https://doi.org/10.7735/ksmte.2017.26.1.20

J. Korean Soc. Manuf. Technol. Eng.
ISSN 2508-5107(Online) / ISSN 2508-5093(Print)

가상로봇과 실제로봇 사이의 운동 동기화를 통한 물체 인식 및 목표물 추적방안

안혜경[a], 강현준[a], 김진범[a], 정지원[a], 옥서원[b], 김동환[c*]

Object Recognition and Target Tracking Using Motion Synchronization between Virtual and Real Robots

Hyeo Gyeong Ahn[a], Hyeon Jun Kang[a], Jin Beom Kim[a], Ji Won Jung[a],
Seo Won Ok[b], Dong Hwan Kim[c*]

[a] *Department of Mechanical System Design Engineering, Seoul National University of Science and Technology, 232, Gongneung-ro, Nowon-gu, Seoul 01811, Korea*
[b] *Department of Mechanical Design and Robot Engineering, Graduate School, Seoul National University of Science and Technology, 232, Gongneung-ro, Nowon-gu, Seoul 01811, Korea*
[c] *Department of Mechanical System Design Engineering, Seoul National University of Science and Technology, 232, Gongneung-ro, Nowon-gu, Seoul 01811, Korea*

ARTICLE INFO

Article history:
Received 23 September 2016
Revised 26 December 2016
Accepted 5 January 2017

Keywords:
Virtual reality
Virtual object
Synchronization
Surface's noise removal
Object recognition
Target tracking

ABSTRACT

Motion synchronization between developed real and virtual robots for object recognition and target tracking is introduced. ASUS's XTION PRO Live is implemented as a sensor and configured to recognize walls and obstacles, and perceive objects. In order to create virtual reality, Unity 3D is adopted to be associated with the real robot, and the virtual object is controlled by using an input device. A Bluetooth serial communication module is used for wireless communication between the PC and the real robot. The motion information of a virtual object controlled by the user is sent to the robot. Then, the robot moves in the same way as the virtual object according to the motion information. Through motion synchronization, two scenarios, which map the real space and current object information with virtual objects and space, were demonstrated, yielding good agreement between the two spaces.

1. 서 론

현대 기술사회에서 가상현실 분야는 다방면으로 적용되고 있다. 현실 세계에서 표현할 수 없는 요소들을 영상 안에서 구현하거나 가상현실에서 현실 세계와 비슷하게 재현하는 등 많은 응용기술에 적용되고 있다. 게임분야 뿐만 아니라 간접체험, 영상홍보 등 많은 분야에서 가상현실 기술이 쓰이고 있다. 가상현실(VR, virtual reality) 기술은 컴퓨터에 의해 시뮬레이션되는 3차원 가상환경을 통해 효율성과 유용성을 제공하므로 데이터의 시각화, 시뮬레이션 뿐 아니라 교육 및 의료 등 다양한 분야에 걸쳐 폭넓게 이용되고 있다[1-4].

가상현실이 발전함에 따라 이를 로봇에 적용하는 사례도 많아졌다. 의료용 로봇인 Da Vinci 로봇을 가상현실에서 훈련하는 프로그램도 만들어졌다[5]. 이를 통해 사용자는 현실의 고비용의 장비를

* Corresponding author. Tel.: +82-2-970-6362
Fax: +82-2-974-8270
E-mail address: dhkim@seoultech.ac.kr (Dong Hwan Kim).

사용하지 않고도 로봇 사용법을 훈련할 수 있는 이점을 가지게 된다. 이 밖에도 재활훈련, 시설 장비 사용 시뮬레이션, 가상현실 게임 등 가상현실을 적용한 로봇 시스템은 효율이나 비용 측면의 이점이 있어 여러 분야에서 그 사용성이 획기적으로 증대되고 있다.

본 논문에서는 가상 이미지의 정보를 이용하여 현실에서 영상처리를 보완하는데 목표를 두고 있다. 기존의 탐사 로봇[6,7] 같은 경우 영상정보 데이터의 손실이나 오류가 발생할 경우 경로 확보에 문제가 생긴다는 한계가 있다. 무인자동차의 SLAM (self localization and mapping) 시스템의 경우 영상 데이터가 기하급수적으로 누적되어 많은 데이터를 처리해야 하는 부담이 있다. 또한 기존의 가상현실이나 증강현실을 로봇에 접목시킨 사례들은 한정된 공간이나 장비가 설치된 공간에서만 사용이 가능하다. 이를 해결하기 위해, 본 논문에서는 실시간 영상데이터를 기반으로 가상공간을 구축하고 사용자에게 시각적인 정보를 효과적으로 실제 시스템에 전달하는 방법을 제시한다.

제안한 시스템은 두 개의 시나리오로 구성되었다. 시나리오 1에서는 영상데이터를 기반으로 실제 로봇이 벽과 장애물을 판단하여 가상세계로 정보를 알려주고 가상세계에서 장애물을 형성하여 회피한다. 또한 목표물의 3차원 좌표를 인식하여 동일한 위치에 가상의 목표물을 설정한다.

시나리오 2에서는 영상장치를 사용 할 수 없는 야간이나 시야 판독이 불가한 지역에서 로봇을 제어하는 시스템이다. 시나리오 1에서 구축된 가상공간을 바탕으로 사용자에게 판독 불가한 지역에서 가상공간을 통하여 시각적인 판단을 가능하게 하여 로봇을 제어한다. 사용자는 가상현실 속의 객체를 컨트롤하고 실제 로봇은 가상객체와 동일하게 장애물을 피하거나 목표물을 잡는 시스템이다.

본 연구에서 제안한 시스템은 가상현실 구현을 통해 지진이나 화재, 건물 붕괴와 같은 재난이 발생했을 때 투입되는 재난 구조로봇 또는 사람이 가기 힘든 곳을 탐사해야 하는 인간 대신 탐사하는 탐사로봇의 개발에 활용될 수 있다.

2. 로봇 구조 및 영상처리

2.1 시스템 구조

본 연구에서는 실제 로봇 시스템에서는 주변 정보를 얻기 위한 영상장치 및 영상처리 프로그램이 구축되어 있다. 영상데이터를 토대로 가상세계를 구축하기 위해 UNITY 3D 엔진을 사용하였다. 실제 제작한 로봇에는 AVR MCU를 이용하여 블루투스 모듈을 장착하여 가상공간과 통신을 수행한다. 그리고 통신으로 받아온 데이터를 이용해 LED Cube 디스플레이와 구동 모터를 제어하여 구동한다. Fig. 1과 같이 본 논문에서 개발한 로봇은 전/후진과 제자리회전의 구동을 하며 2자유도의 구동과 4자유도의 그리퍼를 설계

