경희대학교
KYUNG HEE UNIVERSITY

Field painting robot using ROS-based DQN reinforcement learning and PID control

팀 구성 : 김정윤 (2018100670), 방지호 (2018100694)

Human-Robot Interaction Laboratory

Contents

01. 과제 설계 배경 및 필요성

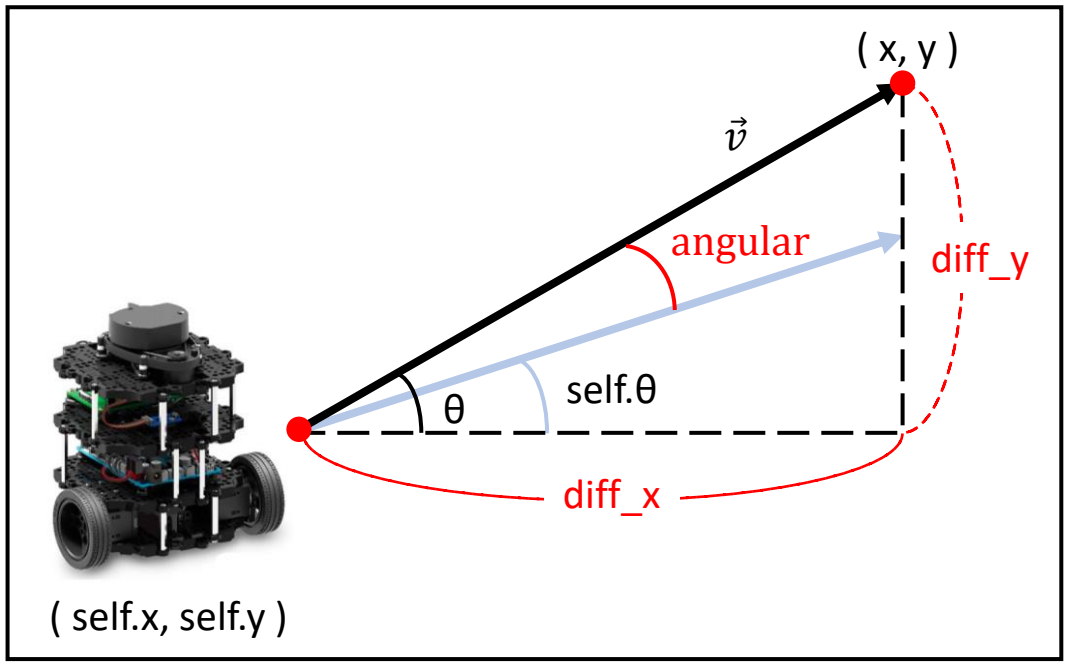
모든 아스팔트 도로에는 차선이 존재하며, 색칠부터 길이 측정까지 대부분 인간의 손에 의해 그려짐. 몇 백 km나 되는 도로와 경사면, 여름철 등 특정 환경이 차선 그리는 작업을 굉장히 어렵게 만듦.

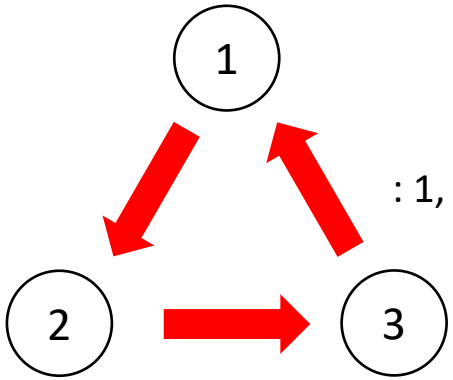
축구, 야구, 테니스와 같은 스포츠 계열, 최근 유행했던 오징어게임과 같이 필드를 만들어야 하나 매번 그리기 귀찮은 경우가 많음.

위와 같은 문제 해결을 위해 Field painting robot을 구현하고자 함. PID제어로 속도를 제어해 원하는 위치로 이동하는 waypoint 방법과 machine learning 중 DQN 강화학습모델을 이용해 이동하는 2 가지 방법에 대해 robot model을 설계함.

02. Waypoint : PID 제어 이론 및 구현

- **기본 이동 원리**





: 1, 2, 3번 동작이 계속 반복되며 이동

① 초기 위치 & 목표점을 바탕으로 필요한 값 계산

$$\begin{cases} \vec{v} = (x - self.x, y - self.y) = (\text{diff\_x}, \text{diff\_y}) \\ \hat{v} = \left(\dfrac{\text{diff\_x}}{\sqrt{\text{diff\_x}^2 + \text{diff\_y}^2}}, \dfrac{\text{diff\_y}}{\sqrt{\text{diff\_x}^2 + \text{diff\_y}^2}}\right) \\ \theta = \tan^{-1}\left(\dfrac{\text{diff\_y}}{\text{diff\_x}}\right) \end{cases}$$

② 제자리 회전 (angular = 0이 되도록)

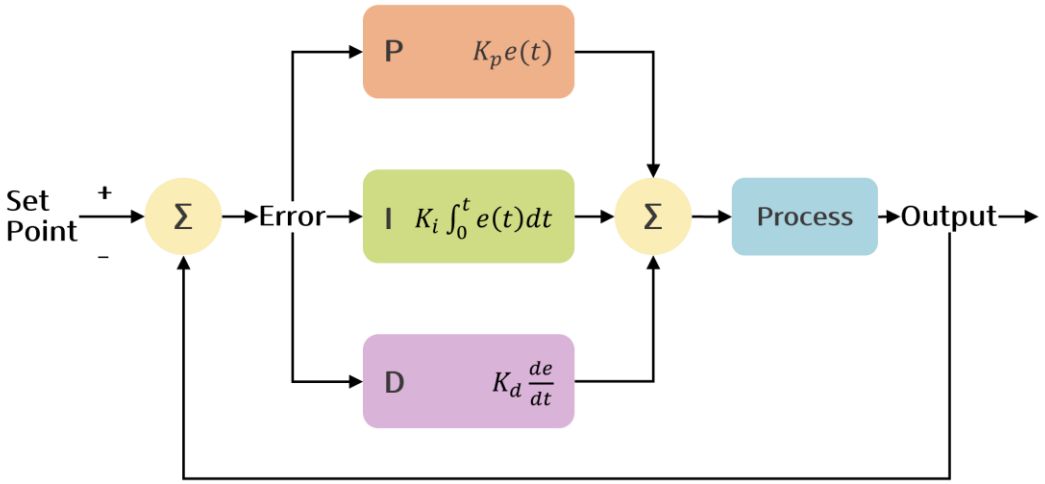
$$angular = \theta - \text{self.}\theta = \tan^{-1}\left(\frac{\text{diff\_y}}{\text{diff\_x}}\right) - \text{self.}\theta$$

