KIPS Tr. Software and Data Eng.
Vol.5, No.2 pp.107~116 pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2016.5.2.107

A Visual Programming Environment on Tablet PCs to Control Industrial Robots

Park Eun Ji[†] · Seo Kyeong Eun[††] · Park Tae Gon[†††] · Sun Duk Han[††††] · Cho Hyeonjoong[†††††]

ABSTRACT

Industrial robots have been usually controlled using text-based programming languages provided by each manufacturer with its button-based TP(Teaching Pendent) terminal. Unfortunately, when we consider that people who manipulate TPs in manufacturing sites are mostly unskilled with no background knowledge about computer programming, these text-based programming languages using button-based interaction on manufacturing sites are too difficult for them to learn and use. In order to overcome the weaknesses of the text-based programming language, we propose a visual programming language that can be easily used on gesture-enabled devices. Especially, in our visual programming environment, each command is represented as a block and robots are controlled by stacking those blocks using drag-and-drop gestures, which is easily learnable even by beginners. In this paper, we utilize a widely-spread device, Tablet PC as the gesture-enabled TP. Considering that Tablet PC has limited display space in contrast to PC environments, we designed different kinds of sets of command blocks and conducted user tests. Based on the experiment results, we propose an effective set of command blocks for Tablet PC environment.

Keywords : Visual Programming Language, Robot Programming

산업용 로봇 제어를 위한 태블릿 PC 기반의 비주얼 프로그래밍 연구

박 은 지[†] · 서 경 은[††] · 박 태 곤[†††] · 선 덕 한[††††] · 조 현 중[†††††]

요 약

산업용 로봇 제어는 버튼 기반의 TP(Teaching Pendant) 단말기를 통해 제조사가 제공하는 텍스트 기반의 프로그래밍 언어를 사용하여 이루어진다. 그러나 제조 현장에서 TP를 조작하는 사용자가 주로 언어의 배경지식이 없는 비전공자임을 고려할 때, 버튼 기반의 TP에서의 텍스트 기반의 프로그래밍 언어는 학습과 사용의 어려운 단점이 있다. 텍스트 기반의 프로그래밍 언어의 단점을 극복하기 위해 비숙련 사용자들도 쉽게 프로그래밍을 할 수 있는 그래픽 기반의 비주얼 프로그래밍 환경이 제안되었으며, 특히 명령 블록을 사용한 비주얼 프로그래밍 환경은 드래그 앤 드롭 기반으로 조립할 수 있는 블록 형태의 명령어를 지원해주므로 초보자가 쉽게 프로그래밍을 할 수 있다. 본 논문에서는 태블릿 PC를 로봇 제어를 위한 TP로 활용하고, 이를 기반으로 블록을 이용한 비주얼 프로그래밍 개발 환경을 제공하여 비숙련 사용자도 드래그 앤 드롭으로 쉽게 프로그래밍하는 환경을 제안한다. 또한, 사용 환경이 PC 환경 대신, 한정된 디스플레이 공간을 가지고 있는 태블릿PC 환경에 적용된 점을 고려하여 총 3가지의 서로 다른 명령어의 범위를 포함하고 있는 블록을 디자인하여 사용자 실험을 진행했다. 실험 결과를 바탕으로 한정된 디스플레이 공간인 태블릿 PC 환경에서의 효과적인 명령 블록들을 제안한다.

키워드 : 비주얼 프로그래밍 언어, 로봇 프로그래밍

1. 서 론

최근 연구 결과에 의하면 기계가공 산업에서 산업용 로봇이 3%의 낮은 점유율을 가지고 있는 이유 중 하나가 로봇 프로그래밍의 기술 부족이라고 언급된 바 있다[1]. 이로 인해 비전공자들을 위한 프로그래밍 교육 문제가 대두되고 있다.

기존의 TP(Teaching Pendant)는 산업용 로봇을 제어하는 기기로서, 로봇 프로그래밍 환경을 제공한다. 즉 TP에는 사용자 조작을 통해 특정 지점으로의 로봇구동을 제어하는 인터페이스가 내장되어 있다. 통상 TP의 인터페이스로 물리 버튼들이 제공되고, 제조사가 제공하는 텍스트 기반의 프로그램 언어를 지원한다. 이는 오늘날 향상된 사용자 인터페이스 기술에 비해 상대적으로 직관성이 떨어진다고 판단된

※ 이 논문은 중소기업청에서 시행하는 산학연협력 기술개발사업 (Q1430514)지원을 받아 수행되었음.
† 비 회 원 : 고려대학교 컴퓨터정보학과 석사과정
†† 비 회 원 : 고려대학교 컴퓨터정보학과 박사과정
††† 비 회 원 : ㈜프레스토솔루션 부사장
†††† 비 회 원 : ㈜프레스토솔루션 수석
††††† 종신회원 : 고려대학교 컴퓨터정보학과 부교수
Manuscript Received : August 24, 2015
First Revision : January 11, 2016
Accepted : January 11, 2016
* Corresponding Author : Hyeonjoong Cho(raycho@korea.ac.kr)

다. 또한 로봇 상태와 프로그램상태를 제공해 주는 상태표시 창은 TP의 디스플레이 공간이 한정되어[2] 사용자가 복잡한 프로그래밍을 하는데 있어서 제한적이다.

이를 보완하기 위해 본 논문에서는 텍스트 기반의 프로그래밍을 대신하여 그래픽 기반의 비주얼 프로그래밍(Visual Programming, VP) 환경을 제안한다. 그래픽 기반의 VP 환경은 기존 텍스트 언어들을 사용하여 프로그래밍 하는 것과 다르게 이해하기 쉬운 도형, 아이콘 등을 이용하여 프로그래밍을 하는 것으로 일반 사용자들에게 향상된 직관성을 제공하여 프로그래밍의 소요되는 시간을 대폭 줄이는 것을 목표로 한다[3]. 특히 제안하는 VP 환경은 조립이 가능한 가상 블록 형태의 명령어를 지원해주므로 초보자도 쉽게 드래그 앤 드롭으로 프로그래밍을 할 수 있도록 한다. VP 환경을 지원하기 위해 태블릿 PC를 이용함으로써 기존 TP 디스플레이의 한계를 보완할 수 있는 환경을 제공한다. 태블릿 PC는 매우 정확하고 높은 해상도의 터치스크린을 제공하는 장치로서 드래그 앤 드롭과 같은 제스처 기반의 제어로 산업 환경에서 간단한 프로그래밍 과정을 수행하기에 매우 적합하다[1].

로봇 제어를 위해 태블릿 PC를 사용하는 것은 기존 TP의 단점을 보완해줄 수 있으나 일반적으로 PC에서 사용되었던 VP 환경을 그대로 태블릿 PC에 적용시키는 것은 적절하지 않다. 따라서 본 논문에서는 총 3가지의 서로 다른 유연성을 가지는 명령 블록들을 디자인하였다. 디자인한 환경은 사용자 실험을 통해 비교분석하였으며, 실험 결과를 바탕으로 산업 환경에서 사용하기 적절한 명령 블록을 제안한다.