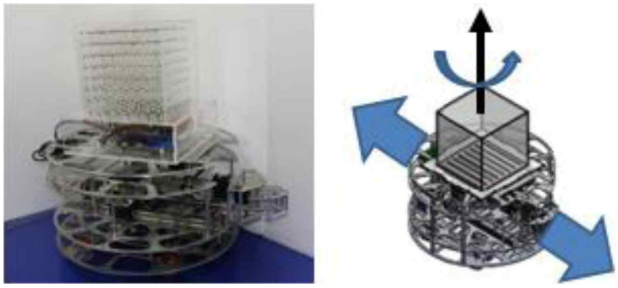


Fig. 1 Developed 6 DOF Robot and driving directions

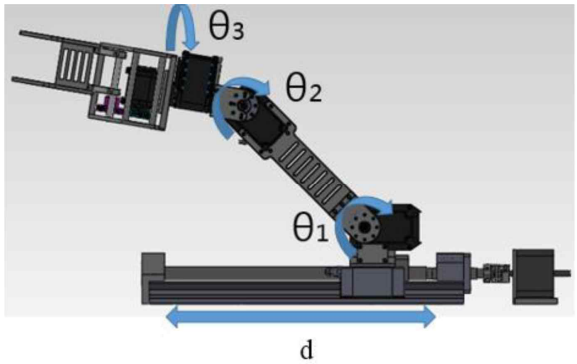


Fig. 2 4 DOF robot arm mechanism

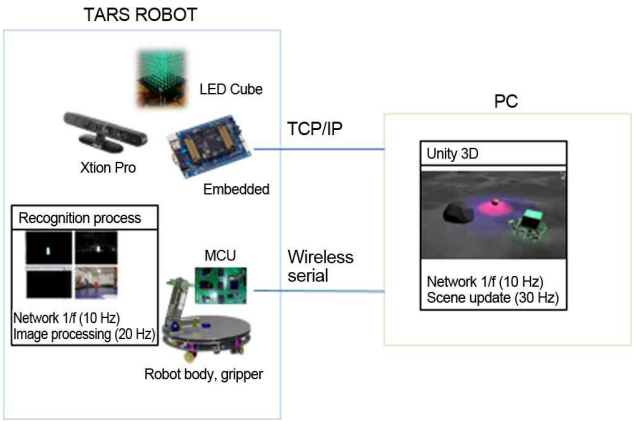


Fig. 3 System configuration between real robot and virtual robot

하여 두 세계에서의 동일 동작, 물체를 잡아오는 동작을 수행할 수 있도록 제작하였다. Fig. 2에서는 로봇 팔 관절에 표시된 각 조인트 각도를 표시하고 있다.

가상 공간의 객체는 사용자에 의해 제어되고 객체의 움직임 양은 실제 로봇에게 전달되어 실제로봇과 가상로봇간의 움직임 동기화를 한다.

Fig. 3은 실제로봇과 가상로봇간의 구성도를 보여주고 있다. 왼쪽의 그림에서는 실제로봇을 구성하는 제어기 및 영상처리를 위한 프로그램, 물체와의 거리를 측정하기 위한 액션프로(Xtion pro), 그리고 임베디드 컴퓨터를 로봇에 장착시켜 영상처리 및 로봇제어기인 마이크로프로세서를 통합적으로 제어하거나 영상데이터를 처

리하는 것을 보여준다. 오른쪽 그림에서는 가상로봇을 표현하며 여기서는 UNITY 3D 엔진을 사용하여 가상공간에서의 물체 및 가상로봇를 보여주고, 실제로봇과의 통신은 TCPIP를 통하여 수행하는 과정을 보여주고 있다. 또한 실제로봇으로 부터의 정보를 입력받고 가상로봇에서 실제로봇으로 위치정보를 전달하는 것을 보여준다.

2.2 영상처리

컴퓨터 그래픽으로 생성한 가상의 로봇이 주변의 물체를 피하거나 잡는 모션을 취할 때 실제로봇도 동일한 동작을 취하도록 하기 위해 현실세계의 환경과 가급적 동일한 가상세계를 구축할 필요가 있다. 따라서 실제의 로봇 주변에 놓인 물체를 인식하여 가상의 로봇 주변에 생성하되, 물체와 로봇의 상대적 위치관계를 설정하는 방법을 정확히 수행해야 한다. 실제로봇 주변 물체의 3차원 위치정보는 액션프로에 내장되어있는 IR센서로 얻은 심도 값(DEPTH VALUE)를 활용하여 심도 이미지(DEPTH IMAGE)를 구성해 추출하였다. Fig. 4에 나와있는 전체적인 영상처리 과정은 심도 이미지를 이용하여 주변물체를 인식하는 Mode 1과 인식한 물체 중 목표물을 선택하여 RGB화면에서 목표물을 추적하는 Mode 2로 구성하였다. 우선적으로 지면과 물체의 잡음을 제거하는 필터를 설계하여 카메라 위치로 인해 지면과 물체가 구분되지 않는 문제를 해결하였다.

그 다음으로 심도 이미지(depth image)에 블롭 레이블링(blob labeling)을 적용하여 물체를 인식하고 물체중심의 심도 값을 활용하여 3차원 위치정보를 얻었다. 이때 심도 값을 얻기 위해서는 액션프로(xtion pro)에 내장된 IR센서의 최소 가시거리 (0.4 m) 이상으로 거리가 확보되어야 한다. 따라서 최소 가시거리(0.4 m) 이내의 물체 위치를 확인하기 위해 인식한 물체의 경계와 RGB영상에 캠시프트(cam shift)[9]를 적용하여 선택한 물체의 위치를 확인한다.

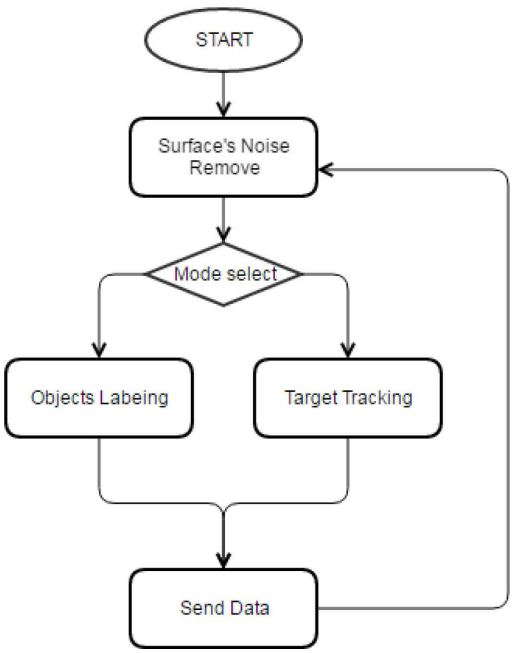


Fig. 4 Flowchart of image processing

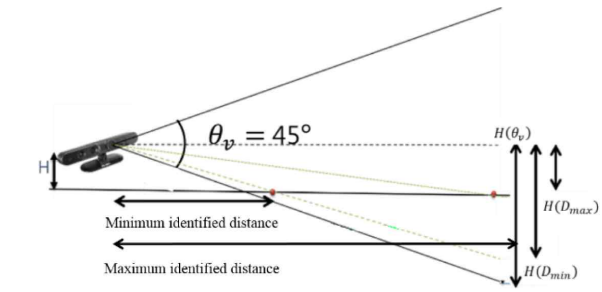
2.3 지면인식

Fig. 5(a)는 액션프로 장치의 심도 값(depth value) 정보를 기반으로 이미지를 구성 할 때 지면과 물체를 구분하기 위해 액션프로(xtion pro)의 인식범위를 표기하며, Fig. 5(b)는 윈도우 화면구성에서 표면 픽셀값의 범위를 표현한다. 여기서는 Fig. 5(a)와 (b)를 고려하여 지면의 잡음을 순차적으로 제거하는 과정을 기술하고자 한다. 식 (1), (3)과 (5)는 액션프로의 최대인식거리(D_{max})와 최소인식거리(D_{min}), 그리고 수직 시야 각 (θ_v)을 고려하여 실제 지면인식 깊이를(H_{θ_v}, $H_{D_{min}}$) 구하는 과정이다. 식 (2), (4), (6)은 비례식과 윈도우의 높이(W_H), 식 (1), (3), (5)를 활용하여 화면상의 지면 픽셀 위치($P_{y\theta_v}$, $P_{yD_{min}}$, $P_{yD_{max}}$)를 구하는 과정이다.

