세계초일류
이공계대학

2019 Spring Robotics Project

Term Project Report

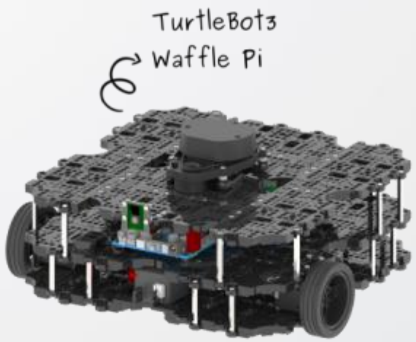
2019. 6. 27

Myungjin Park (Integrated course, IIT, GIST)

Geonhyup Lee (MS cousre, IIT, GIST)

Yongwoo Lee (Undergrad course, GIST)

Jinseo Hong (Undergrad course, GIST)



Gwangju Institute of Science and Technology

CONTENT

Overview

- Turtlebot 3 with Open Manipulator

Onboard sensors : 3D gyroscope, 2D LiDAR

Open manipulator : 4 DoF RRRR robotic arm.

Reference : http://www.robotis.com/

- Overhead 2D camera

RGB color detect

Methodology

Detection & Navigation

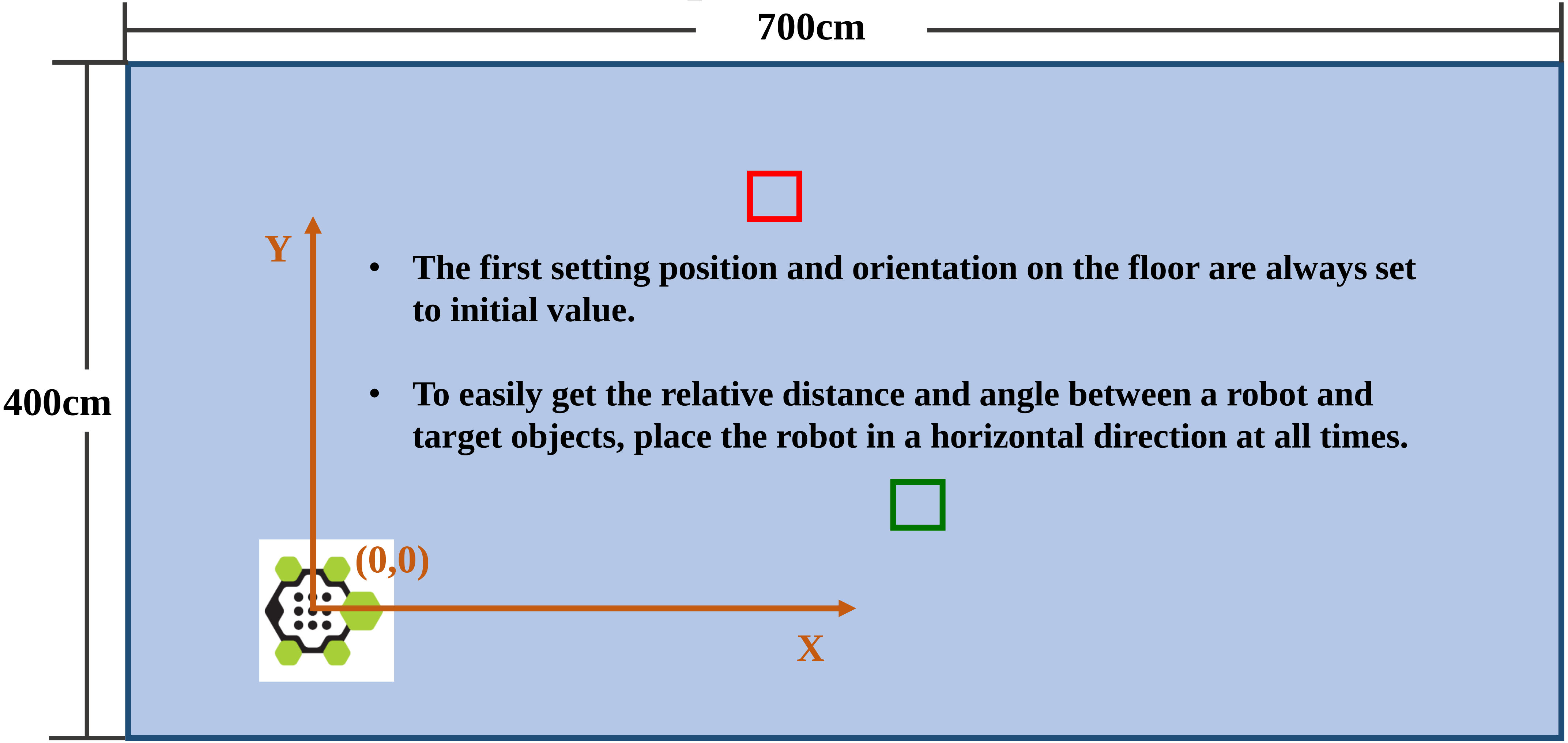
- Past - roughly find location of robot and other object with overhead 2D camera & precise task(manipulate, localization of object) with 3D depth sensor
- But there was a problem to associate 3D depth sensor with ROS and it was hard to solve in a month. So we changed our strategy to simplify the task
- Now – Find location as precisely as we can, keep calibrating robot's path and line up robot and object with 2D overhead camera. After alignment