2. 관련 연구

비숙련 사용자를 위한 교육용 프로그래밍 언어는 비주얼 프로그래밍 언어와 텍스트 기반 프로그래밍 언어로 나눌 수 있다[4]. 그 중에서도 VP 언어는 초등교육에 사용될 정도로 비숙련 사용자도 쉽게 학습할 수 있는 언어로 알려져 있다.

VP 언어의 대표적인 예로 초등교육 수준에서 사용할 수 있는 Scratch, Etoy Alice가 있고, 중등교육 수준에서의 NXT-G, Robolab 그리고 고등교육 수준에서 사용할 수 있는 NXT-MSRDS, Webot, Sinulink, 그리고 MORPHA, LabView, Blockly 등이 소개된 바 있다[4]. 앞서 다양한 제조사에서 출시한 VP 언어가 존재함을 언급하였지만, 본 논문에서는 산업용 로봇 제어를 위한 VP 환경으로 적합한 Scratch, Blockly, MDRDS, LabView와 같은 VP 언어를 중심으로 관련 연구를 논의하고자 한다[5-8].

MSRDS(Microsoft Robotics Developer Studio)는 로봇 시뮬레이션을 위한 개발 도구로써, Fig. 1과 같은 Microsoft의 MSRDS 환경을 기반으로 프로그래밍 환경을 제공한다[9]. 그러나 본 소프트웨어는 윈도우 계열 운영체제를 필요로 한다는 점과 MSRDS는 무료이나 개발을 위한 도구는 유료라는 단점이 있다[9]. 또한 하드웨어에 내장하여 동작할 경우에는 무거운 구조의 프레임워크가 필요하며 별도의 런타임 비용을 지불하여야 한다.

LabView는 텍스트 기반의 언어와 달리 여러 가지 문법적인 사항들을 없애고 블록 다이어그램이라 불리는 흐름도를 사용하여 프로그램을 생성한다. LabView는 그림 기호(Graphical Symbol)을 사용하기 때문에 프로그래밍 경험이 전혀 없는 사람이라도 LabView를 쉽게 배울 수 있다[10]. LabView는 Fig. 2와 같은 환경을 사용하여 프로그램을 아이콘과 선(Wire)을 이용해 그래픽적으로 작성하므로 텍스트 언어에 비해 직관성을 띠며, 대화적이다. 하지만 텍스트 언어에 비해 실행 코드가 좀 더 커진다는 단점이 있다[10].

Scratch는 2007년 MIT 미디어랩(Media Lab)의 Lifelong Kindergarten Group에서 개발된 교육용 프로그래밍 언어이다. Scratch의 가장 큰 특징은 블록 쌓기 방식으로 프로그래밍을 하는 것으로, 다양한 색상으로 구분된 블록들을 원하는 순서대로 가져다가 쌓기만 하면 되는 시각적인 프로그래밍 방식이다. Fig. 3은 Scratch의 실제 개발 환경 및 코드 예제를 보여주고 있다[12].

Scratch는 스테이지(Stage), 스프라이트(Sprite), 스크립트(Script)로 구성된 비주얼 기반 교육용 프로그래밍 언어로 학습자의 흥미와 접근성을 높인다. 또한 Scratch의 각 블록(Block)이 문법적으로 오류가 없어야만 서로 결합될 수 있도록 설계되어 구문 오류가 발생하지 않는다. 따라서 프로그래밍 오류를 근본적으로 줄일 수 있다.

Blockly는 그래픽 프로그래밍 기반의 언어이다. Scratch가 호스트 머신에서 Smalltalk 가상 머신의 실행을 필요로 하는 것에 반해 Blockly는 Javascript로 온전히 구현되었으며 웹(Web)에 이식하기도 훨씬 용이하다. 개발자의 입장에서 보면, Blockly는 간단한 JavaScript 문법을 사용하므로 새로운 그래픽적인 기능 블록을 쉽게 사용할 수 있도록 해준다. Blocky는 Fig. 4와 같이 블록(Block)에서 코드를 생성하여 JavaScript나 Python으로 변환해주는 문법을 가지고 있고, 그 이후에 일반 프로그램 파일로 실행할 수 있도록 해준다[13].

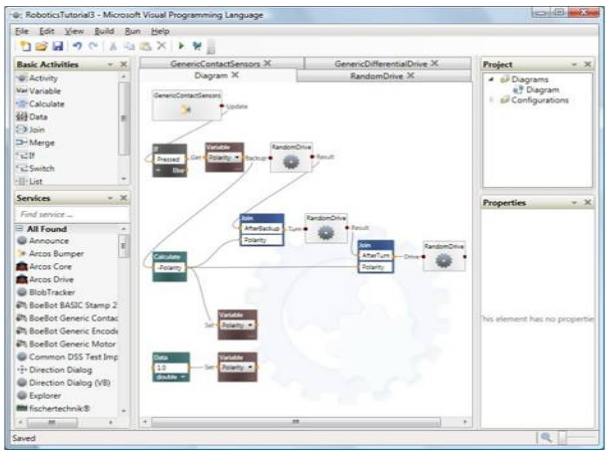


Fig. 1. MSRDS development environment & example code[1)]

1) 출처: July 25, 2008. MSRDS VPL 기초따라하기 [On-line], Available: http://updownme.tistory.com/5

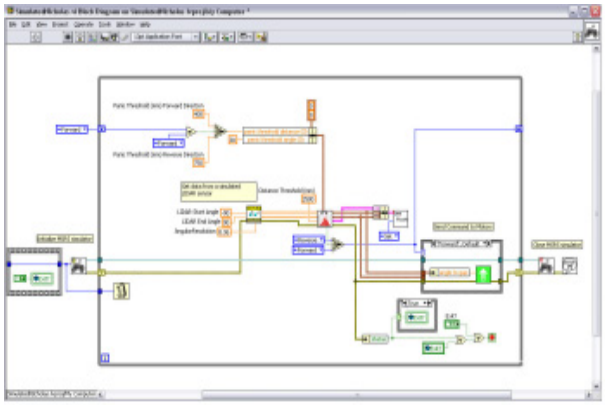


Fig. 2. LabView development environment & example code[2)]