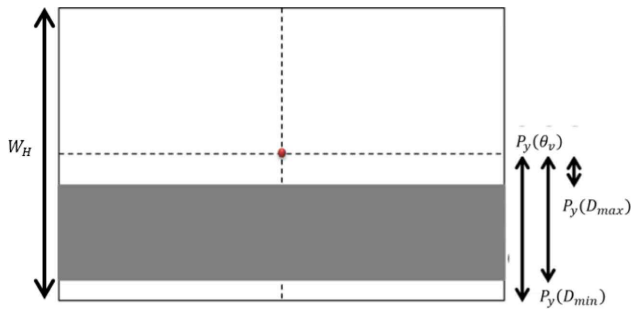
$$H_{\theta_v} = D_{max} \tan\left(\frac{\theta_v}{2}\right) \tag{1}$$

$$P_{y\theta_v} = \frac{W_H}{2} \tag{2}$$

$$H_{D_{min}} = D_{max} \frac{H}{D_{min}} \tag{3}$$



(a)



(b)

Fig. 5 (a) Surface identification depth calculation using maximum and minimum distance, and horizontal field of view of Xtion pro device, (b) Range of surface pixel values on the display

$$P_{yD_{min}} = \frac{W_H\ H}{2\ D_{min}\ \tan\left(\frac{\theta_v}{2}\right)} \quad (4)$$

$$H_{D_{max}} = \mathrm{H} \quad (5)$$

$$P_{yD_{max}} = \frac{W_H\ H}{2\ D_{max}\ \tan\left(\frac{\theta_v}{2}\right)} \quad (6)$$

지면이 화면에서 차지하는 부분을 알아내기 위해, 위 식에서 H를 카메라의 인식각도와 인식거리에 따라 화면의 높이를 픽셀단위로 나타냈다.

본 연구에서 화면의 크기는 320×280 픽셀로 구성하였다. 윈도우의 높이는 280픽셀이므로 W_H=280으로 고정되었다. H=30 cm, D_{max}=4 m, D_{min}=40 cm, W_H=280일 때, $H_{D_{max}}$=125, $H_{D_{min}}$=0이므로 지면의 범위는 0~125 픽셀 이라는 결과를 얻을 수 있었다.

화면상 지면의 범위를 픽셀단위로 구하여 해당 범위에서 픽셀값을 측정하여 수평축은 위에서 구한 화면의 범위 (0~125 pixel)를 그리고 지면의 intensity를 수직 축으로 한 그래프로 나타내었다 (Fig. 6). 지면에서 가장 가까운 거리인 0 픽셀에서는 255이며 멀수록 지면의 intensity가 작아지는 것을 확인할 수 있었다. 파란색으로 표시된 결과를 식 (7)과 같이 2차 함수로 나타내었다.

Fig. 6에서 빨간색 그래프는 10회 실험에서 얻은 실제 데이터의 평균값으로 그래프 끝부분에는 실제거리가 인식 범위(4 m) 밖이어서 데이터가 0으로 인식된 것을 확인 할 수 있다. 초록색 그래프는 중간에 노이즈가 발생한 경우를 나타낸다. 노이즈가 발생할 경우 그 부분의 지면의 intensity는 0으로 인식된다.

$$\mathrm{I} = -0.0035P_y{}^2 - 0.0968P_y + 225.65 \quad (7)$$

여기서 I는 intensity이며 P_y는 y축 pixel 값이다. 이후 지면의 범위 내에서 해당 픽셀의 Intensity 값이 식 (7)에서 계산한 I 값보다 작으면 I 값은 0으로 리턴 받아 화면에서 지면을 제거하는 필터를 설계하여 Fig. 7과 같은 결과를 얻었다. Fig. 7(a)는 바닥면 위에 놓은 물체를 포함하여 화면을 촬영한 후 깊이 데이터를 표현 한 경우 바닥면에 대한 깊이 데이터가 포함되어 명확하게 물체를 구별하기가 어려운 결과를 보여주며 Fig. 7(b)의 경우 제안된 필터를 사용하여 영상을 처리한 결과, 확연하게 바닥 위에 있는 물체의 거리 정보를 정확히 구할 수 있고 동시에 물체에 대한 인식이 가능함을 보여준다.

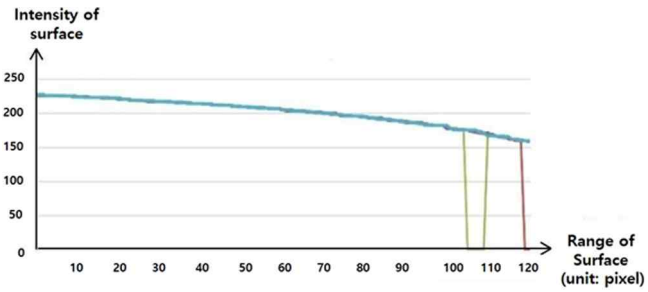


Fig. 6 Experiment result of surface's depth

2.4 물체구별(objects labeling)

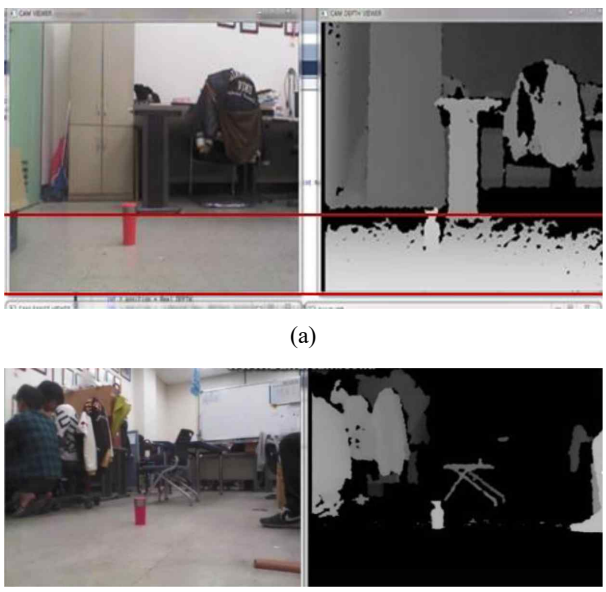
Fig. 8에서 물체인식 과정의 6개 단계를 보여준다. XTION pro live에서 받은 깊이 데이터를 기반으로 그레이 이미지(gray image)를 구성하고 지면 노이즈(Surface's noise)의 제거 및 메디안 필터(median filtering)을 적용했다.