③ 목표점으로 이동

$\text{linear} = \vec{v} \cdot \hat{v}$ ($\hat{v}$ = 초기에 구한 값, $\vec{v}$ = 매번 변화되는 값)

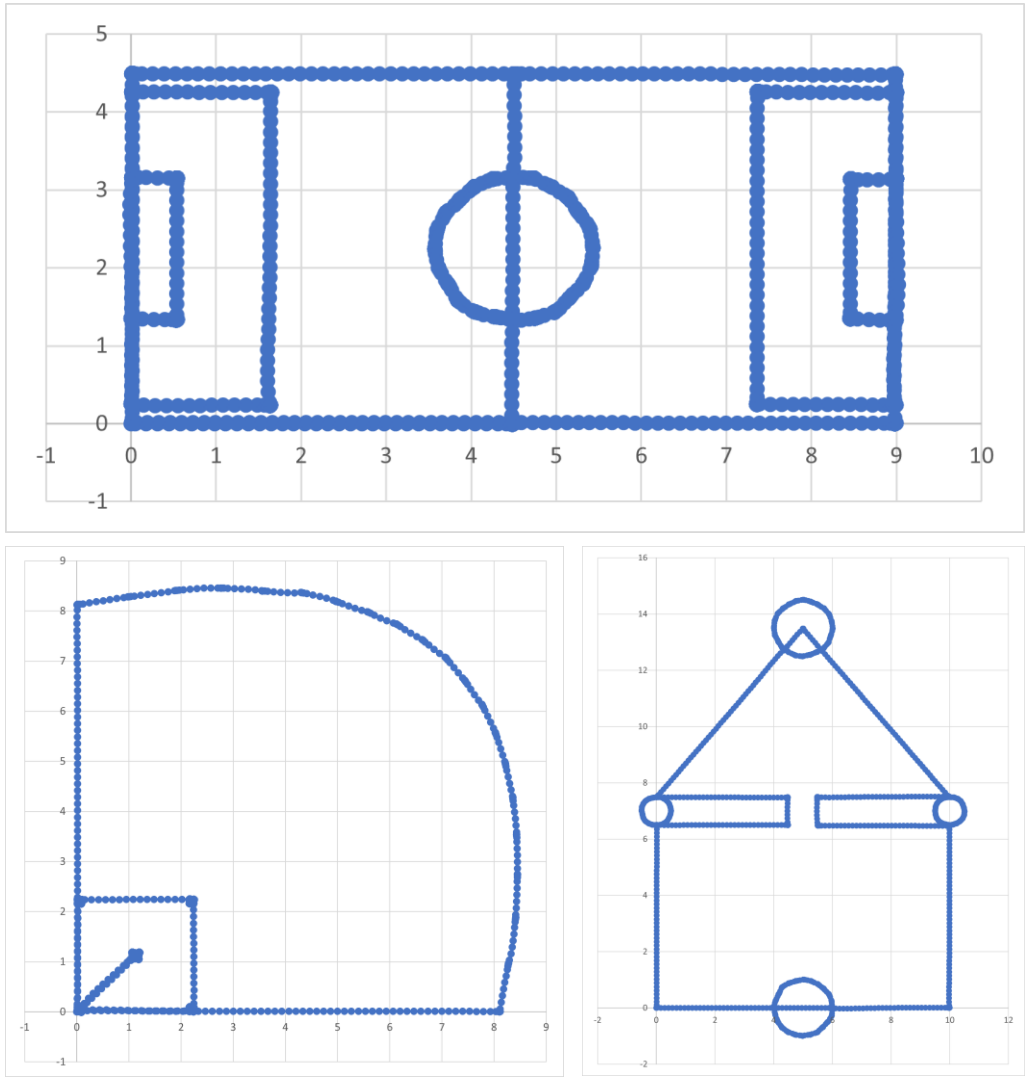
02. Waypoint : PID 제어 이론 및 구현

- **PID 제어기 설계**



[일반적인 PID 제어기]

: 이동시에 생기는 각도 오차를 보정하는데 사용함.

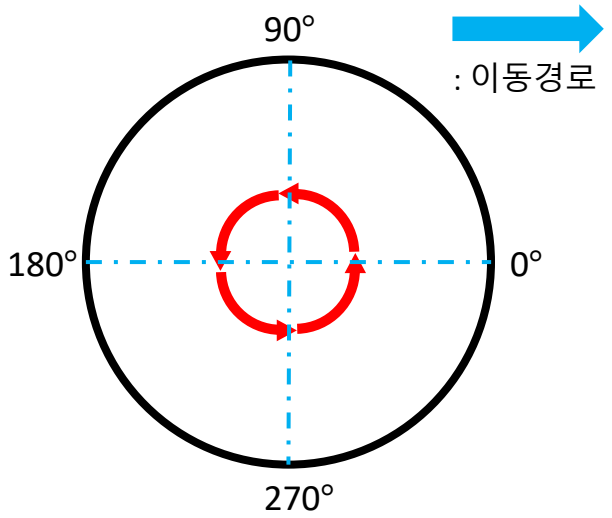


[Gazebo 환경에서 각각 축구장, 야구장, 오징어게임장을 그린 trajectory 그래프]

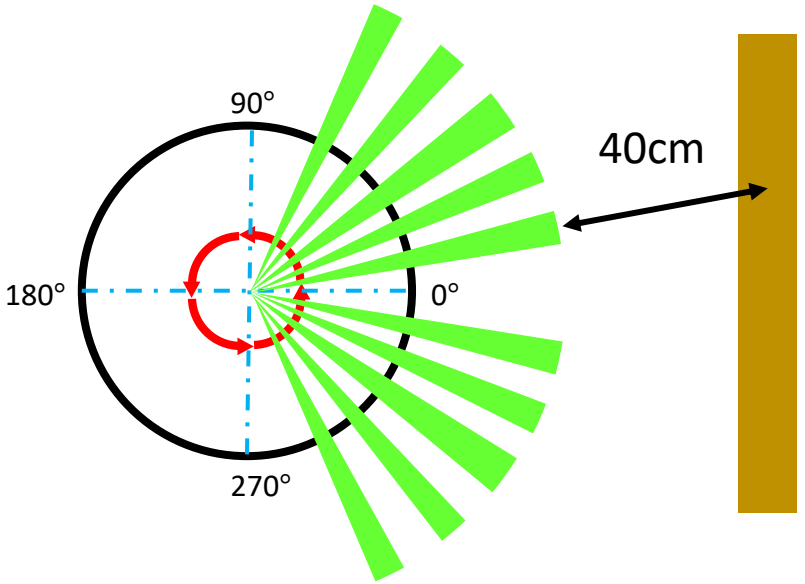
02. Waypoint : PID 제어 이론 및 구현

- **Lidar 장애물 탐지 기능 구현**

[360 Laser Distance Sensor LDS-01]