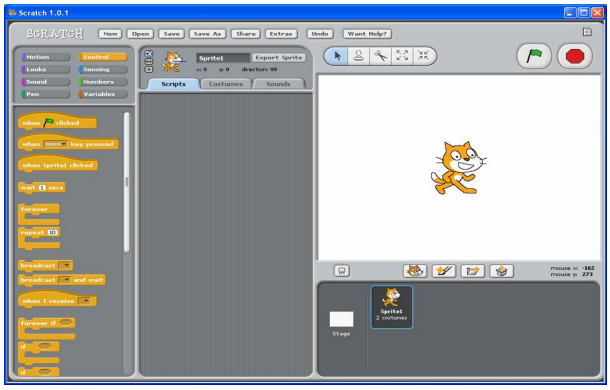


Fig. 3. Mit Scratch development environment & example code[3)]

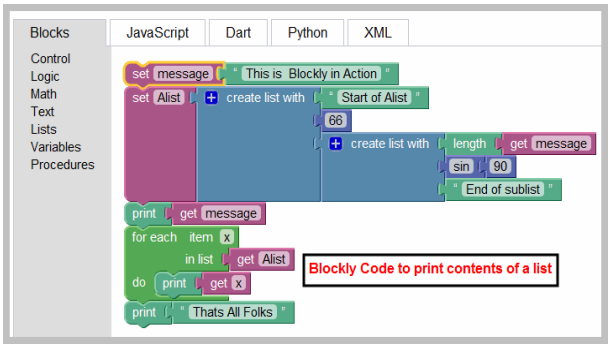


Fig. 4. Google Blockly development environment & example code[4)]

본 논문에서 제시하는 VP 환경에서는 구글의 Blockly를 활용하여 안드로이드 기반 태블릿 PC에서 산업용 로봇을 간단히 제어할 수 있는 환경을 제안하고 평가한다. 특히 본 프로그래밍 환경은 복잡한 소스코드를 작성하는 것보다 명령 블록(Block)을 이용하여 비숙련 사용자도 손쉽게 산업용 로봇제어를 할 수 있도록 제작되었다.

3. 산업용 로봇 제어를 위한 블록 기반의 비주얼 프로그래밍 환경1

본 논문에서는 Fig. 5와 같은 로봇 제어기용 기존 TP를 대체하기 위하여 태블릿 PC를 로봇 제어를 위한 TP로 활용하여, 제조 현장의 비전공 사용자도 쉽게 활용할 수 있는 VP 환경을 제공하고자 한다. 이를 위해 본 연구에서는 블록 기반의 VP 환경을 제작하였다. 제작한 환경은 구글의 오픈소스인 Blockly를 기반으로 작성되었는데, Blockly는 사용자가 명령 블록을 조립하여 프로그래밍 할 수 있는 비주얼 에디터로서 개발자는 자신의 웹 응용프로그램에 이를 적용시켜 초보자에게 보다 쉬운 UI를 제공할 수 있는 것으로 알려져 있다.

본 논문에서는 총 세 가지 프로그래밍 환경을 디자인하였으며 각 환경은 서로 다른 유연성을 가지는 명령 블록을 프로그래밍 작업에 사용할 수 있도록 했다. 3가지 프로그래밍 환경은 공통적으로 명령 블록을 왼쪽 명령어 풀에서 선택하여 드래그 앤 드롭 방법으로 작업공간에 블록을 추가한다.

Table 1은 세 가지 프로그래밍 환경에서 서로 다른 형태로 제공되는 Move(Motion) 명령어의 명칭과 각 문법, 그리고 기능을 표시하고 있다. Move(Motion) 명령어는 로봇 프로그래밍에 사용되는 다른 명령어에 비해 많은 속성들을 선택해야하는 명령어로서, 제안 환경에서 명령 블록을 디자인하는 기준 명령어로 사용되었다. Axis(축)와 Pxx(위치)는 로봇 제어 시 사용되는 매개변수이다.

본 논문에서는 명령 블록 설계가 프로그램 완성시간, 사용자 유용성 및 편리성에 끼치는 영향을 분석하기 위해 세 가지 프로그램 환경을 제안하였다. 각각의 명령어마다 해당되는 명령 블록을 제공하는 환경은 하나의 명령 블록을 선택하는 것만으로 명령어 완성이 가능하다. 그러나 많은 명령 블록 중 입력하고자 하는 명령 블록을 찾는데 시간이 걸리는 단점이 있다. 반면 되도록 적은 명령 블록을 제공하는 환경은 명령 블록을 쉽게 찾을 수 있는 반면, 최종 명령어 입력을 위해 추가로 속성 값을 선택해야 한다는 단점이 있다. 예를 들어 MOVS 명령 블록을 입력할 때 첫 번째 환경은 MOVS 블록을 프로그래밍 작업 창에 추가하는 것으로 명령이 완성되지만, 두 번째 환경에서는 명령 블록을 추가한 후 명령어 옆에 있는 명령 속성 창을 눌러 Straight 속성을 선택해야 MOVS 명령어를 입력할 수 있다.

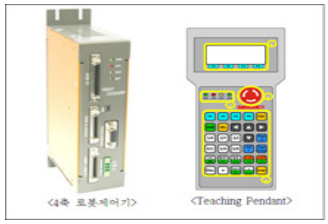


Fig. 5. Robot Controller & TP

2) 출처: January 27, 2010. LabVIEW Robotics Connects to Microsoft Robotics Studio Simulator [On-line], Available: https://labviewrobotics.wordpress.com/tag/arm/

3) 출처: September 25, 2013. How to create game typing on scratch [On-line], Available: http://wep2014b.studentscenter.org/2013/09/25/how-to-create-game-typing-on-scratch/

4) 출처: June 12, 2012. Google Blockly - A Visual Programming Language [On-line], Available: http://www.theopensourcery.com/keepopen/2012/google-blockly/

