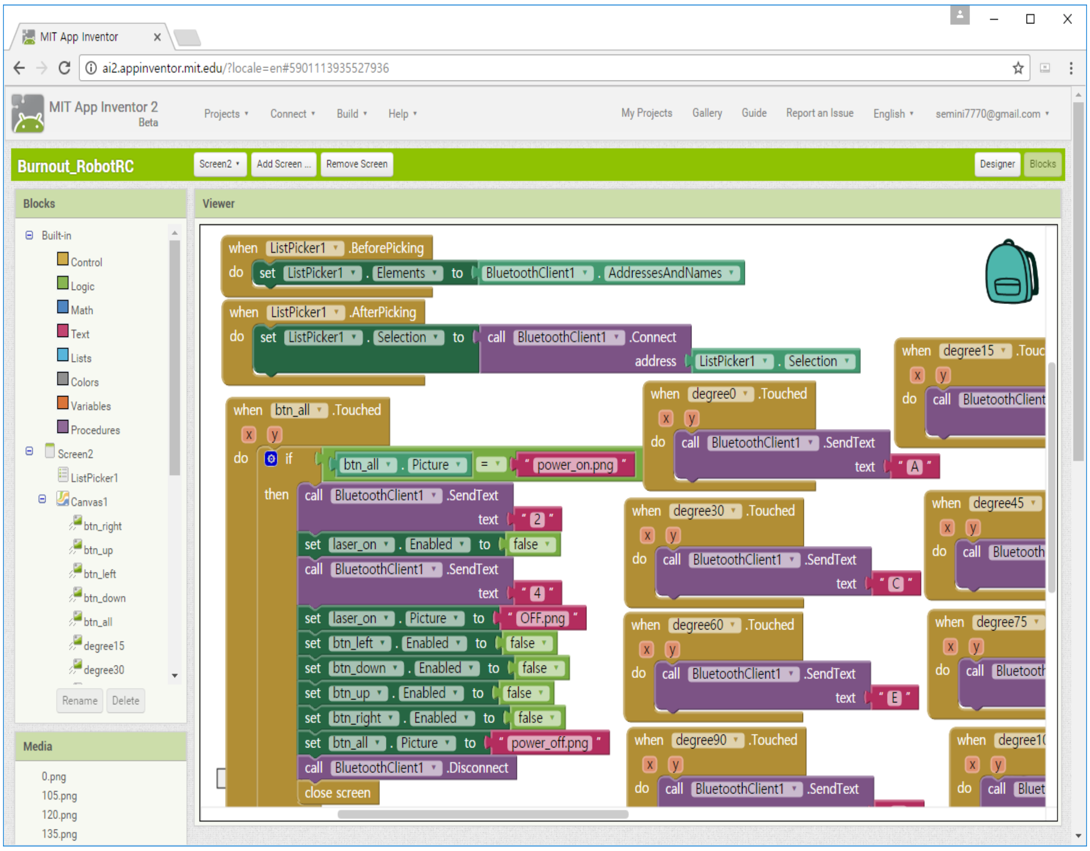
Title

Figure 6. Design Screen of App Inventor (Level High)



```
to MrSim
do  set imgsp_MrSim . X to  random fraction × Canvas1 . Width - imgsp_MrSim . Width
    set imgsp_MrSim . Y to  random fraction × Canvas1 . Height - imgsp_MrSim . Height

when Timer .Timer
do  call MrSim

initialize global Score to 0

to ScoreBoard
do  set lb_ScoreLabel . Text to  get global Score

when btn_ResetButton .Click
do  set global Score to 0
    call ScoreBoard

when imgsp_MrSim .Touched
  x  y
do  set global Score to  get global Score + 1
    call ScoreBoard
    call MrSim
    call Sound_Noise .Vibrate
                 millisecs 300
```