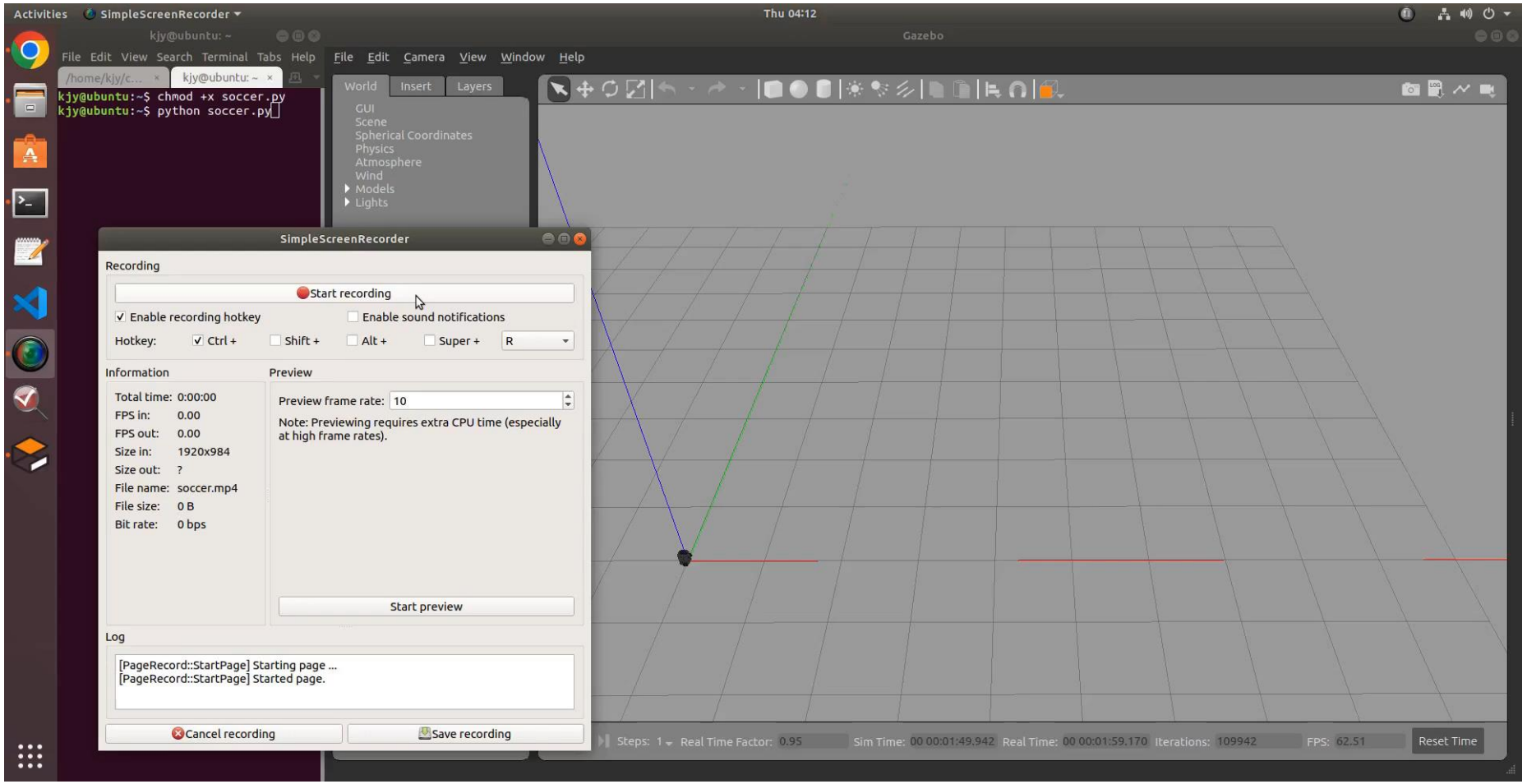
이동하며 바라보는 방향 기준 반시계 방향으로 0°~359°까지 총 360개의 거리 정보 데이터를 받아들임.

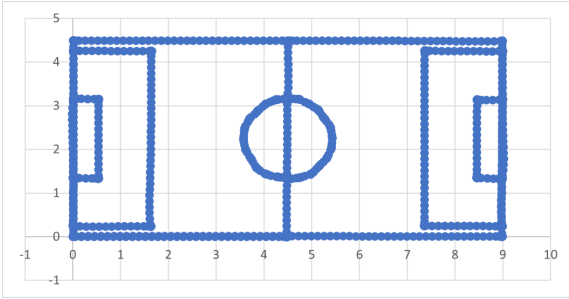


40cm보다 가까운 라이다 정보가 4개 이상 감지되면 정지하도록 함.

02. Waypoint : PID 제어 이론 및 구현

- Gazebo 상에서 구현한 waypoint simulation





03. Machine learning : DQN 강화학습 이론 및 구현

- **Comparison Classic Q-learning with Deep Q-learning**

Classic Q-learning

State s_t

Action a_t

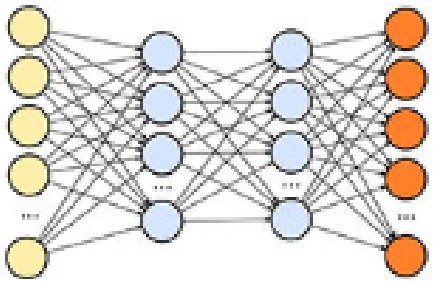
	a_1	a_2	a_3
s_1	$Q(s_1, a_1)$	$Q(s_1, a_2)$	$Q(s_1, a_3)$
s_2	$Q(s_2, a_1)$	$Q(s_2, a_2)$	$Q(s_2, a_3)$
s_3	$Q(s_3, a_1)$	$Q(s_3, a_2)$	$Q(s_3, a_3)$

→ Q-value $Q(s_t, a_t)$

Deep Q-learning

State s_t

Action a_t



→ Q-value $Q(s_t, a_t)$

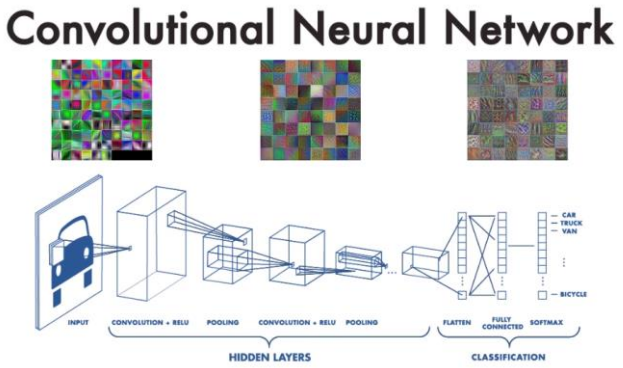
Bellman Equation :