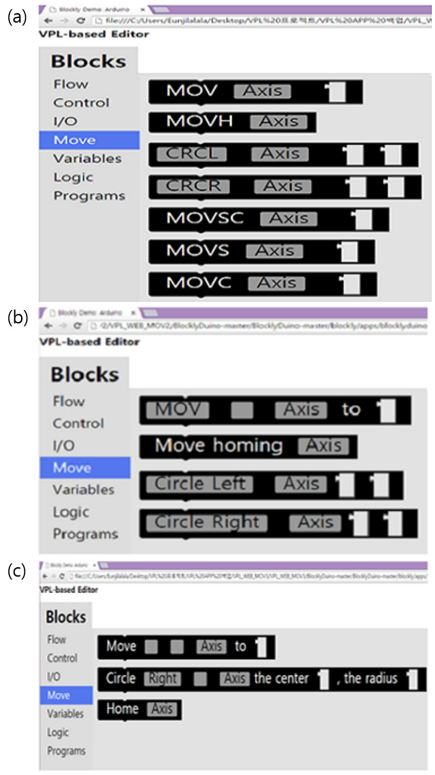
Table 1. MOVE(Motion) Command

	명령어	문법	기능
1	MOV	MOV Axis, Pxx	현 위치에서 Position 변수 값 또는 위치로 이동
2	MOVS	MOVS Axis, Pxx	직선보간 명령으로 현 위치에서 Position변수 값 또는 위치로 이동
3	CRCR	CRCR Axis, Pxx, Pxx	Circle Right 명령어로 [축 변수, 종점 P변수, 반지름 P변수]로 구성되어 있으며, 현 위치에서 시계방향으로 원호보간 이동
4	CRCRC	CRCRC Axis, Pxx, Pxx	CRCRC 명령어는 Continue 속성으로 원호보간 동작 시작 후, 바로 아래 명령어 실행
5	CRCL	CRCL Axis, Pxx, Pxx	Circle Left 명령어로 [축 변수, 종점 P변수, 반지름 P변수]로 구성되어 있으며, 현 위치에서 반 시계방향으로 원호보간 이동
6	CRCLC	CRCLC Axis, Pxx, Pxx	CRCLC 명령어는 Continue 속성으로 원호보간 동작 시작 후, 바로 아래 명령어 실행
7	MOVH	MOVH Axis	현 위치에서 해당 축을 Homing(고유 원점) 지점으로 이동
8	MOVC	MOVC Axis, Pxx	MOV명령어의 Continue 명령어
9	MOVSC	MOVSC Axis, Pxx	MOVS 직선 보간 Continue 명령어

본 논문에서는 Move명령어에 대해 각 7, 4, 3개의 명령 블록을 가진 세 가지 프로그램 환경을 고려한다. 프로그램 환경1에서 Move명령어는 총 7개이며 Fig. 6(a)와 같이 제공된다. 프로그램 환경1의 대부분의 명령어는 명령 블록에 일대일로 대응된다. 다만 예외적으로 CRCR, CRCRC, CRCL, CRCLC 명령어는 4개의 명령 블록을 제공하는 대신, 2개의 명령 블록, CRCR, CRCL을 제공하고 추가적인 변경을 통해 CRCRC 또는 CRCLC 명령 블록을 입력하도록 제공했다. 예를 들어 사용자가 CRCRC 명령 블록을 입력하려면 CRCR 명령 블록을 추가한 후 CRCR 명령어 이름을 눌러 CRCRC로 변경하여 명력 블록을 입력한다. 이는 제한된 태블릿PC 화면에 모든 명령 블록이 스크롤 없이 한눈에 보이도록 하기 위함이다.

프로그램 환경2는 Fig. 6(b)와 같으며 총 4개의 명령 블록을 제공한다. 또한 기존 TP에서 제공했던 명령어 대신 직관적인 명령 블록 이름, Move, Circle, Home 등을 제공하고, 추가적으로 속성 값을 선택하도록 디자인하였다. 따라서 사용자가 이동 명령어(MOV, MOVS, MOVC, MOVSC)를 입력할 경우 Move 명령 블록을 추가하고 옆에 위치한 속성 창에서 세부적인 속성(Straight, Continue, Straight&Continue)을 선택한다. MOVH은 MOV 명령어와 달리 Axis 매개변수만을 입력하는 다른 형태이므로 다른 명령어로 분류하고 별도의 명령 블록을 제공하였다. 좌우 원호보간 이동 명령어인 CRCR, CRCL 명령어는 각각 Circle Right, Circle Left 명령 블록을 추가한다. CRCRC와 CRCLC 명령어는 두 번째 속성 창에서 추가적으로 Right, Left 속성을 선택하여 입력한다.

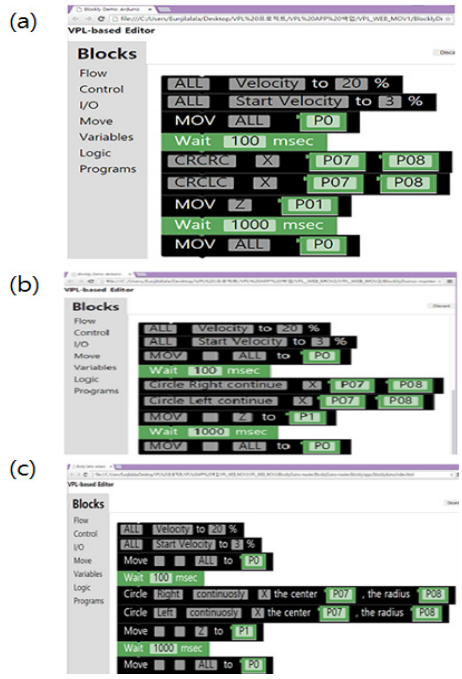
마지막으로 프로그램 환경3은 그림 Fig. 6(c)와 같이 최소한의 명령 블록을 제공한다. 디자인 방법은 유사기능을 수행하는 명령어와 같은 매개변수의 개수(Axis, Pxx)를 가지는 명령어들을 하나의 명령 블록으로 축약하여 여러 속성 값을 선택하도록 디자인하였다. 특히 원호보간 이동 명령어인 CRCR, CRCRC, CRCL, CRCLC명령어를 하나로 축약하여 환경2와 차이점을 보였다. 따라서 총 3개의 블록을 제공을 제공한다. 또한 프로그램 환경2와 같이 명령 블록의 이름은 직관적인 성격을 띠도록 Move, Circle, Home로 명명했다.



(a) Environment1, (b) Environment2, (c) Environment3

Fig. 6. A web-based environment

세 가지 다른 형태 Fig. 7(a), 7(b) 그리고 7(c)는 같은 로봇 동작을 생성하는 프로그램을 작성한 것이다. Fig. 7에서 볼 수 있듯이 프로그램 환경1의 명령 블록 디자인은 축약된 명령어 없이 모든 명령어가 블록으로 제공되어 블록선택까지의 단계가 간단하다. 프로그램 환경3의 명령 블록 디자인은 가장 적은 개수의 블록이 제공됨에 따라 속성 값을 매번 선택해야 하는 단계가 추가된다. 매번 선택해야 하는 속성 값은 환경2는 Continue와 Straight가 있으며, 환경3은 Continue와 Straight 외에도 원호보간이동 명령어의 Right, Left가 있다. 또한 추가된 속성 값에 의해 블록의 길이도 길어진다.