Figure 7. Block Programming Screen of App Inventor (Level Low)



```
when ListPicker1 .BeforePicking
do set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames

when ListPicker1 .AfterPicking
do set ListPicker1 . Selection to call BluetoothClient1 .Connect
                                   address ListPicker1 . Selection

when btn_all .Touched
 x y
do if btn_all . Picture = " power_on.png "
   then call BluetoothClient1 .SendText
                    text " 2 "
        set laser_on . Enabled to false
        call BluetoothClient1 .SendText
                    text " 4 "
        set laser_on . Picture to " OFF.png "
        set btn_left . Enabled to false
        set btn_down . Enabled to false
        set btn_up . Enabled to false
        set btn_right . Enabled to false
        set btn_all . Picture to " power_off.png "
        call BluetoothClient1 .Disconnect
        close screen

when degree0 .Touched
 x y
do call BluetoothClient1 .SendText
                 text " A "

when degree15 .Touc
 x y
do call BluetoothClient

when degree30 .Touched
 x y
do call BluetoothClient1 .SendText
                 text " C "

when degree45
 x y
do call Bluetooth

when degree60 .Touched
 x y
do call BluetoothClient1 .SendText
                 text " E "

when degree75
 x y
do call Bluetoot

when degree90 .Touched
 x y
do call BluetoothClient1 .SendText

when degree1(
 x y
do call Bluet
```

Figure 8. Block Programming Screen of App Inventor (Level High)

3. Main Title

3.1. Subject of Study

The subjects of this study were composed of 20 second year students of A technical high school and 20 second year students of B technical high school. All of the subjects were learning C language in other classes in the semester. The students of A technical high school learned App Inventor first and MSRDS later. The students of B technical high school learned MSRDS VPL first and then App Inventor. Their performances were evaluated every week with a given mission. In addition, the results of programming were commonly confirmed by a LEGO Mindstorm robot.

2.2. Lesson Plan

The Subjects were taught following the lesson plan like Table 1 and Table 2. “A course” was the plan before a mid-term examination, and “B course” was the plan before a final examination. In school A and B, “A course” and “B course” proceeded only by changing the order.

Table 1. Lesson Plan (A Course)

A Course	Contents
1st week	MSRDS VPL basics
2nd week	basic Activity
3rd week	robotic simulation
4th week	variable
5th week	operator
6th week	input-output statement
7th week	control statement
8th week	Mid-term exam

Table 2. Lesson Plan (B Course)

B Course	Contents
9th week	App Inventor basics
10th week	design
11th week	sensor utilization
12th week	variable
13th week	operator
14th week	input-output statement
15th week	control statement
16th week	Final exam

4. Robot Programming Application in Classes and Result Analysis

In this study, the units which had no matching points between both programming tools were excluded in the analysis. However, the performance assessment scores of students were measured for learning. The important result of this study was the part with similarities of the two programming languages and with the direct relationship of programming Logic. This corresponds to variables, operators, input-output statements, and control statements. The learning results of units would be analyzed as shown in Table 3 and Table 4.

The Students of A technical high school learned App Inventor first and then MSRDS. In general, the understanding of App Inventor has an average of 4% or higher. Because App Inventor was the programing language mostly similar to the interface of Scratch and Entry which was in recent trend, App Inventor might be familiar to the eyes of students and influenced mutual language intelligibility. Particularly in the section of variable of MSRDS VPL, there was the most gap, 4% deviation of performance assessment because the procedure for setting variables in MSRDS VPL was more difficult than App Inventor. Therefore, if the use of programming language tools is difficult, the mutual language intelligibility will be lower.

The students of B school learned MSRDS VPL first and then learned App Inventor. In general, the understanding of App Inventor was averaging 6% or higher. Like A school students, there was the most deviation (3%) in the variable section. However, A school's average percentage of mission performance was 87% and B school's 81%. It can be said that A school's students who learned App Inventor first accepted another language more easily than B school's students who learned MSRDS first, and A school's students had a higher mutual language intelligibility than B school's students.

Table 3. Lesson Result of A School

Week	A School mission performer/total (percentage)
MSRDS VPL basics	25/25 (100%)
basic Activity	24/25 (96%)
robotic simulation	24/25 (96%)
variable	20/25 (80%)
operator	23/25 (92%)
input-output statement	20/25 (80%)
control statement	21/25 (84%)
App Inventor basics	25/25 (100%)
design	24/25 (96%)
sensor utilization	20/25 (80%)
variable	24/25 (96%)
operator	23/25 (92%)
input-output statement	21/25 (84%)
control statement	21/25 (84%)