$$Q(s, a) := r(s, a) + \gamma \max_a Q(s', a)$$

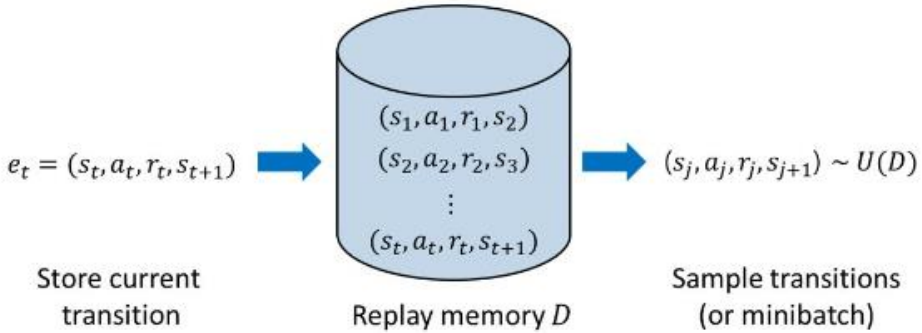
03. Machine learning : DQN 강화학습 이론 및 구현

- **Deep Q-Network (DQN)**

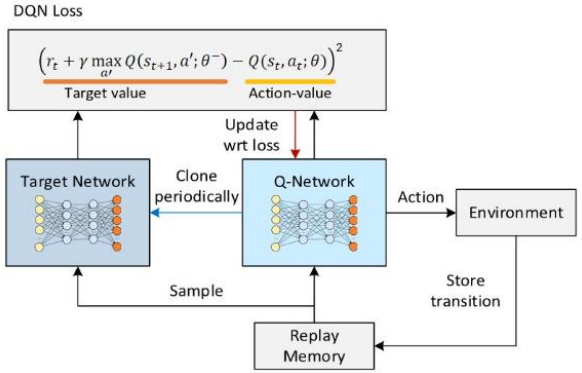
① CNN (convolutional neural network)



② experience replay



③ target network



$Q(s,a)=r(s,a)+\gamma max_a Q(s',a)Q(s,a)$

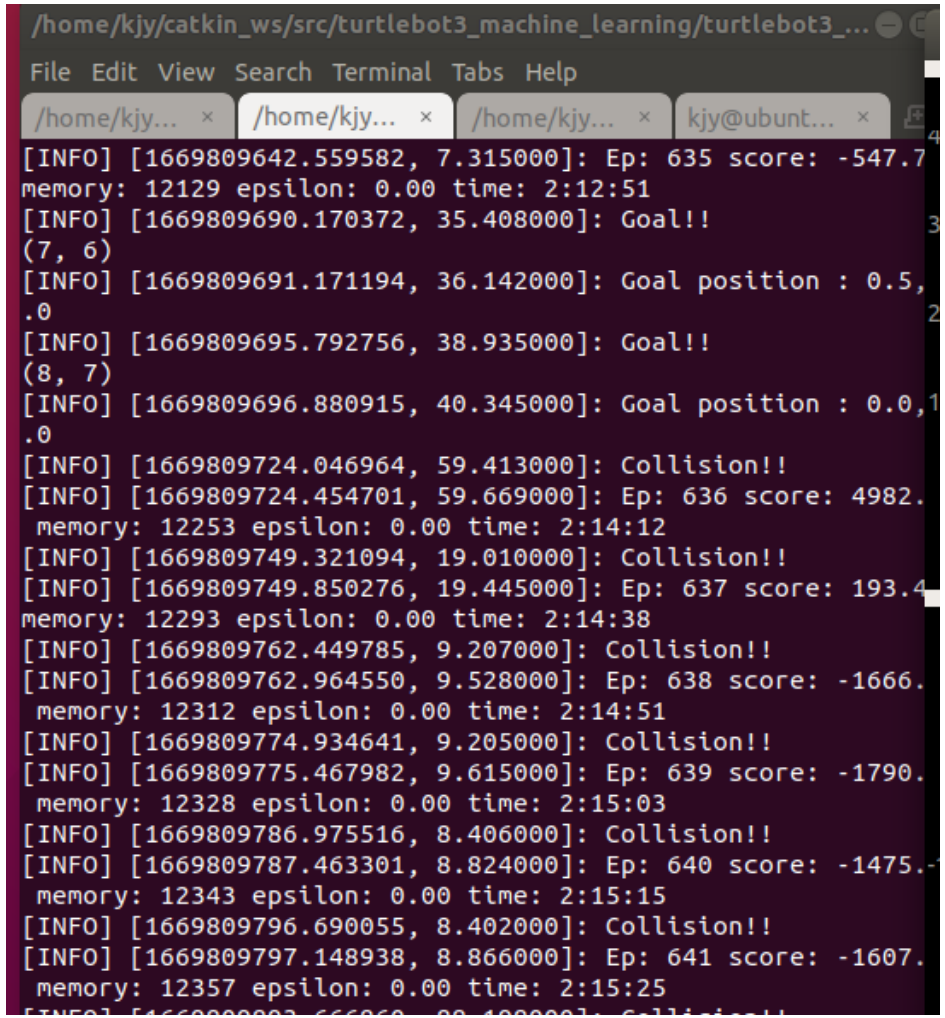
$Cost = [Q(s,a;\theta) - (r(s,a) + \gamma max_a Q(s',a;\theta))]^2$

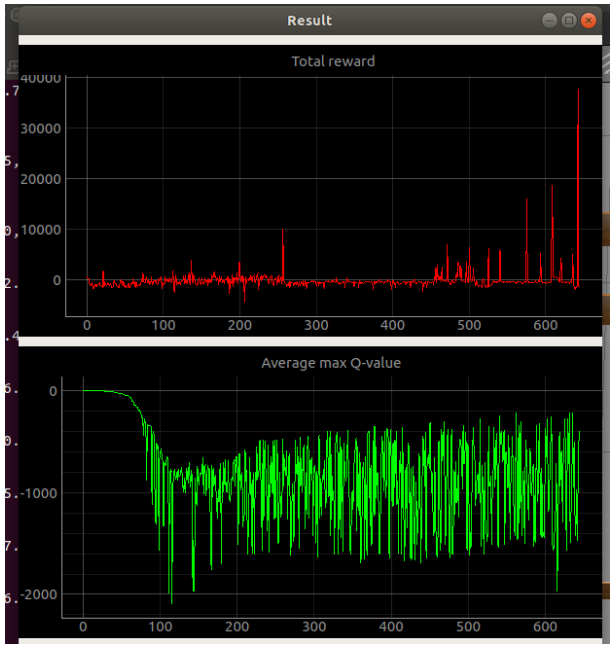
$MSE = \frac{1}{n}\sum_1^n (y_i - y_i')^2$: 평균 제곱근 오차