(a) Environment1, (b) Environment2, (c) Environment3

Fig. 7. Programming examples

```
LINE 1: VEL Y, 30                LINE 15: CRCL X, P00, P01
LINE 2: SVEL Y, 5                LINE 16: WAIT 500
LINE 3: ACC Y, 100               LINE 17: CRCLC X, P02, P01
LINE 4: ABS                      LINE 18: MOV Z, P03
LINE 5: MOVSC Y, P01             LINE 19: VEL Y, 30
LINE 6: VEL Z, 30                LINE 20: SVEL Y, 5
LINE 7: SVEL Z, 5                LINE 21: ACC Y, 100
LINE 8: ACC Z, 100               LINE 22: MOVH Y
LINE 9: ABS                      LINE 23: VEL Z 30
LINE 10: MOVS Z, P01             LINE 25: SVEL Z, 5
LINE 11: CRCR U, P00, P01        LINE 26: ACC Z, 100
LINE 12: WAIT 500                LINE 27: ABS
LINE 13: CRCRC U, P02, P01       LINE 28: MOVC Z, P01
LINE 14: MOV Z, P03              LINE 29: MOV X, P02
```

Fig. 8. A program for the experiment

3.1 실험

제안된 세 가지 웹 기반의 프로그래밍 환경을 평가하기 위하여 사용자에게 프로그램을 제시하고 제시된 블록 기반의 비주얼 프로그래밍을 수행하는 시간을 측정했다. 본 실험에서는 총 6명의 24~28 사이의 연령대를 가진 실험자를 모집하였으며 실험에 사용한 기기는 삼성 갤럭시 노트 10.1이다.

1) 실험 구성

실험에 사용된 프로그램은 Fig. 8과 같으며 실제 로봇 프로그래밍 시 사용되는 코드이다. 프로그래밍은 코드 라인 순서대로 진행된다. 실험자는 각 라인에 맞는 명령 블록을 프로그래밍 작업을 할 수 있는 공간에 드래그 앤 드롭 시킨 후 해당 축, 변수 값을 입력한다. 하나의 라인에 대한 프로그래밍 수행시간은 블록을 선택한 순간부터 블록의 모든 변수를 입력하여 블록이 완성될 때까지의 시간이다.

본 블록 기반의 비주얼 프로그래밍 환경은 실험자가 한 번도 접해보지 못한 환경이기 때문에 실험을 시작하기 전에 실험 시 사용자가 프로그래밍을 원활하게 할 수 있도록 사전 인지과정을 수행했다. 충분히 학습이 되었을 경우 본 프로그램에 숙련되어 환경 비교에 어려움이 생길 수 있으므로 각 환경에서 한 번씩만 실험용 프로그램을 작성하도록 했다. 서로 다른 3가지 웹 기반의 프로그래밍 환경은 학습효과를 줄이기 위해 랜덤으로 수행하였다. 또한 실험이 끝난 후에는 LabView를 사용한 텍스트/시각 프로그래밍 교육의 유용성 평가 항목 중 목표 프로그램 완성을 위한 이해도 및 성취도, 유용성 테스트, 프로그램 작성 과정에서의 난이도 부분을 평가항목으로 활용하였다[14]. 설문조사는 1부터 5까지 5단계로 나누어 응답하도록 했다.

Fig. 9. The average execution time to complete the given programming task

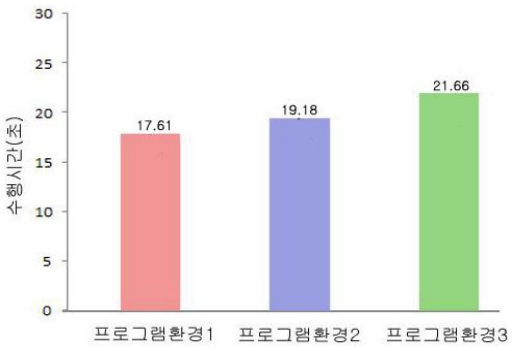


Fig. 10. The average execution time to select properties of a single block

2) 실험 결과

실험 결과는 Fig. 9, 10과 같다. Fig. 9는 각 프로그램 환경에 대하여 프로그램을 모두 작성할 때의 전체 시간을 측정하여 평균을 낸 그래프이다. 가장 빠른 것은 프로그래밍 환경2로 6분 14초로 가장 빨랐으며 가장 느린 것은 프로그래밍 환경3으로 6분 72초이다.

Fig. 10은 각 라인의 수행시간 중 여러 가지 속성 값을 가진 변수를 완성하기 위한 블록 완성 수행시간의 평균을 나타낸 그래프이다. 속성 값을 선택하는 라인은 5, 10, 11, 13, 14, 15, 17, 18, 22, 28, 29이다. 각 프로그램 환경간 차이

를 분석하기 위해 속성 입력의 실험 결과를 분석하였다. 실험 결과 분석방법은 One-Way ANOVA이고 유의수준은 0.1이다. 프로그램 환경 간의 주 효과는 유의하다($F(2,10)=4.747$, $p<0.1$). 사후 검증 결과 웹기반의 서로 다른 세 가지 환경의 수행시간의 차이는 유의적이라고 할 수 있다($p=.010<0.1$). 사후 집단간 비교분석한 결과, 환경1과 환경2는 차이를 보이지 않았고 환경1과 환경3 간의 차이가 유의적인 것으로 나타났다($p=.007<0.1$). 즉, 환경1은 환경3 보다 수행시간에 있어서 평균 4.05만큼 높은 결과를 가져왔으며, 그 차이는 유의하다($p=.007<0.1$).

Fig. 10에서 볼 수 있듯이 평균 블록 완성 수행시간은 프로그래밍 환경1이 가장 수행시간이 적고 프로그래밍 환경3이 가장 수행시간이 길다. 각 환경은 서로 약 3초 정도의 수행시간에 차이를 보이며, 프로그램 완성 시 수행시간이 2초에서 4초 정도 차이가 난다는 것을 감안하였을 때 큰 차이를 보인다고 할 수 있다. 또한, 앞서 보았던 결과와 달리 프로그래밍 환경1이 수행시간이 더 적게 걸리는 것을 볼 수 있다.

이는 한정된 화면 안에서 블록 기반의 비주얼 프로그램을 작성하기 때문이다. 명령 블록을 추가할 때 사용자는 MOVE 메뉴를 누른 후 명령 블록을 화면에 추가한다. 명령 블록의 메뉴가 길면 사용자는 명령 블록을 입력하기 위해 화면의 위치를 상대적으로 많이 조정해야 한다. 이로 인해 프로그램 환경1에서의 속도가 빠름에도 불구하고 전체적인 프로그램의 작성 속도는 프로그램 환경2가 빠르게 나타나는 것이다.