겹쳐 있거나 붙어있는 물체들은 하나의 물체로 인식되기 때문에 물체들을 구별하기 위해 이진화 이미지(binary image)에서 케니엣지 이미지(canny edge image)로 제거하여 각 물체의 경계를 구분하는 결과 이미지를 얻는다. 이 결과 이미지에 블롭 레이블링(blob labeling)을 적용하여 물체들을 구분 해서 인덱스를 붙인다. 다음으로 물체에 대한 3차원 정보를 액션프로를 사용하여 계산하는 과정을 서술한다.

Fig. 9(a)는 실제공간에서 물체와 그것의 중심좌표를 나타내며, (X_o, Y_o, Z_o)는 물체(object)의 실제 위치를 나타낸다. Fig. 9(b)는 액션프로로 물체의 심도정보를 받아 구성한 영상이며 물체중심의 pixel의 한 점을 (u_o, v_o)라 표시한다. 식 (8-10)은 액션프로의 수평 시야각과 수직 시야각(58°), 그리고 윈도우의 너비와 높이를 (W_{width}, W_{height})라고 했을 때 특정위치의 픽셀(u_o, v_o)로 측정된 물체의 중심좌표를 얻는 과정이다[9].

$$X_{o'} = \frac{(Z_{o'}\ \tan(\theta_v))}{W_{width}}\ u_o \quad (8)$$

$$Y_{o'} = \frac{(Z_{o'}\ \tan(\theta_v))}{W_{width}}\ u_o \quad (9)$$



(a)

(b)

Fig. 7 (a) Surface noise, (b) Output image after surface noise removal

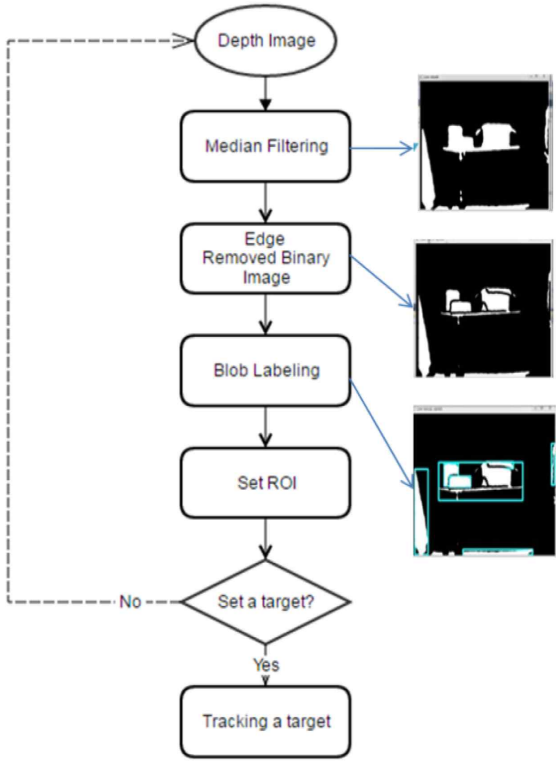


Fig. 8 Process of recognizing separated objects

$$Z_{o'} = Z_o \tag{10}$$

본 연구에서는 실험을 통해 액션프로에서 측정된 (X_o, Y_o, Z_o) 좌표와 실제 물체의 실제 위치좌표($X_{o'}$, $Y_{o'}$, $Z_{o'}$)가 대략 2% 미만 (0~4 mm)의 오차를 가짐을 확인하였다.

2.5 물체추적

물체인식은 깊이 이미지를 기반으로 하는데, 액션프로의 심도값 (depth value)의 인식범위는 0.4 m~4.0 m이므로 카메라와 물체와의 거리가 0.4 m 이내이면 깊이 값을 인식하지 못한다. 깊이 인식 범위 밖에서도 물체를 인식하기 위해 RGB화면에 물체추적 알고리즘을 적용했다.

Fig. 10(a)는 심도 이미지(depth image)에 블롭 레이블링을 적용시켜 물체를 인식한 화면이다. Fig. 10(c)는 인식한 물체의 경계를 기준으로 관심영역(ROI: region of interest)을 설정하고 관심영역내의 휴(hue)값의 히스토그램이다. Fig. 10(b)는 RGB화면에 Fig. 10(c)의 히스토그램을 바탕으로 밀도분포를 나타낸 것이다. Fig. 10(d)는 Fig. 10(b)의 밀도분포의 0~2차 모멘트를 계산하여 목표물의 위치를 추적하는 화면이다.

Table 1은 인식한 물체의 영상 데이터를 수집하여 실제 길이 및 물체의 크기를 측정하여 비교한 것이다. 물체의 3차원 위치 데이터는 0.5% 미만 즉 2 mm 이하의 오차를 보인다. 반면에 지면의 노이

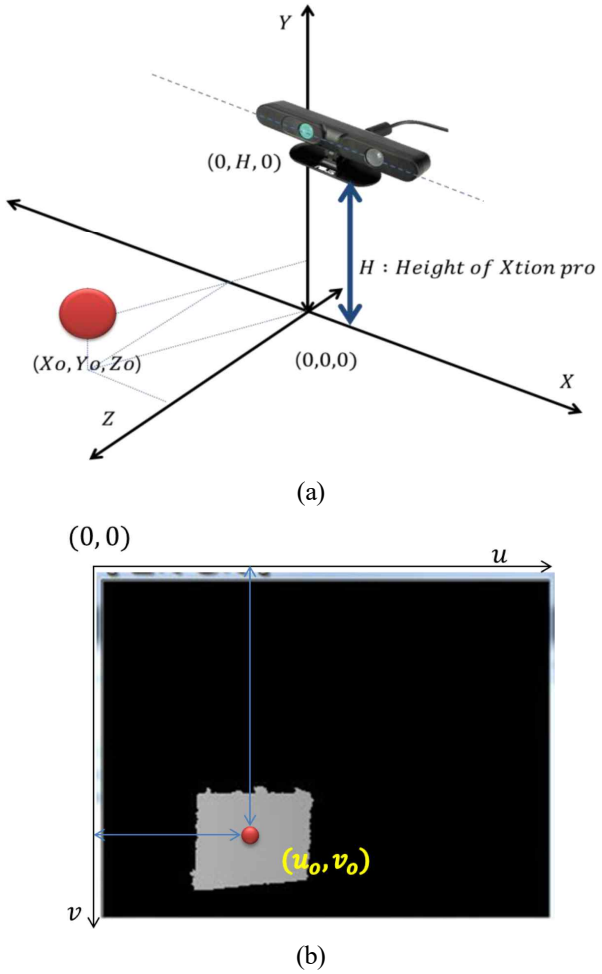


Fig. 9 (a) Reference coordinates and object 3D information, (b) Xtion pro depth image and its center value