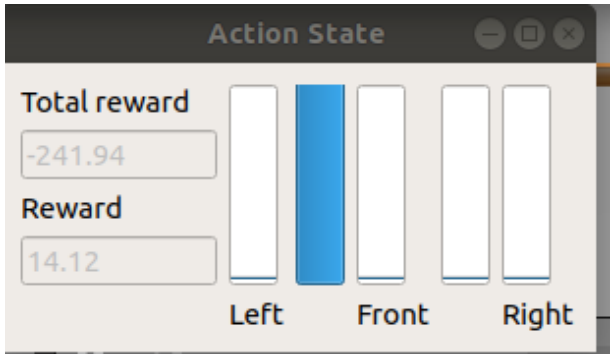
$Q(s',a;\theta)$: Q-target

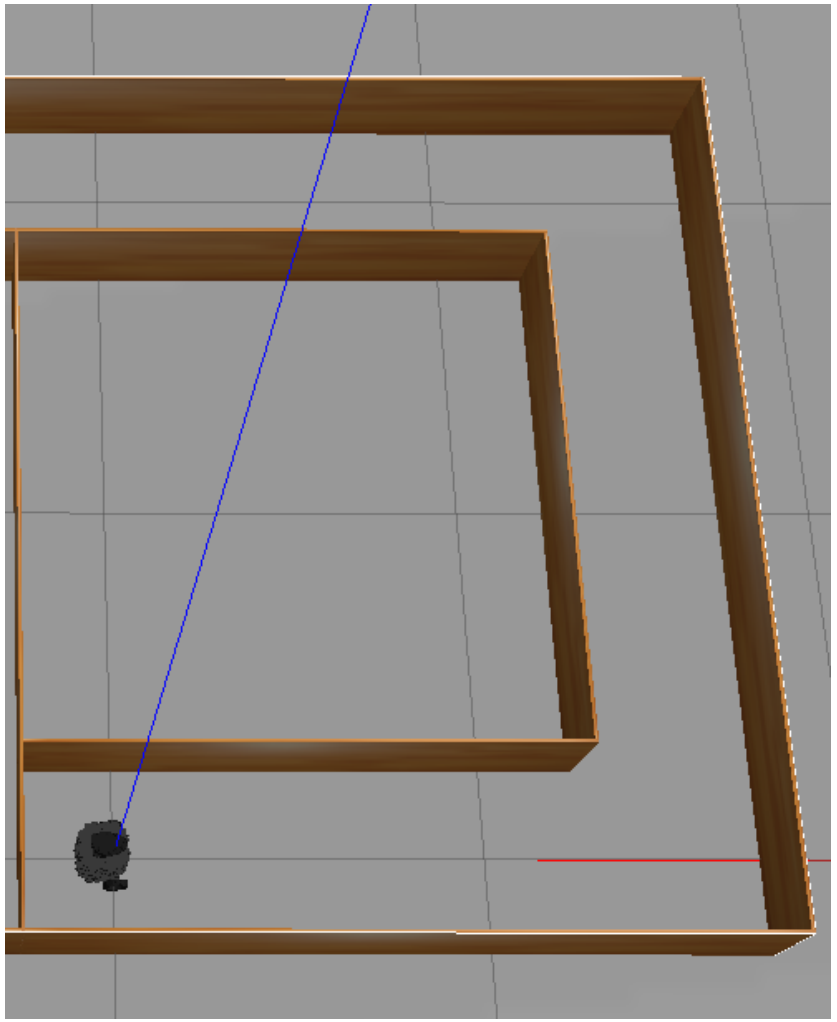
03. Machine learning : DQN 강화학습 이론 및 구현

- **Gazebo simulation composition**



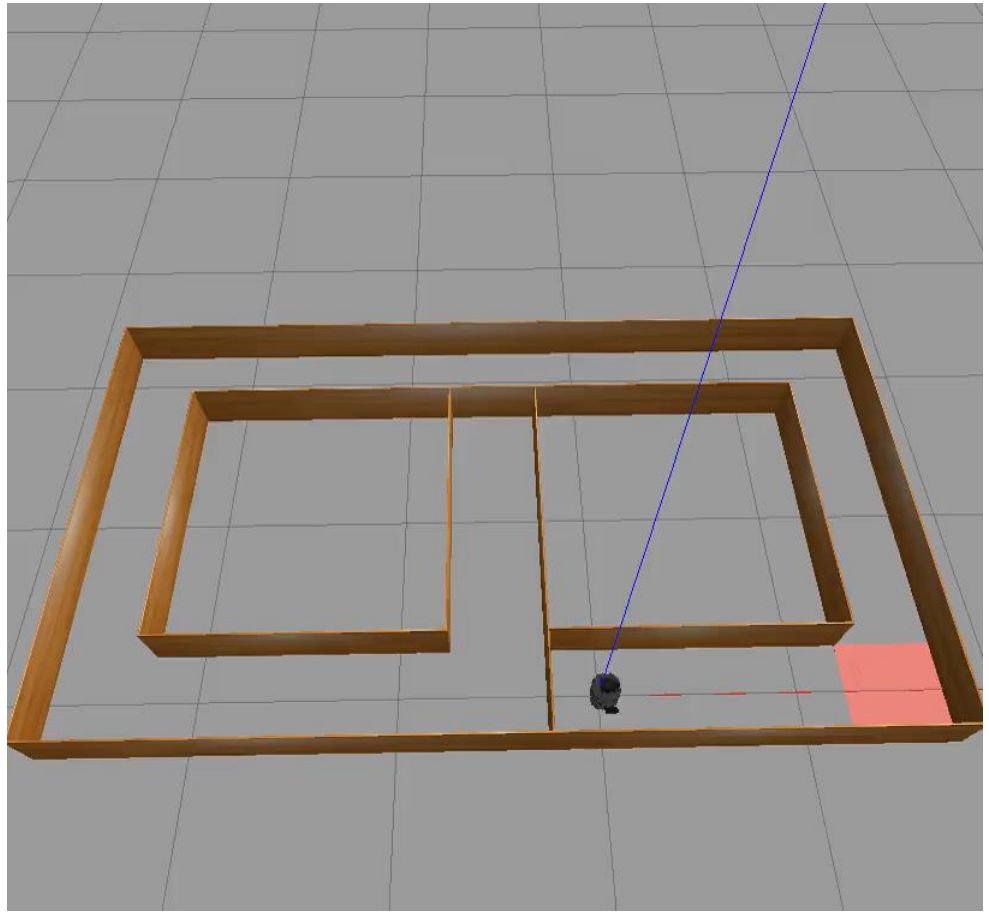