Table 4. Lesson Result of B School

Week	B School mission performer/total (percentage)
MSRDS VPL basics	24/25 (96%)
basic Activity	23/25 (92%)
robotic simulation	22/25 (88%)
variable	19/25 (80%)
operator	20/25 (80%)

input-output statement	20/25 (76%)
control statement	19/25 (76%)
App Inventor basics	24/25 (96%)
design	24/25 (92%)
sensor utilization	20/25 (80%)
variable	22/25 (88%)
operator	21/25 (84%)
input-output statement	20/25 (80%)
control statement	21/25 (84%)

4. Conclusions and Recommendations

This paper studied the mutual language intelligibility between two visual programming languages and following results was obtained. First, because of App Inventor's similarity of structure to Scratch, App Inventer has an advantage in the mutual language intelligibility rather than MSRDS VPL. Second, a somewhat complicated procedure of setting like variables setting in SRDS VPL can affect the mutual language intelligibility. Third, after learning the programming language with a higher mutual language intelligibility than other existing programming languages, the learning can be helpful to study next programming languages. Although this study compared languages among icon or block-based programming languages, subsequent studies need to investigate the mutual language intelligibility of the existing C language, Basic, Java, etc.

References

[1] D. K. Kim, H. C. Lee, Y. W. Lee and S. Y. Shin, "Instructor's Smart Learning Acceptance : Focusing on TAM Model", Journal of the Institute of Information and Communication Engineering. Vol 20, no 6, (2016), pp 1081-1086.

[2] S. M. Chung, J. H. Choi and J. W. Park, "Design of Software Quality Evaluation Model for IoT", Journal of the Institute of Information and Communication Engineering, Vol 20, no 7, (2016), pp. 1342-1354.

[3] J. S. Lee, "Effects of LOGO Programming Language on Elementary School Students' Creativity", Jeonju National University of Education, M.A. (2008).

[4] K. S. Oh and S. J. An, "A Study on Development of Educational Contents about Computational Thinking", Vol 19, no 2, (2016), pp. 11-20.

[5] C. J. Park, J. S. Hyun, H. R. Jin and H. S. Chung, "Relation Analysis of High School Students' Abstract Thinking, Language Familiarity and Academic Achivement of C and Scaratch Program Understanding", The Journal of Korean Assosiation of Computer Education, Vol 18, no 6, (2015), pp. 1-12.

[6] S. W. Seo, D. S. Nam and T. W. Lee, "The Effect of Computational Thinking Ability Using Text-base vs. Visual-base Programming Language on Robot Programming Learning", Proceedings of 26th Korean Association of Computer Education, Seoul, Korea, (2010), August 15-16.

[7] T. G. Yang and H. Y. Lee, "Visual Programming Language's Basic of Microsoft Robotics Developer", C&Robot, Daejoen Korea, (2008).

[8] S. J. An and Y. J. Lee, "Elementary and Secondary Programming Education Plan Using App Inventor", The Journal of Korean Association of Computer Education, Vol 17, no 5, (2014), pp. 79-88.

Authors

Se-Min Kim received his B.S. and M.S. Degrees from the Department of Computer Education at Woosuk University, Wanju, South Korea, in 2003 and 2006, respectively. He has been a Ph.D. Candidate in Graduate School of Information Communication Engineering from Hanbat National University, Daejeon, South Korea since 2014. He started working at New Vision, CO. LTD, in 2008, where he now works as a senior researcher. His current research interests include Computer Education, Robot Education, STEAM and Mobile Application.

Chang-Su Ryu received his M.S and Ph.D. Degrees in Computer Education, Computer Engineering from Mokwon University, Daejeon, Korea in 2006 and 2013. He is currently a Professor of the Department of Cartoon and Animation in Yewon Arts University. His research interests include Animation, Mobile 3D Game, Computer Graphic and Robotics.

Choong Ho Lee received his B.E. and M.E. Degrees ins Electronic Engineering from Yonsei University, Seoul, Korea, in 1985 and 1987, respectively. He also received his Ph.D. Degree in Information Science from Tohoku University, Sendai, Japan in March of 1998. From 1985 to 2000, he was with KT as a researcher. Since 2000, he has been a Professor in Graduate School of Information Communication Engineering, Hanbat National University. His current research interests include pattern recognition, digital image processing and mobile robot control.