Table 2. The questionnaire

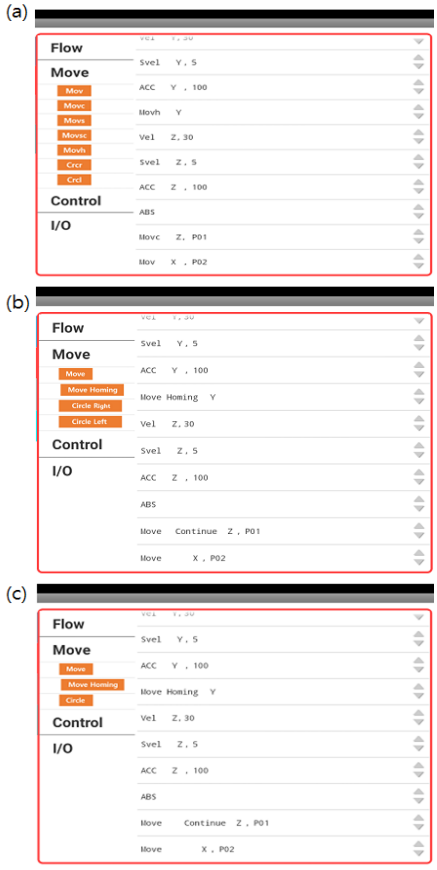
유용성테스트	환경 1	환경 2	환경 3
프로그램 작성이 용이한가?	4.33	4.50	4.00
프로그램 이해가 용이한가?	4.50	4.50	3.33
문제 분석이 용이한가?	4.67	4.33	3.83
프로그램 설계가 용이한가?	4.00	4.67	3.83
프로그램 변경이 용이한가?	3.33	4.17	3.83
프로그램 작성과정에서의 난이도			
문법 배우기의 난이도	2.83	2.83	3.50
구문 오류 정도	2.83	3.00	3.33
프로그램 수정 관점에서의 난이도	3.17	2.50	3.67
목표 프로그램 완성을 위한 이해도 및 성취도			
프로그램 표기법에 대한 이해 정도	3.17	3.17	4.17
프로그램 표기법의 이해에 따른 구문 선택 능력	2.83	3.00	3.33
프로그램 조합에 의한 프로그램 작성 능력	2.83	2.67	4.17
최종 프로그래밍 학습 성취도 결과	4.00	4.50	3.17

Table 2는 6명의 사용자 설문 결과이다. 결과 중에서도 프로그램 수정관점에서의 난이도와 최종 프로그래밍 학습 성취도 결과에서 환경2가 가장 효율적이라고 나타났다. 비숙련자가 사용한다는 측면에서 주목할 만한 결과라고 할 수 있다. 이외에도 환경1과 환경2가 전체적으로 비슷한 점수를 받았다. 환경3은 실험 결과와 동일하게 설문조사에서도 사용자가 선호하지 않는 것으로 나타났다.

4. 산업용 로봇 제어를 위한 블록 기반의 비주얼 프로그래밍 환경2

기존의 프로그래밍 환경은 한정된 화면 안에서 프로그램을 작업해야 함으로 블록의 변수가 많아지거나 블록이 길어지게 되면 화면의 위치를 조정하는데 어려움이 있었다. 이를 보완하기 위해 명령어 블록을 디자인하여 어느 정도 수행시간을 단축할 수 있었으나 여전히 한정된 화면 문제를 가지고 있었다. 따라서 실험 1의 결과를 바탕으로 태블릿 PC 환경에서 사용자의 프로그래밍 수행시간 및 사용자 경험을 향상시킬 수 있도록 최적화된 VP 환경을 설계하였다.

본 환경은 웹기반에서 안드로이드 기반으로 변경하여 제작되었다. 이는 태블릿 PC환경에 최적화된 환경을 사용하기 위함이다. 디자인 된 프로그램은 Fig. 11(a)-(c)와 같으며 각각 서로 다른 명령 블록을 프로그램 작성 시 제공한다. 명령 블록은 이전 실험에서 디자인하였던 것을 그대로 각 환경에 적용하였으며 이전의 웹 기반 환경과 다른 점은 블록 생성 방법 및 프로그램 표현 방법이다.



(a) Environment1, (b) Environment2, (c) Environment3

Fig. 11. A android-based environment

본 환경에서는 이전의 환경에서 한정된 화면으로 인해 작업 공간을 재조정해야하는 불편함을 최소화하고자 하였다. 이를 위해 화면에 명령어 블록을 드래그 앤 드롭 하여 추가하는 방법이 아닌 해당 명령어를 터치하여 자동으로 명령 블록을 추가하는 방법을 사용하였으며 스크롤바를 이용하여 화면을 조작하도록 하였다. 또한 명령 블록의 오른쪽에 이벤트 핸들러를 추가하여 명령 블록의 순서를 자유롭게 변경하도록 디자인하였다. Fig. 12는 명령어의 변수 입력을 위한 창으로 사용자가 터치를 이용하여 쉽게 변수를 입력할 수 있도록 제작하였다.

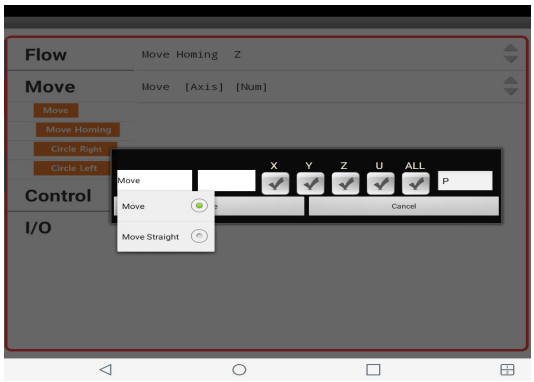


Fig. 12. A example to select and input variable values

4.1 실험

1) 실험 구성

안드로이드 기반의 블록 기반 프로그래밍 환경의 성능을 평가하기 위해 실험을 진행하였다. 본 실험에서는 실험1과 동일한 방법을 이용하여 사용자 실험을 진행하였다.

실험이 끝난 후에는 총 2가지 설문조사를 진행하였으며 설문조사는 각각 교육적 측면과 디자인 측면에서 유용성을 평가한다. 교육적 측면의 유용성 평가는 실험 1에서 사용된 설문조사를 이용하였으며, 디자인 측면의 유용성 평가는 사용자 인터페이스 디자인의 유용성에 관한 설문지로서 유용성, 사용자 편리성 부분을 활용하였다[15]. 설문 점수는 1단계부터 5단계까지로 나누어 응답하도록 하였으며 1에서 5로 숫자가 높아질수록 그 응답에 동의하는 것으로 하였다.

실험참가자는 총 7명으로 17~45 사이의 연령대로 구성되었으며, 실험에 사용한 기기는 엘지 G패드이다.

2) 실험 결과

실험 결과는 Fig. 13, 14와 같다. Fig. 13은 프로그램을 모두 작성할 때의 걸린 시간을 측정한 것으로서 웹 기반의 실험 결과와 비교하였을 때, 최대 2분 30초 가량 단축된 결과를 얻을 수 있다. 가장 빠른 프로그래밍 환경은 1로써, 환경2와 불과 0.05초의 차이를 나타낸 것을 확인할 수 있다.