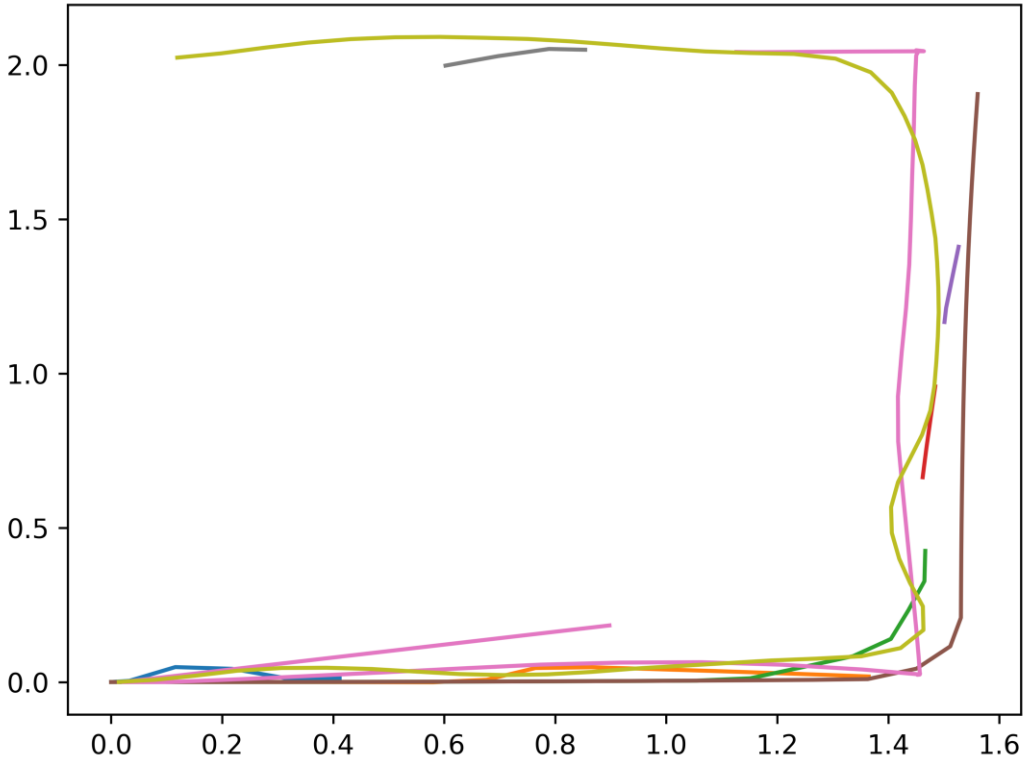


03. Machine learning : DQN 강화학습 이론 및 구현

- **Gazebo 상에서 구현한 DQN simulation (Low Learning)**



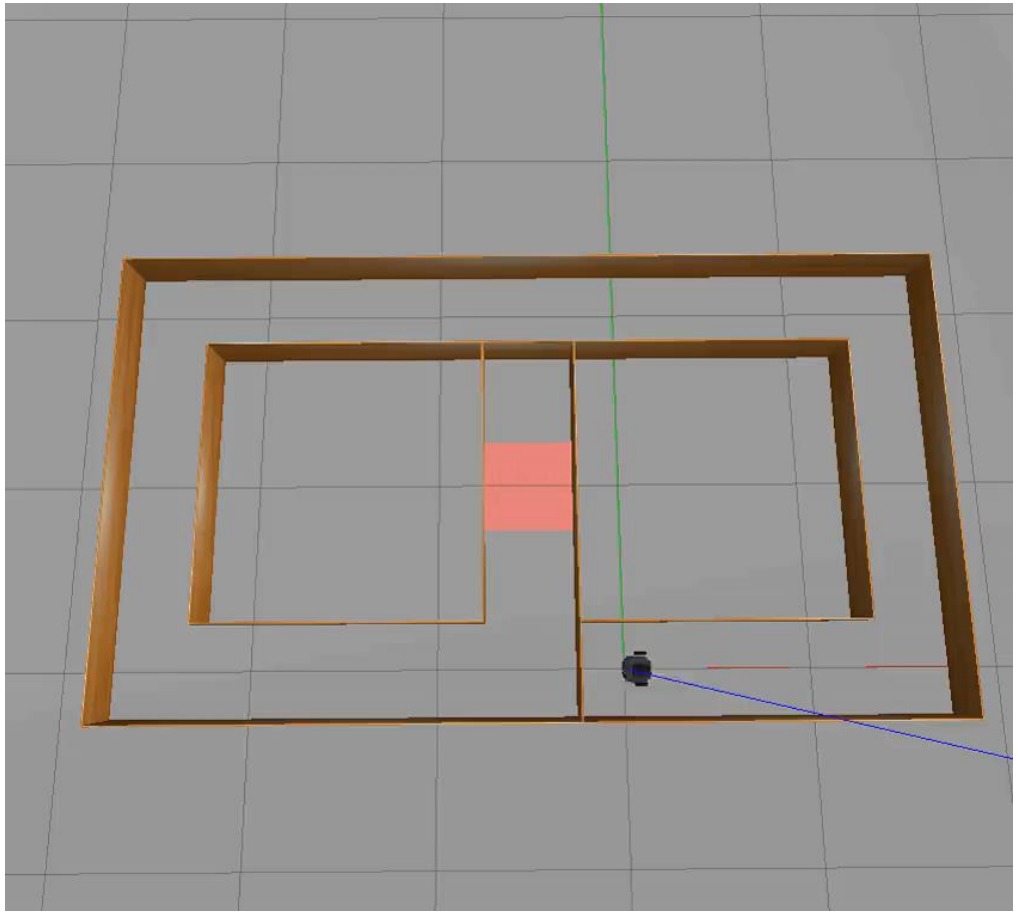
[Turtlebot3 DQN 학습 영상]