Manipulation

- First we tried to use a package given in ROS. But it also had a problem about connection, we solved inverse kinematics of Open Manipulator by hand.
- The pick & drop task is performed by inverse kinematic method with known position data of objects.

Results – Task 1

Set initial position & orientation



- The first setting position and orientation on the floor are always set to initial value.
- To easily get the relative distance and angle between a robot and target objects, place the robot in a horizontal direction at all times.

Results – Task 2

- **Steps of Object Detection**

We used Overhead USB camera and OpenCV.

1. **Restrict the area of detection**

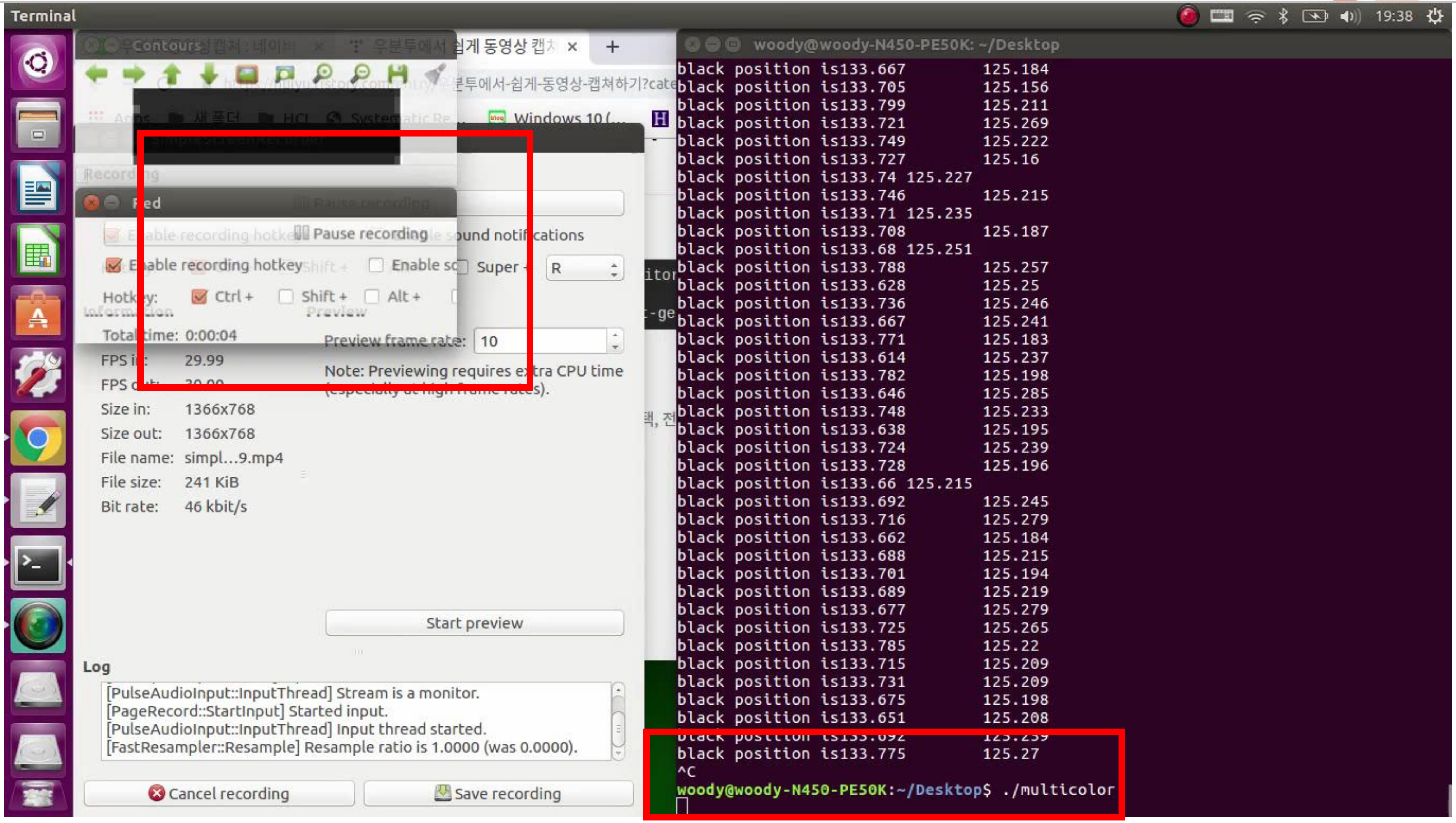
 (To prevent the effect of edge sides.)