Fig. 14는 속성 값을 선택하는 실험 결과로써 실험 결과 분석방법은 One-Way ANOVA이고 유의수준은 0.1이다. 프로그램 환경 간의 주 효과는 유의하다($F(2,12)=2.547$, $p<0.1$). 사후검증 결과 안드로이드 기반의 서로 다른 세 가지 환경의 수행시간의 차이는 유의적이라고 할 수 있다($p=.081<0.1$). 사후 집단간 비교분석한 결과, 환경1과 환경2는 차이를 보이지 않았고 환경1과 환경3간의 차이는 유의적인 것으로 나타났다($p=.067<0.1$). 즉, 환경1은 환경3 보다 수행시간에 있어서 평균 1.75만큼 높은 결과를 가져왔으며, 그 차이는 유의적이다($p=.067<0.1$). 웹 기반의 실험 결과와 비교하였을 때, 최대 11.49초가량 단축된 결과를 얻었고, 프로그래밍 환경1이 10.17초로 가장 적은 수행시간이 결렸다.

대체적으로 웹기반의 실험환경보다 훨씬 적은 시간의 수행시간이 걸렸고, 속성 값을 선택해야 하는 Task의 수행시간은 세 가지 프로그래밍 환경 모두 11초 안팎으로 많은 차이를 보이지 않았다.

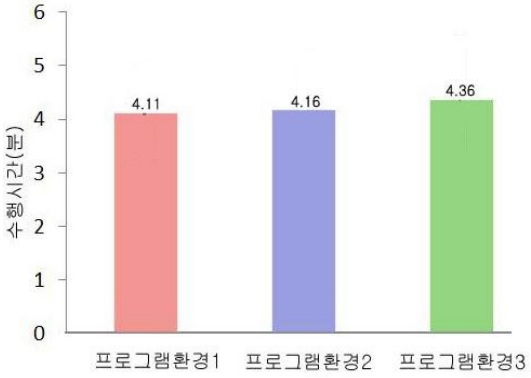


Fig. 13. The average execution time to complete the given programming task

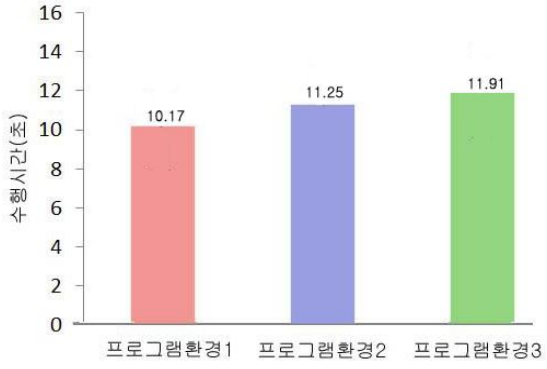


Fig. 14. The average execution time to select properties of the block

이와 같은 결과가 얻어진 이유는 웹 기반의 실험 환경은 태블릿의 제한된 공간 안에서 블록의 위치와 화면의 위치를 재조정해야 하는 필요가 있었던 반면, 안드로이드 기반의 실험환경은 터치기반의 자동 블록 추가 기능과, 자동으로 마지막 블록으로 포커스가 이동하는 장점이 수행시간의 단축을 유도했다.

Table 3은 7명의 사용자 설문 결과이다. 기존 실험과 동일하게 프로그램 수정 관점에서의 난이도와 최종 프로그래

Table 3. The questionnaire of educational aspects

유용성테스트	환경1	환경2	환경3
프로그램 작성이 용이한가?	4.29	4.43	3.57
프로그램 이해가 용이한가?	4.29	4.86	3.86
문제 분석이 용이한가?	4.00	4.29	3.57
프로그램 설계가 용이한가?	4.00	4.43	3.57
프로그램 변경이 용이한가?	3.57	4.00	3.86
프로그램 작성과정에서의 난이도			
문법 배우기의 난이도	2.43	2.00	2.57
구문 오류 정도	1.71	1.29	2.43
프로그램 수정 관점에서의 난이도	2.57	1.86	2.57
목표 프로그램 완성을 위한 이해도 및 성취도			
프로그램 표기법에 대한 이해 정도	3.14	2.43	3.29
프로그램 표기법의 이해에 따른 구문 선택 능력	2.57	2.57	3.43
프로그램 조합에 의한 프로그램 작성 능력	2.86	2.86	3.29
최종 프로그래밍 학습 성취도 결과	3.29	3.57	3.57

Table 4. The questionnaire of design aspects

질문	환경1	환경2	환경3
1. 사용하기 쉬웠는가?	4.57	4.29	4.00
2. 사용할 때 즐거웠는가?	3.86	3.86	3.86
3. 로봇 프로그래밍 시 도움이 되겠는가?	3.14	3.57	4.00
4. 프로그래밍을 배울 시 도움이 되겠는가?	3.57	3.86	3.71
5. 프로그래밍을 배우는 활동이 흥미로웠는가?	3.71	4.14	3.86
6. 응용프로그램의 색조화가 적절했는가?	3.86	3.86	3.57
7. 배경색이 좋았는가?	3.86	3.86	3.57
8. 제스처가 명확하고 효과적이었는가?	4.57	4.57	4.57
9. 언제든지 프로그램을 종료할 수 있었는가?	4.71	4.71	4.71
10. 모든 지시사항이 이해하기 쉬웠는가?	3.00	4.14	3.71
11. 프로그램을 사용할 기회가 다시 주어진다면 사용하겠는가?	3.86	4.14	4.00
12. 이 프로그램을 사용하여 로봇 프로그래밍을 배우는 것을 추천하겠는가?	4.14	4.29	4.43

밍 학습 성취도 결과에서 환경2가 가장 효율적이라고 나타났다. 웹 기반의 실험환경과 달리 앱 기반의 실험 환경에서는 환경3은 사용자가 환경1, 2와 전체적으로 비슷한 점수를 받았다.

Table 4는 서로 다른 프로그래밍 환경에 대한 사용자 인터페이스 디자인의 유용성에 관한 설문지로서 유용성, 사용난이도 항목의 결과에서 대체로 사용자는 환경2를 더 선호하는 것으로 나타났다.

Fig. 13과 Fig. 14는 명령 블록 개수와 추가적인 속성 입력이 프로그램 작성 시 성능, 사용자 유용성 및 편리성 측면에 영향을 미치는 요소임을 나타낸다.

가장 많은 명령 블록과 가장 적은 속성 입력을 하도록 구성되어 있는 프로그램 환경1은 성능과 사용자 유용성 및 편리성 측면에서 전체적으로 높은 점수를 받았다.

프로그램 환경2의 명령 블록은 프로그램 환경1의 명령 블록의 절반정도 이나 명령어마다 속성은 프로그램 환경1 보다 하나를 더 입력해야 한다. 이러한 추가적인 속성 입력으로 인하여 Fig. 14와 같이 많은 속성 입력 시간이 소요되지만, 전체적인 수행시간은 프로그램 환경1과 0.05초만의 차이를 보였다. 이는 프로그램 작성 시 여러 명령 블록 사이에서 원하는 명령 블록을 찾는 시간이 명령어와 관련된 간단한 속성을 찾아 입력하는 시간에 상당하는 시간이 걸린다는 것을 의미한다. 또한 사용자 설문 결과 중 프로그램 작성과정에서의 난이도 측면에서 프로그램 환경2가 전체적으로 낮은 난이도를 갖고 있음을 확인할 수 있다.