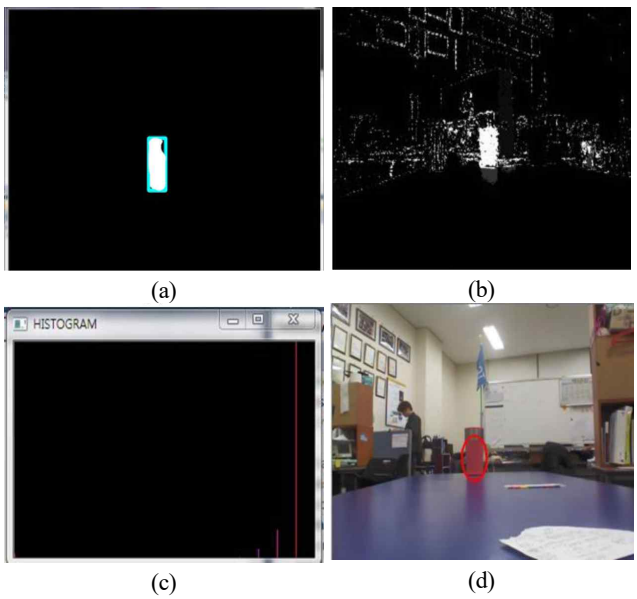


Fig. 10 The entire system processes

Table 1 Data reliability (unit: mm)

	x position	y position	z position	Width	Height
1	-406	59	899	79	111
2	-405	57	897	76	116
3	-404	57	895	76	112
4	-405	57	897	76	110
5	-405	59	897	76	112
6	-404	57	895	77	110
7	-405	59	897	79	112
8	-405	59	897	76	110
9	-405	57	897	73	115
10	-405	57	897	76	110
Average	-404.9	57.8	896.8	76.4	111.8
True value	-400	58	900	76	116
Error (%)	1.22	0.34	0.35	0.52	3.62

즈를 제거하면 일부 데이터 손실로 물체 길이 정보는 5% 이하의 오차를 보였다.

3. 가상공간 설계 및 모션 동기화

3.1 가상공간 설계

시야 확보가 불가능한 환경하에서 실제 로봇에게 목표물 추적 및 파지를 지시하기 위하여 가상공간을 구축하였다. 이 가상공간을 통하여 사전에 입력된 환경 정보를 표시하여 로봇의 조정이 가능하게 할 수 있다.

Fig. 11은 본 논문에서 가상 공간을 구축하기 위한 Unity3D 엔진을 구축하여 작성된 프로그램의 모습이다. Unity3D는 3D 컨텐츠와 같은 소프트웨어 분야, Window, Mac, Wii, iPhone 등과 같은 다양한 플랫폼에서 개발 가능하다. 가상공간 환경을 구축하고 블루투스, TCP/IP 소켓 무선 통신을 이용하여 현실정보를 가상공간에 mapping하는 시나리오1과 미리 현실정보가 구축된 가상공간에서 로봇을 제어하는 시나리오2를 설계하고 그 속에서 가상 객체를 제어하여 실제로봇과의 모션을 동기화하였다.

3.2 시나리오 1 구조

시나리오 1은 앞서 기술한 대로 로봇에 있는 영상 장치를 통해 물체나 장애물을 인식하고 그 데이터를 토대로 가상 공간에 객체들을 생성하면서 가상공간을 현실공간과 mapping하는 방법이다.

Fig. 12는 이러한 일련의 운영 절차에 대해 나타낸다. 여기서 영상 장치가 보는 물체에 대한 위치와 크기 값을 (x, y, z, height, width)의 순으로 받는다. 이는 Fig. 13의 위 그림과 같은 물체의

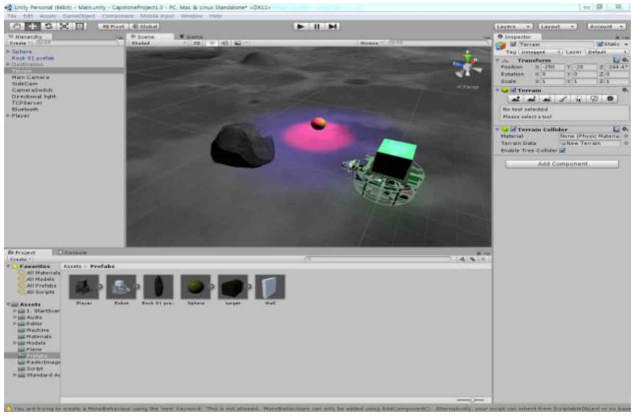


Fig. 11 Virtual environment creation using Unity3D

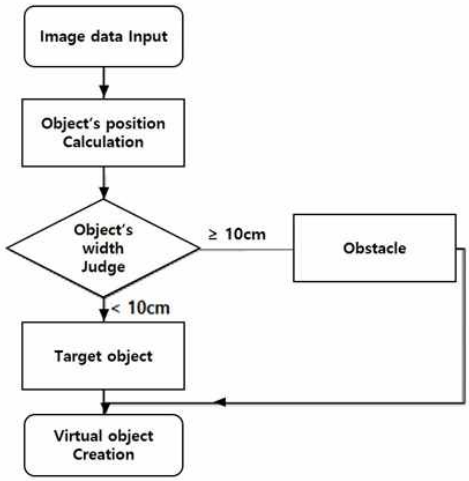


Fig. 12 Operation procedure of scenario 1

좌표값을 계산하기 위하여 로봇 좌표계를 기준으로 한 값과 월드좌표를 기준으로 한 값으로 각각 표현할 수 있다. 로봇과 독립적으로 가상 공간에 물체를 표현하기 위해 물체를 월드 좌표계로 변환해야 한다. 수식 (11)에서는 인식된 물체를 영상의 해당 위치에 생성하기 위한 좌표 변환식을 표현하였다.

$$P_{xyz} = T_{robot}\, P_{noa} \tag{11}$$

여기서 P_{noa}는 로봇 좌표 계 기준의 물체의 위치 값(x, y, z)이며 P_{xyz}는 월드 좌표계를 기준으로 한 위치값 x', y', z' 이다. T_{robot}는 월드 좌표계와 로봇 기준 좌표계 간의 변환 행렬이다. 영상 장치를 기준으로 측정된 물체의 위치를 월드 좌표계로 변환하기 위해 영상 장치가 있는 로봇의 위치와 방향(T_{robot})을 파악하고 둘을 연산하여 물체의 절대 위치(P_{xyz})을 월드 좌표 계 기준으로 계산하여 물체의 위치(x', y', z')을 파악하고 그 위치에 현실 물체와 동일시 할 수 있도록 가상 이미지를 생성한다. T_{robot}의 각 값은 현재 로봇의 월드

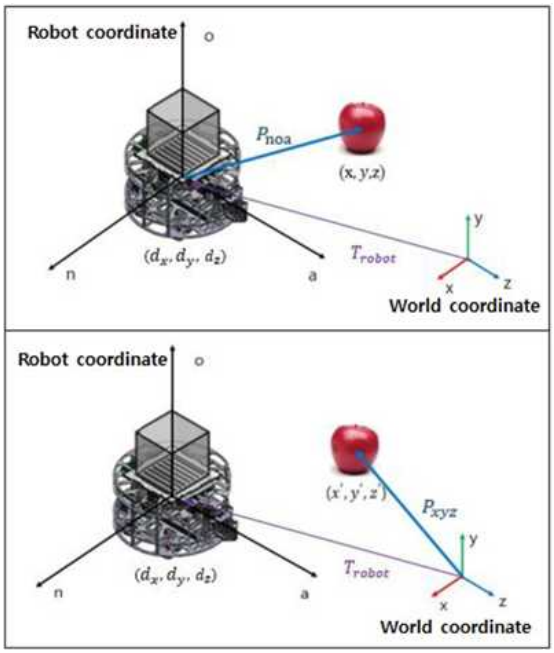


Fig 13 The object's position w.r.t robot coordinate and world coordinates

좌표 기준으로 중심위치값과 로봇의 회전각을 외부에서 또는 내부의 센서로부터 측정하여 계산한다.