2. **Detect the objects first and remember each object’s position**

 (Using Color filter in OPENCV)

3. **Detect the turtlebot’s position continuously and calculate distances from objects to turtlebot.**

Results – Task 2



Results – Task 2

- **Detectable objects**

It was easy to detect color objects, but it was difficult to detect white and transparent objects.

Before competition, we can easily calculate 100 pixel = 1.15m, so we can convert the scale from pixel to meter.

Then we send meter information to Turtlebot.

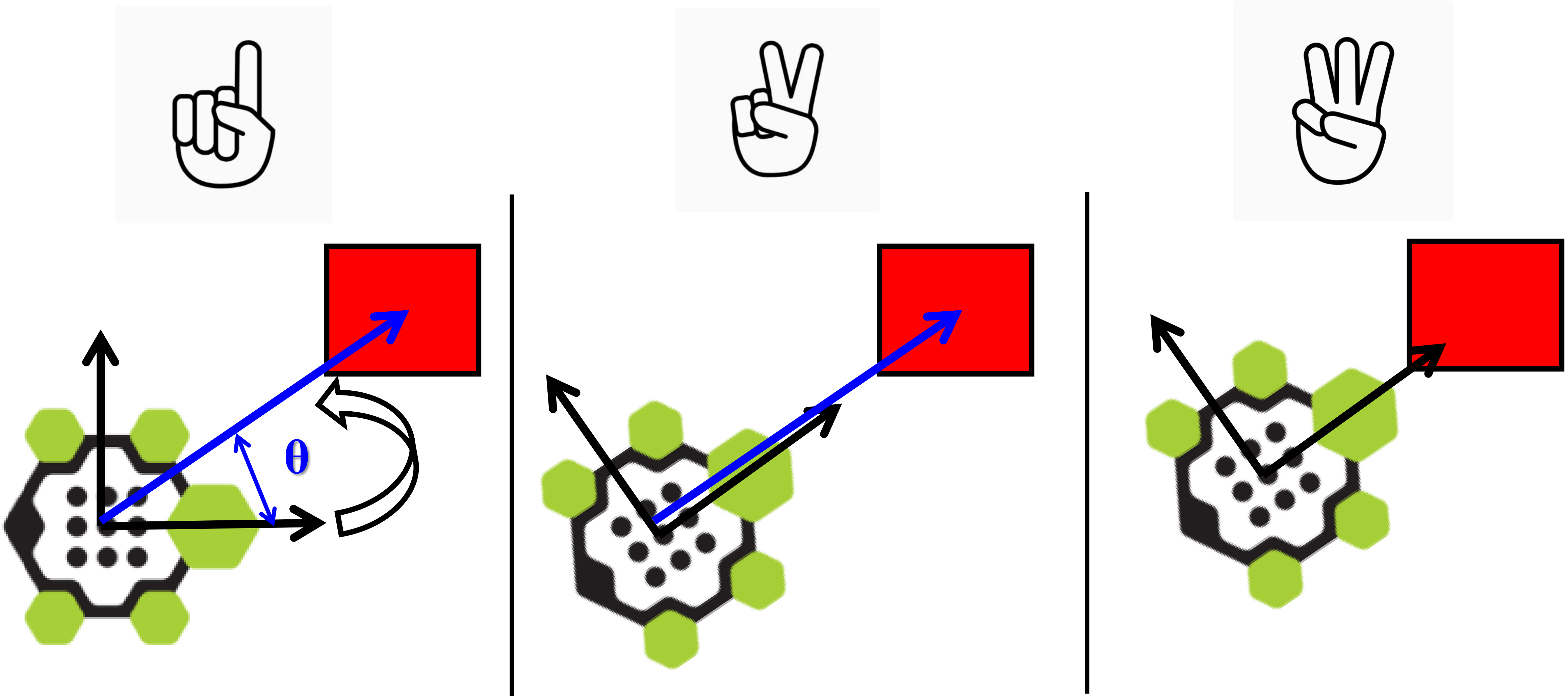
We tried to use Canny Edge Detector in OpenCV, but it has problems.