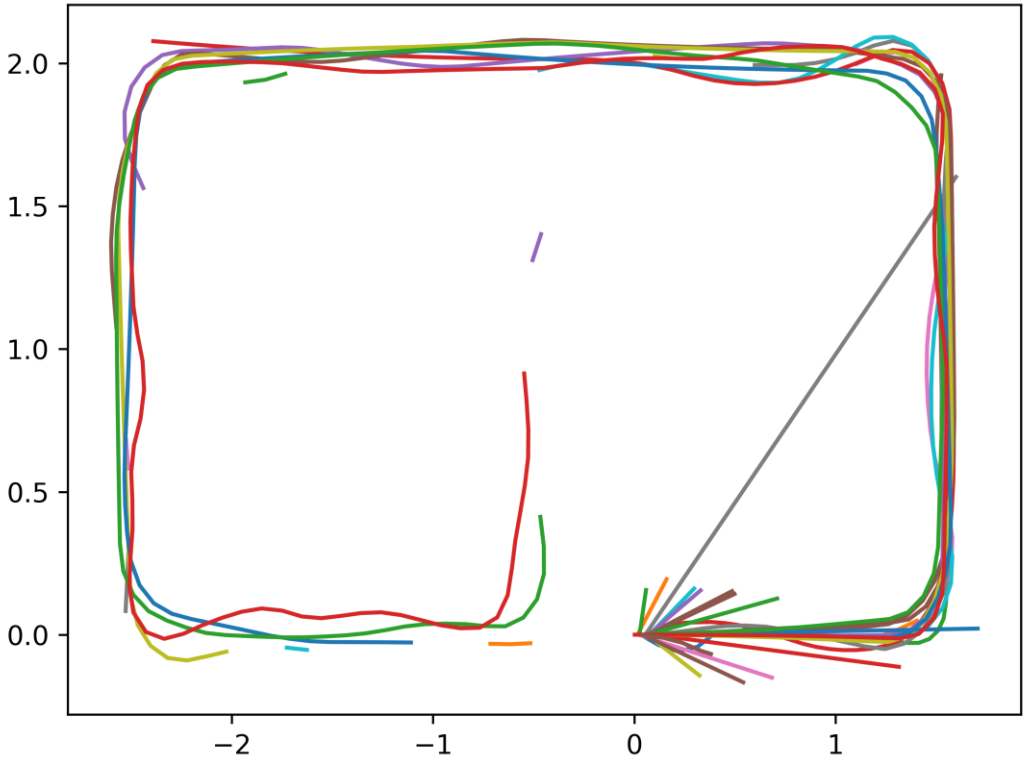
[목표점 성공 trajectory 그래프]

03. Machine learning : DQN 강화학습 이론 및 구현

- **Gazebo 상에서 구현한 DQN simulation (High Learning)**



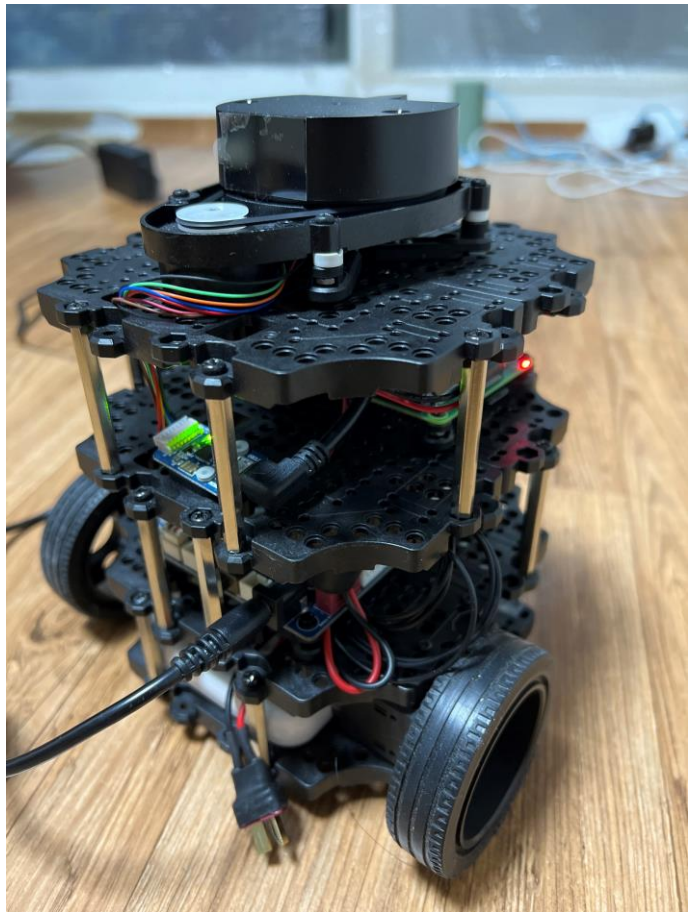
[Turtlebot3 DQN 학습 영상]



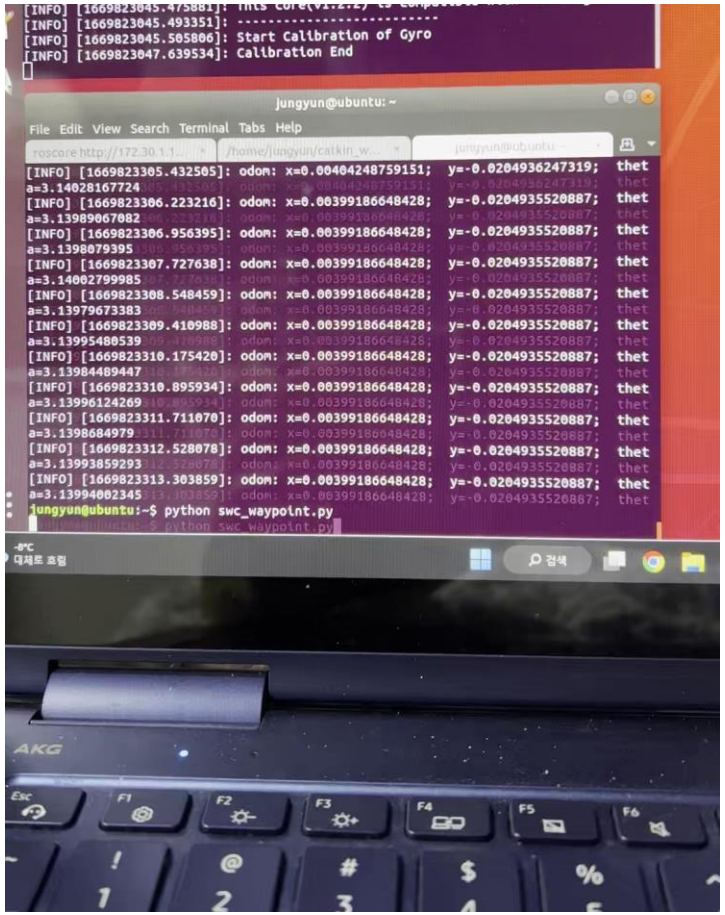
[목표점 성공 trajectory 그래프]

04. Turtlebot3 Hardware Implement

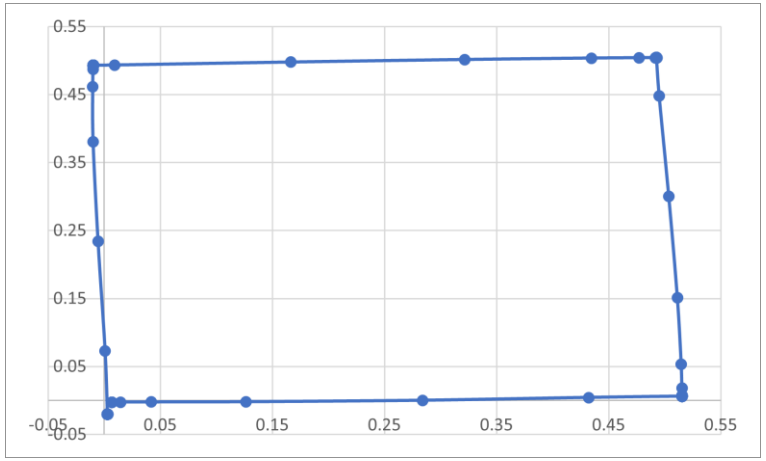
- **PC와 Turtlebot3 연결 & 시현 영상**



[PC + Turtlebot3]