그리하여 영상 데이터에서 위치를 알게 되면 그 위치에 가상 객체를 생성한다. 이때 로봇이 잡을 수 있는 물체는 10 cm 내외의 폭이며 10 cm 폭을 넘는 물체는 장애물로 인식하여 물체와 장애물을 구별하여 Fig. 14와 같이 이미지를 생성한다.

이와 같이 로봇이 현실 공간을 이동하면서 보게 되는 물체들에 대해서 가상 공간이 현실 공간과 일치될 수 있도록 이미지를 생성한다. 이러한 이미지들을 활용하여 사용자가 로봇을 제어하면서 동시에 장애물을 피할 수 있게 하고 물체를 잡을 수 있도록 한다.

3.3 시나리오 2 구조

시나리오 2는 시나리오 1를 통해 구축된 가상공간과 객체들을 이용하거나 미리 현실공간을 가상공간에 구현하여 그 가상 이미지를 통해 사용자가 실제 로봇을 제어하는 시스템이다. 이는 영상 장치가 사용될 수 없는 경우 및 시야가 확보되지 않는 지역 등에서 로봇을 제어할 수 있는 방안이 된다. 그러므로 시나리오 2에서는 영상 데이터의 통신을 활용하지 않고 로봇의 구동부 만을 제어하여 장애물을 피하고 지정 물체를 가져오는 동작을 수행한다.

3.4 가상 객체의 모션 데이터

가상 로봇은 다양한 동작이 가능하게 하기 위하여 6자유도의 움직임을 가질 수 있는 형태로 설계하였다. 그러나 제작된 실제 로봇은 바퀴 구동 방식의 로봇 메커니즘을 채택하여 구동부는 2자유도

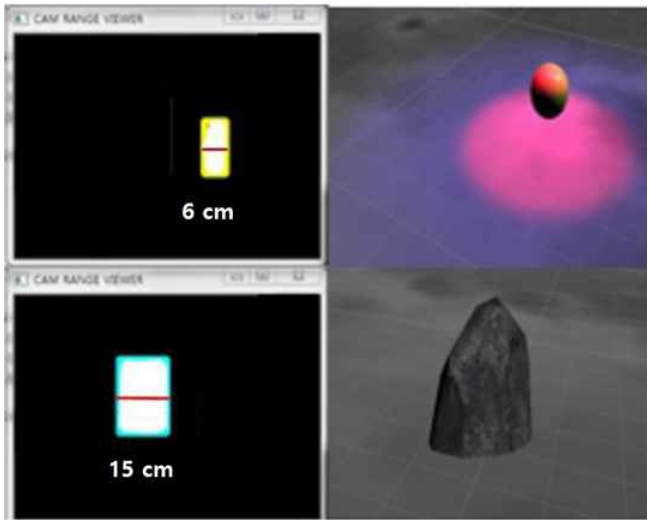


Fig. 14 Object and obstacle creation according to the object width

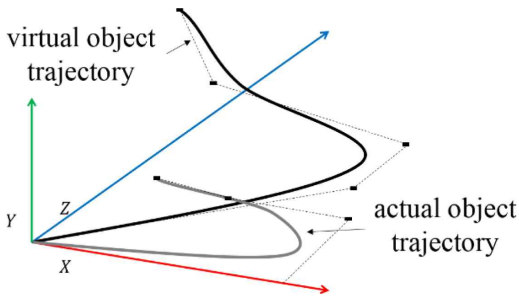


Fig. 15 Flat projection conversion

(Z, Yaw) 동작만이 가능하게 설계 되어있기 때문에 가상 객체의 6자유도 데이터(x, y, z, roll, pitch, yaw)를 바로 로봇에 적용할 수 없다.

따라서 가상 로봇의 위치 및 방향 데이터를 실제 로봇에게 mapping하고 모션 동기화를 이룰 수 있도록 하기 위해서는 6자유도 데이터를 변환하는 과정이 필요하다.

가상 객체의 3차원 움직임을 Fig. 15처럼 평면 정 투영 변환을 통해 2차원으로 변환하여 실제 로봇의 바퀴 구동으로 움직일 수 있도록 한다. 빨간 선은 가상 객체의 이동 경로이고 이를 정 투영 변환(y=0)으로 현실 객체가 녹색 선을 따라 움직일 수 있도록 데이터를 변환한다. 그에 따른 통신 프로토콜은 Table 2와 같이 정의하고 가상 객체의 자유도의 변화 값으로 설계하였다.

가상 객체의 6자유도 값을 단위시간마다 버퍼에 저장 후 여기서 y 데이터를 제외하고(y=0) 나머지의 변화 값 즉 직선 이동값 2개, 회전값 3개(Δx, Δz, $\Delta roll$, $\Delta pitch$, Δyaw)를 4 byte에 맞춰 전송한다. 기존의 가상 객체의 위치 데이터는 디버깅을 위해 text 파일로 따로 저장한다.

이 데이터를 단위 시간마다 로봇에게 전달하여 로봇의 위치와

Table 2. Protocol structure of the motion of virtual robot

Name	Start bit	Position (x, z)	Roll	Pitch	Yaw	End bit
Byte	1	8(4,4)	4	4	4	1

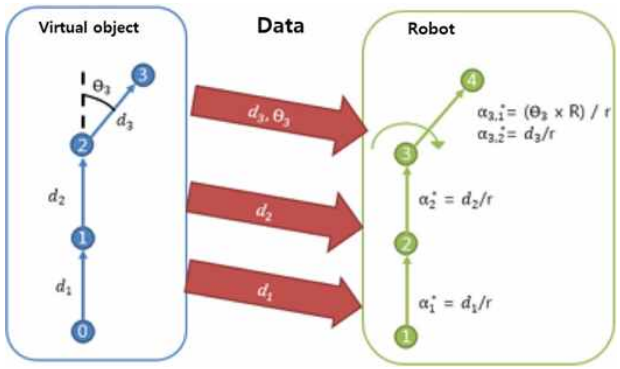


Fig. 16 Displacement and angle per unit time

방향이 가상로봇과 동기화가 될 수 있도록 구동부를 제어한다. 실제 로봇이 컨트롤될 수 없는 나머지 움직임(roll, pitch)에 대해서는 예외처리를 통하여 즉 roll과 pitch 각도는 0으로 처리하여 두 로봇간 모션에 동기화를 이룰 수 있도록 하였으며 실시간으로 무선 네트워크를 통해 데이터를 주고 받는 구조를 갖게 설계하였다.