(1) Camera was too far from objects. Sometimes objects have no difference with other noises.

(2) Square blocks were also detected as circles. It seems to happen because of distance from camera.

However, it might be possible if we spend some time to find best edge thresholds.

Results – Task 3



Results – Task 3

- If x-axis matches with the direction of object, Turtlebot can reach object using only x-direction moving.

=> We match the x-axis with direction of object by rotating yaw-axis.

- From the rostopic open_manipulator_with_tb3_tools/odometry, we can read odometry of turtlebot.

However, it is quaternion based orientation expression. Thus, we need to convert Quarterternion to Euler.

We can calculate y-axis tilt using x,y,z,w in Quaternion.

$$\begin{bmatrix}\phi \\ \theta \\ \psi\end{bmatrix} = \begin{bmatrix}\operatorname{atan2}(2(q_0 q_1 + q_2 q_3), 1 - 2(q_1^2 + q_2^2)) \\ \operatorname{asin}(2(q_0 q_2 - q_3 q_1)) \\ \operatorname{atan2}(2(q_0 q_3 + q_1 q_2), 1 - 2(q_2^2 + q_3^2))\end{bmatrix}$$

```
// yaw (z-axis rotation)
double siny_cosp = +2.0 * (q.w() * q.z() + q.x() * q.y());
double cosy_cosp = +1.0 - 2.0 * (q.y() * q.y() + q.z() * q.z());
yaw = atan2(siny_cosp, cosy_cosp);
```

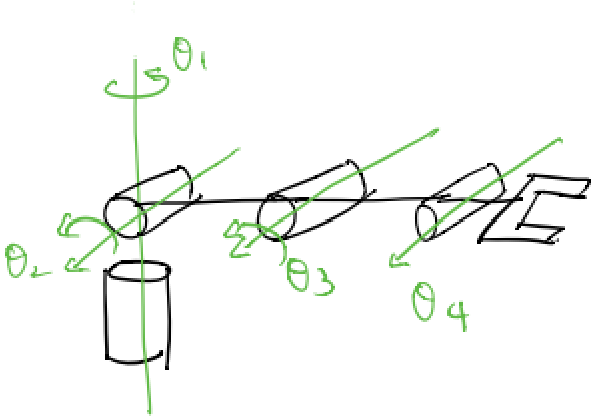
Results – Task 3

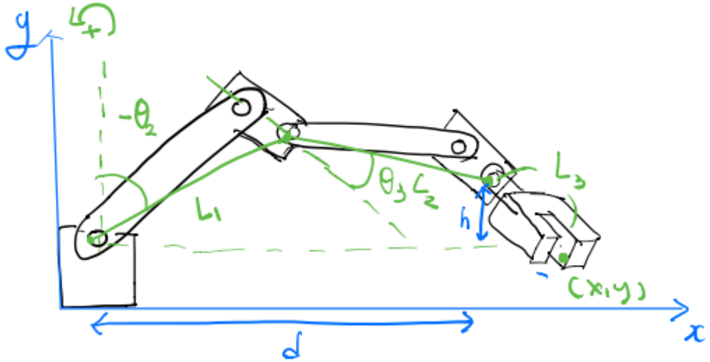
1. Rotate until matching Turtlebot’s face with object direction