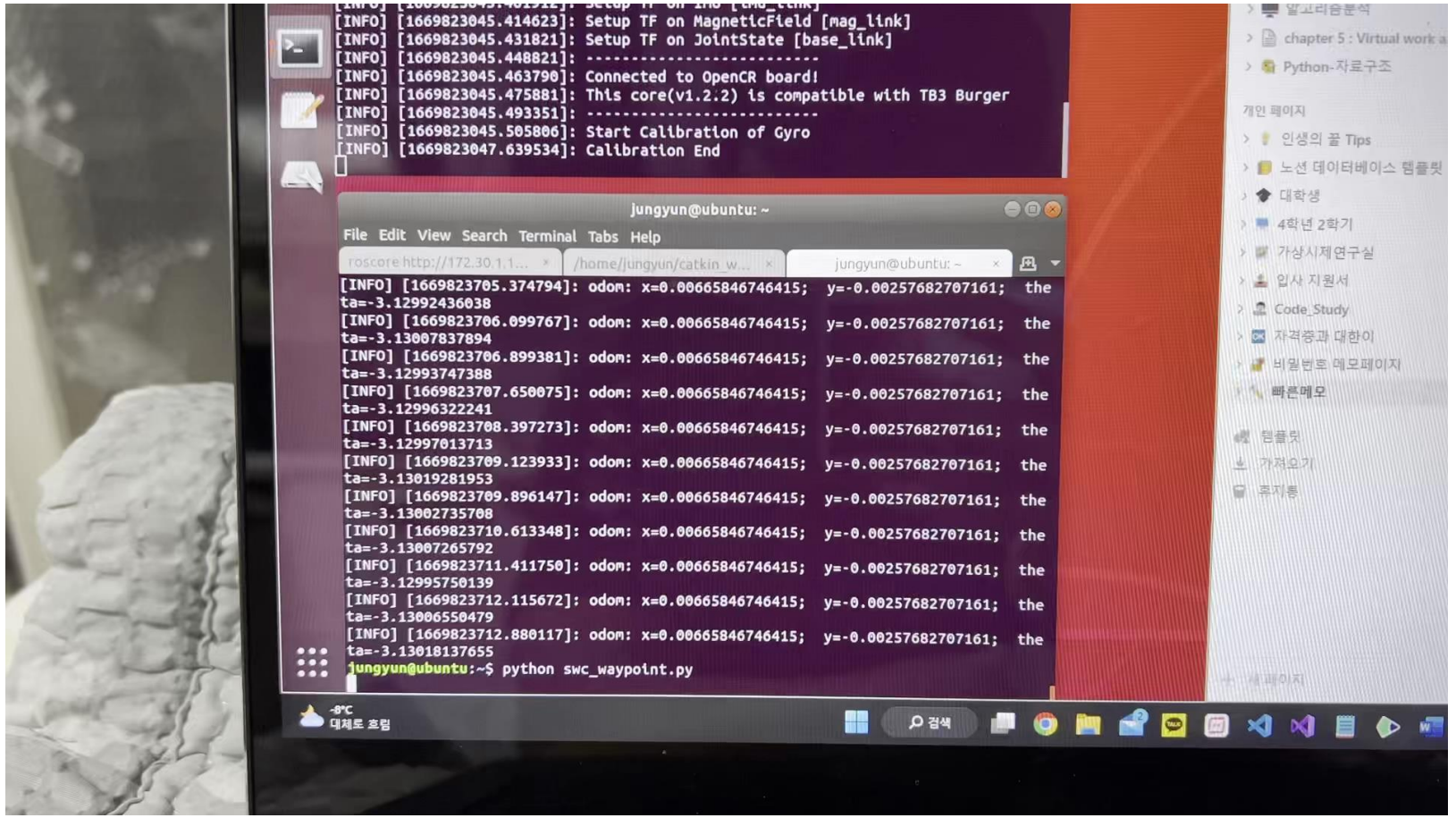
[시현 영상]



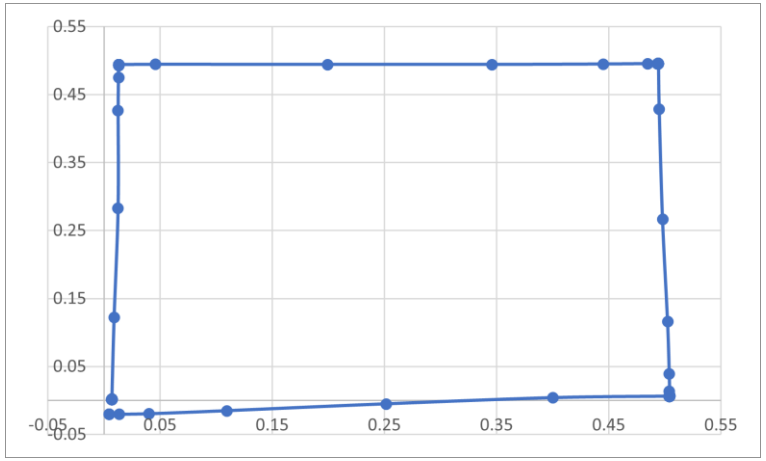
[이동경로 trajectory 그래프]

04. Turtlebot3 Hardware Implement

- 실제 환경에서 waypoint + lidar 장애물 탐지 기능 구현 영상



[실환경에서 Waypoint + Lidar 장애물 탐지 기능 구현 영상]



[이동경로 trajectory 그래프]

Implementation tool

Hardware + **Software** + **Methodology**

ubuntu

ROS

TensorFlow

Keras

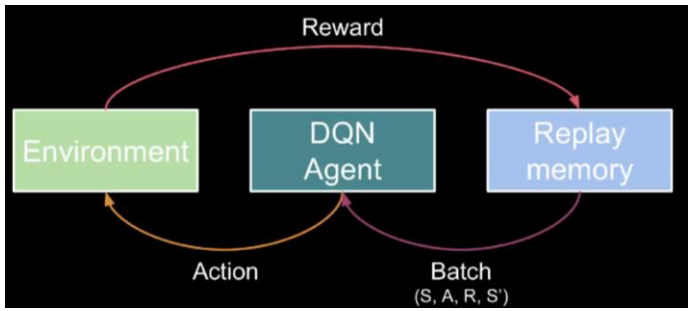
A deep learning library

TURTLEBOT3 Burger

Raspberry Pi®

OpenCR

PID Controller



Thank You For Your Kind Attention