Fig. 16은 가상 객체의 단위 시간당 위치와 방향을 전달하여 그에 대한 이동변위 값을 단위 시간마다 실제 로봇에게 전달하는 과정을 보여준다. 위치 0에서 위치 1로 가는 경로에서는 직선 변위 값인 d_1을 전달하기 위하여 실제 로봇이 전진할 수 있도록 가상공간과 실제공간과의 크기 비인 r로 나눈 값인 d_1/r 로 전달한다. 위치 2에서 위치 3 구간은 변위 값인 d_3와 회전 값 θ_3을 전달하고 이 값들을 r로 나누어 최종으로 실제 로봇에게 제자리 회전 후 전진할 수 있도록 지시하여 해당 모터를 제어하였다. 이와 같이 가상로봇의 단위 시간당 변위를 계산하여 로봇이 그를 따라 움직일 수 있도록 하여 위치와 방향을 동기화한다. 이 값들이 정확히 실제로봇에서 구현되도록 엔코더 값을 피드백 받아 위치 및 방향에 대해 PID 피드백 제어하여 수행하였다.

3.5 실험 결과

Table 3에서는 본 연구에서는 실험을 통해 액션프로에서 측정된 물체의 ($X_{o'}$, $Y_{o'}$, $Z_{o'}$)좌표와 실제 물체의 실제 위치좌표 (X_o, Y_o, Z_o)차이를 보여주고 있다. 물체의 3차원 위치 데이터는 0.5% 이하의 즉 실제 거리 2 mm 이하의 오차를 보인다. 반면에 지면의 잡음을 제거하면서 영상처리를 수행한 결과 물체의 길이 정보는 4% 오차인 4 mm 이하의 오차를 보인다. 발생하는 오차에 대해 몇 가지 원인을 알아보았다. 첫째, 센서의 오차이다. 액션프로의 내장된 IR센서의 경우 적외선 센서처럼 깊이 값을 측정하기 때문에 매번

Table 3 Data reliability (unit: mm)

Cases	$X_{o'}$	$Y_{o'}$	$Z_{o'}$	Width	Height
1	-405	59	899	79	111
2	-405	57	897	76	116
3	-405	57	895	76	112
4	-405	57	897	76	110
5	-405	59	897	76	112
6	-404	57	895	77	110
7	-405	59	897	79	112
8	-405	59	897	76	110
9	-405	57	897	73	115
10	-405	57	897	76	110
Average	-404.9	57.8	896.8	76.4	111.8
True value	-405	58	900	76	116
Error (%)	0.03	0.34	0.35	0.50	3.62

Table 4 Position comparison between the virtual robot and the actual robot

Unit time (0.5 s)	Virtual object (cm)	Robot (cm)	Error (%)
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	15.28	14.80	3.19
6	39.59	38.36	3.12
7	64.33	62.41	2.99
8	89.06	87.32	1.96
9	100	97	3

일정하지 않고 어느 정도의 오차가 발생한다. 둘째, 지면의 잡음을 제거하고 물체 구분을 위해 경계를 제거하면서 인식되는 물체의 크기가 실제보다 적게 나온다. 본 연구에서는 약 1~4 mm정도로 오차 값이 작아 로봇이 목표위치로 이동하는데 큰 문제가 되지 않았다.

본 논문에서 제안한 가상 로봇과 실제 로봇간의 동기화는 지정된 위치와 방향을 실제로봇이 얼마나 정확히 갈 수 있는지를 평가하는데 있다. 이를 테스트하기 위해 1 m 직선 주행과 복합적인 궤적에 대해 테스트를 진행하였다.

먼저 1 m 직선 주행 결과는 Table 4처럼 나타나는데 이는 1 m 주행하면서 각각의 단위 시간마다 현실 로봇과 가상 로봇의 위치를 비교한 것이다. 평균 오차 2.852%를 보이고 있는데 이는 로봇의 구동 부 백래쉬와 이동 중인 로봇의 기구 부가 흔들림으로 인하여 구동 모터에 작용되는 부하가 변화하면서 오차가 발생한 것이며

우수한 동기화가 구현됨을 알 수 있다. 거리를 더 늘려 실험한 경우도 큰 차이가 없음을 확인하였다.

다음은 직진주행과 회전이 포함된 복합적인 궤적에 대한 테스트를 진행하였다. 가상 로봇의 단위 시간마다 위치에 따른 궤적과 실제 로봇의 엔코더 값에 따른 궤적을 계산해보니 Fig. 17과 같이 나타났다.

Fig. 17에서 보면 직선 구간보다 회전하는 구간에서 실제로봇간에 더 큰 오차가 존재한다. 이는 실제로봇의 회전 축이 가상 객체의 회전 축과 정확히 일치하지 못해 오차가 발생한 것으로 판단된다. 양 바퀴의 평행도가 확보되어 직선 운동에는 오차가 적었으나, 회전 축의 차이로 인해 회전하는 구간에서는 더 큰 오차를 띄게 되었다.

궤적 끝 부분에서 오차가 많이 발생하는 것은 물체를 잡는 동작이 진행할 시 잠시 동기화를 해제하고 로봇이 자동으로 물체를 잡는 과정에서 정확히 잡기 위해 움직인 것이다. 이 부분에서는 물체 잡는 동작이 완료된 후 가상 객체의 위치를 현실 로봇의 위치변화에 맞춰 이동시키는 방식으로 오차를 제거할 수 있다.

다음으로 영상처리를 하는 임베디드 컴퓨터 사양의 한계로 인해 실시간으로 통신하는데 어려움이 존재하였다. 본 연구에서는 데이터 통신 속도를 정확히 판단할 수 없기에 로봇의 이동속도를 30 cm/s으로 제한하여 신속성보다 정확성에 초점을 맞추어 연구를 진행하였다.

무선 통신을 통해 가상 및 실제 로봇간의 데이터를 실시간으로 송수신 시 MCU가 모터를 구동시키는 시간에서 지연이 발생하였고 시간이 흐를수록 누적되어 가상로봇과의 오차가 증가함을 알 수 있다. 하드웨어 지연이 발생하면서 정확한 시점 파악에 오류가 발생할 수 있다. 이를 극복하기 위하여 이동 구간별로 위치를 업데이트하여 오차를 최소화하여 시각적으로는 두 로봇간의 오차를 감지할 수 없을 정도로 시스템을 구축하였다.