2. Angle compensation while reaching the object
3. Angle compensation while reaching trashbin.

Results – Task 4

Arm Manipulation 수식





$$d^2+h^2 = L_1^2+L_2^2+2L_1L_2\cos\left(\frac{\pi}{2}+\theta_3\right)$$

$$\theta_3 = \sin^{-1}\frac{L_1^2+L_2^2-d^2-h^2}{2L_1L_2}$$

$$-\theta_2 = \frac{\pi}{2}-\left(\cos^{-1}\frac{L_1+L_2\cos\left(\frac{\pi}{2}-\theta_3\right)}{\sqrt{d^2+h^2}}+\tan^{-1}\frac{h}{d}\right)$$

$$\theta_2 = \cos^{-1}\frac{L_1+L_2\cos\left(\frac{\pi}{2}-\theta_3\right)}{\sqrt{d^2+h^2}}+\tan^{-1}\frac{h}{d}-\frac{\pi}{2}$$

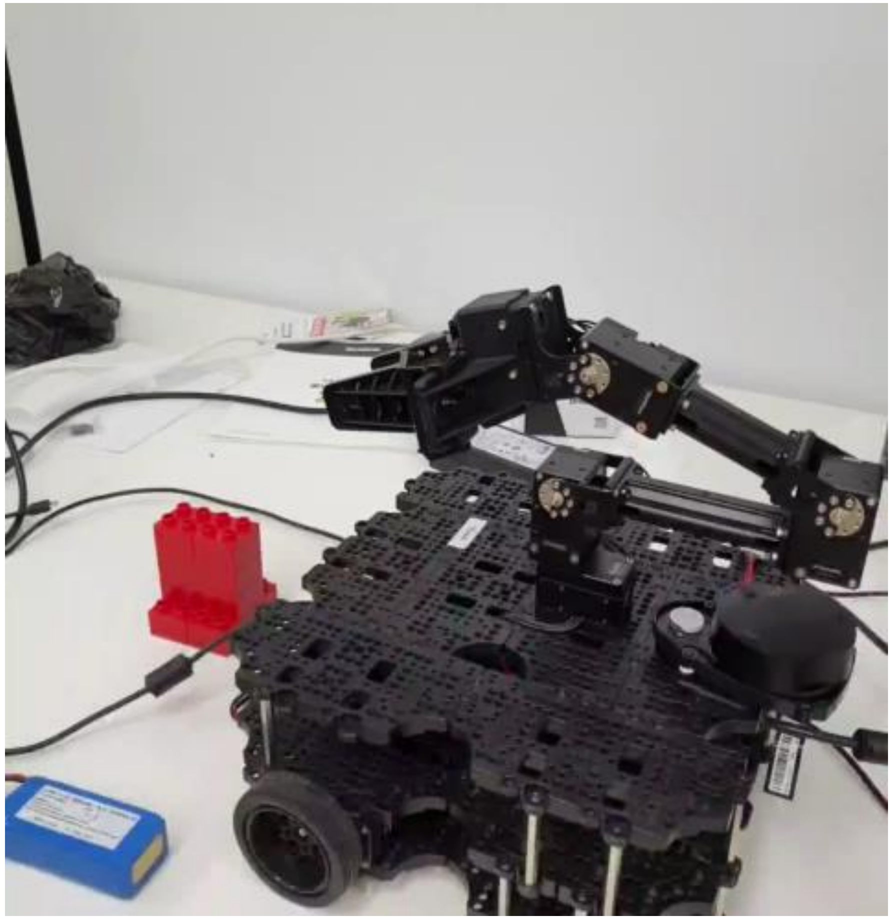
$$d = x-L_3\cos\theta_3, \quad h = y_3-L_3\sin\theta_3$$

2-Dimension equation 사용
총 4개의 joint중 실제로 제어하는 joint의 개수는 3개로 2개의 joint로 position에 도달하도록 inverse kinematic을 구현하였다.