프로그래밍 환경3은 다른 프로그래밍 환경과 비교하였을 때 오랜 프로그램 완성시간과 낮은 사용자 선호를 보이므로 가장 낮은 성능을 보이는 환경이다.

이를 바탕으로 본 논문에서는 프로그램 환경2와 같이 유사한 기능을 하는 명령어를 묶어 프로그램 환경을 디자인

하는 것이 프로그램 완성시간, 사용자 유용성 및 편리성 측면에서 가장 좋은 디자인 방법이라는 것을 발견했다.

5. 결 론

본 논문에서는 태블릿 PC환경에서 비주얼 프로그래밍 언어를 기반으로 드래그 앤 드롭 기법을 사용하여 프로그래밍을 위한 서로 다른 3가지의 명령어 블록 디자인을 두 가지 플랫폼을 거쳐 비교실험 하였다. 실험의 목적은 적절한 명령어의 개수와 효율적인 분류 기준을 정하고 사용자 친화적인 프로그래밍 환경을 제공하는데 있다.

같은 프로그램을 프로그래밍을 수행하는데 걸린 시간을 비교한 결과, 한정된 디스플레이 창과 실제 산업용 로봇 제어를 위한 프로그램 길이를 고려하면 속성 값을 입력했을 때의 수행 시간이 가장 느린 프로그래밍 환경3이 가장 부적절한 환경이라고 볼 수 있다. 또한 프로그램을 완성하는데 걸린 시간과 사용자 설문 결과를 보아 프로그래밍 환경2가 프로그래밍 하는데 적절한 환경을 제공해 준다고 판단할 수 있다. 또한 명령 블록 제공 방법 외에도 블록 추가하는 방법과 작업 화면을 제공하는 방법이 프로그래밍 작업에 영향을 끼치는 것을 확인하였다.

References

[1] Carlos Mateo, Alberto Brunete, Ernesto Gambao, and Miguel Hernando, “Hammer: An Android Based Application for End-User Industrial Robot Programming,” in *Mechatronic and Embedded Systems and Applications(MESA)*, 2014.

[2] Yasir Jan, Syed Hassan, Sanghun Pyo, and Jungwon Yoon, “Smartphone Based Control Architecture of Teaching Pendant for Inustrial Manipulators,” in *International Conference on Intelligent Systems, Modelling and Simulation*, 2013.

[3] W. G. Ji and B. J. Park, “Visual Programming Tool Based on Blocks,” *The Korean Institute of Communications and Information Sciences(KICS)*, 2013(in Korean).

[4] H. L. Kim, E. K. Park, H. J. Kim, and J. M. Bae, “An Integrated C Programming Environment for Novices Based on Visuals,” *The Journal of Korean Association of Computer Education*, Vol.16, No.6, pp.111-120, 2013(in Korean).

[5] David J. Malan, and Henry H. Leitner, “Scratch for Budding Computer Scientists,” 2007.

[6] Adiel Ashrov, Assaf Marron, Gera Weiss, Guy Wiener, “A use-case for behavioral programming: An architecture in JavaScript and Blockly for interactive applications with cross-cutting scenarios,” *Science of Computer Programming*, Vol.98, Pt.2, pp.268-292, 2015.

[7] Jesus S. Cepeda, Luiz Chaimowicz, and Rogelio Soto, “Exploring Microsoft Robotics Studio as a Mechanism for Service-Oriented Robotics,” *IEEE*, pp.7-12, October, 2010.

[8] Nesimi Ertugrul, “Towards Virtual Laboratories: a Survey of LabView based Teaching/Learning Tools and Future Trends,” *The International Journal of Engineering Education*, Vol.16, No.3, pp.171-180, 2000.

[9] B. W. Choi, “A Review and Outlook of Robotic Software Frameworks,” *Korea Robotics Society*, Vol.5, No.2, pp.169-176, 2010(in Korean).

[10] B. Y. Lee, Y. J. Lee, and S. J. Choi, “Basic Engineering Experiment using the LabView,” Ohm, 2006(in Korean).

[11] D. Y. Gwak “A computer-based instrumentation and control,” Ohm, 2006(in Korean).

[12] J. Y. Park, “An Investigation of the Structural Relationships among Students’ Characteristics, Flow, and Learning Effects in a SCRATCH Programming Course for Elementary School Students,” The graduate school of Ewha womans university, 2015(in Korean).

[13] I. Iturrate, G. Martin, J. Garcia-Zubia, I. Angulo, O. Dziabenko, P. Orduna, G. Alves, and A. Fidalgo, “A mobile robot platform for open learning based on serious games and remote laboratories,” *2013 1st International Conference of the Portugese Society for Engineering Education(CISPEE)*, pp.1-7, October, 2013.

[14] H. J. Lee, “A Study on the Improvement of Usability in PDA Website Based on User Activity Analysis,” *HCI KOREA 2013*, pp.599-604, 2003.(in Korean).

[15] Mokhtar, Shamsul Anuar, and Siti Mashitah Shamsul Anuar. “Learning application for Malaysian sign language: content design, user interface and usability,” *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication,* ACM, 2015.

박 은 지

e-mail : eunji1454@korea.ac.kr

2014년 고려대학교 컴퓨터정보학과(학사)

2014년～현 재 고려대학교 컴퓨터정보학과 석사과정

관심분야 : Human Computer Interaction

서 경 은

e-mail : chemd111@korea.ac.kr

2012년 고려대학교 컴퓨터정보학과(학사)

2015년 고려대학교 컴퓨터정보학과(석사)

2015년～현 재 고려대학교 컴퓨터정보학과 박사과정

관심분야 : Human Computer Interaction

박 태 곤

e-mail : tgpark93@gmail.com
1993년 경북대학교 전자공학(학사)
2008년~현 재 ㈜프레스토솔루션 부사장
관심분야 : 로봇 및 정밀 모션제어

선 덕 한

e-mail : sundukhan@prestosolution.co.kr
2001년 경일대학교 제어계측공학과(학사)
2009년 경북대학교 전자공학과(석사)
2008년~현 재 ㈜프레스토솔루션 수석
관심분야 : 로봇, 모션제어

조 현 중

e-mail : raycho@korea.ac.kr
1996년 경북대학교 전자공학부(학사)
1998년 포항공과대학교 전자전기공학과 (석사)
2006년 Computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg VA(Ph.D)
2009년~현 재 고려대학교 컴퓨터정보학과 부교수
관심분야 : Real-time computing for embedded systems, Human-computer interactions for smart devices