Fig. 18에서는 가상 로봇(왼쪽 그림)과 실제로봇 (오른쪽 그림)이 가상공간에서 물체를 집는 동시에 실제로봇이 동기화 되어 실제 물체를 집는 전 과정을 보여주고 있다. 전체적으로 가상로봇과 실

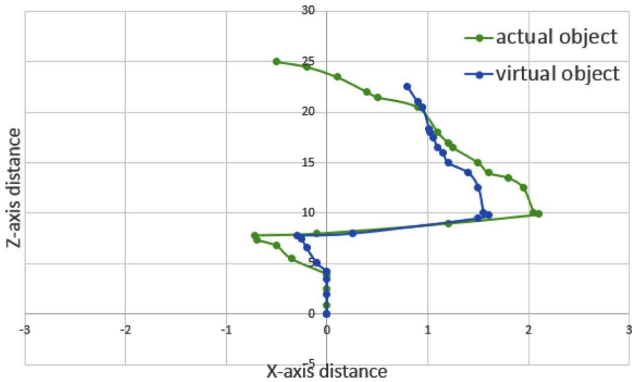


Fig. 17 Trajectories of the virtual object and the actual robot for linear and rotational travelling (unit: cm)

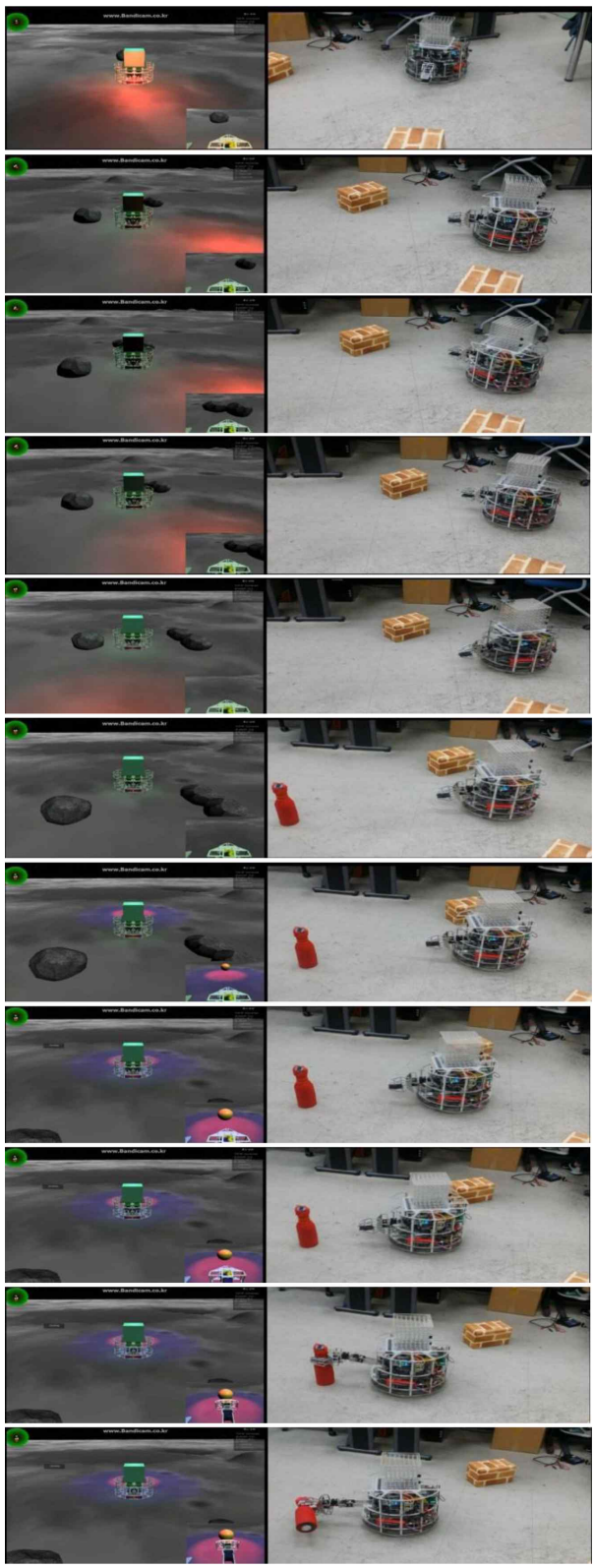


Fig. 18 Demonstration of moving and grapping an object for virtual robot (left) and real robot (right)

제로봇간 동작에서 동기화가 이루어져 영상으로부터 실제 로봇을 구동시키거나 반대로 가상 로봇의 위치를 지정하면 실제 로봇이 도달함을 알 수 있다.

4. 결 론

본 논문에서는 가상 로봇과 실제 로봇간의 움직임을 동기화하여 mapping 시킴으로써 영상획득이 불가한 환경에서도 실제 로봇을 가상로봇을 통하여 제어할 수 있으며, 또한 가상로봇에서 환경변화를 제공할 경우 실제 로봇이 인지하여 여러 유형의 움직임을 가질 수 있게 개발하였다. 하드웨어와 소프트웨어를 통신으로 연결하여 같은 궤적을 움직이면서 동기화되는 것을 확인할 수 있었다. 평지를 기준으로 시스템을 개발하였으며, 가상현실 환경과의 모션 동기화 과정에서 기술적인 문제점을 확인하고 이를 개선하는 방안도 수립하였다. 차후 로봇의 절대적인 위치를 인식할 수 있는 기법을 추가하여 두 로봇간의 오차를 더 개선할 수 있는 지속적인 연구가 필요하다. 이렇게 가상 로봇의 움직임 및 가상객체의 설정과 실제 로봇 움직임 및 실제 객체를 잡는 과정을 동기화하는 시스템은 지속적인 연구를 통하여 게임 분야를 넘어 극한 환경이나 가정 등 다양한 분야에 적용 가능할 것으로 사료된다.

후 기

이 연구는 서울과학기술대학교 교내 연구비 지원으로 수행되었습니다.

References

[1] Lee, S. Y., 2009, Develope of Robot Game Kernel for Linkage Virtual Space and Real Space, Korean Soc. of Computer Game, 19 161-166.

[2] Noh, J. S., Lee, G. H., Jung, S., 2008, Control of a Mobile Pendulum System for a Boxing Robot Game, Int. Conf. on Smart Manufacturing Application, 408 -412.

[3] Nakano, E., Asada, M, Tadokoro, S., Osuka, K., Nagai, K., Masutani, Y., Kitano, H., 2000, The Outline of The International Robot Games Festival, IEEE Int. Conf. on Robotics and Automation 2000, 820-825.

[4] Azuma, R. T., 1997, A survey of augmented reality, Presence: Teleoperators and Virtual Environments, 355-385.

[5] Aris, A.J., Lim, I., Iqbal, D., Noor, P., Ismail, S. B. M., Kamal, N. A., 2003, Design of Entertainment Mobile Robot, Student Conf. on Research and Development 2003, 298 -303.

[6] Hwang, S.W., Lee, S.H., Lee, H. K., 2008, Multi-hop Based Real-time Streaming Service for Emergent Robot, Korean Inst. of Information Scientists and Engineers, 205-206.

[7] Berg, J. V. D., Lin, M., Manocha, D., 2008, Reciprocal Velocity Obstacles for Real-time Multi-agent Navigation, IEEE Int. Conf. on Robotics and Automation 2008, 1928-1935.

[8] Hai, H., Bin, L., BenXiong, H., Yi, C., 2011, Interaction System of Treadmill Games Based on Depth Maps and CAM-Shift, 2011 Int. Conf. on Comm. Software and Networks, 219-222.

[9] Seo, H. K., Kim, J. C., Jung, J. H., Kim, D. H., 2013, Wearable Robot System Enabling Gaze Tracking and 3D Position Acquisition for Assisting a Disabled Person with Disabled Limbs, Trans. Korean Soc. Mech. Eng. A, 37:10 1219-1227.