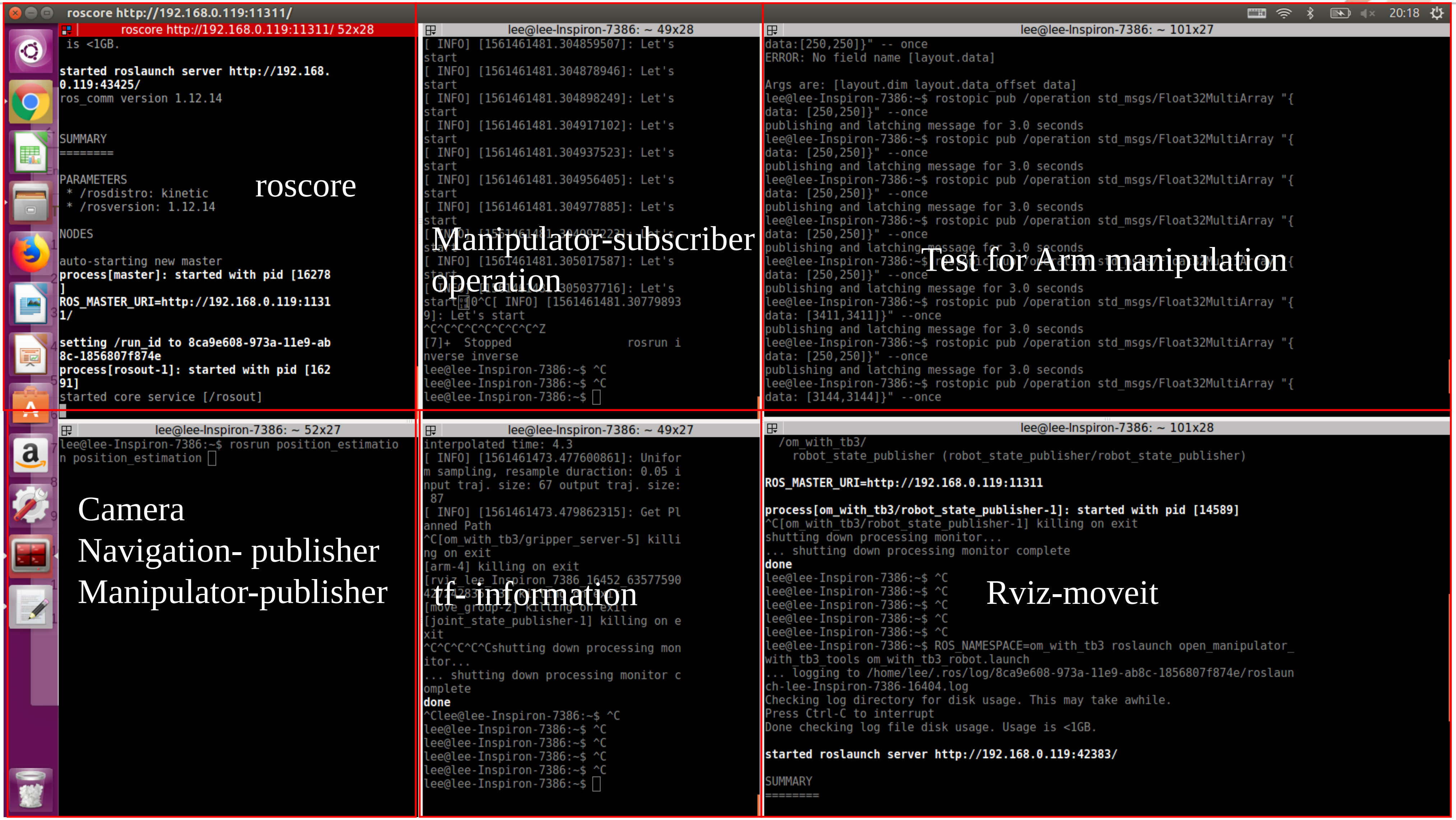
Using 2-Dimension equation
Only three of four joint is actually controlled.
Two joints for position and the other is for picking angle.
So, we made inverse kinematic about two joint for position.

Results – Task 4

Arm Manipulation 영상



Appendix


roscore http://192.168.0.119:11311/
roscore
Manipulator-subscriber
operation
Test for Arm manipulation
Camera
Navigation- publisher
Manipulator-publisher
tf- information
Rviz-moveit

Appendix

Software

세계 초일류 이공계 대학

- Thank you -

GIST

Gwangju Institute of Science and Technology