

## PJT05. ROS with RRbot

### 1장. 개발 환경 구축

SSAFY 출신 개발자 A 씨는 로봇 기술을 개발하는 대기업에 취업에 성공하였다. 첫 출근하여 팀장님께 지시받은 업무는 리눅스 환경에서 로봇 알고리즘을 개발할 수 있는 환경을 구축하고 간단한 로보틱스 알고리즘을 구현한 후 보고였다. A씨는 임베디드 로봇 트랙에서 학습했던 내용이 떠올려 이를 구현해볼 계획을 세웠다. 먼저 장기간 지원 가능한 리눅스 버전과 로봇 알고리즘을 쉽게 개발할 수 있는 ROS를 활용하려고 검토해보니 Ubuntu 22.04와 ROS 2 humble이 적합한 것을 알 수 있었다. 그리고 로보틱스 알고리즘은 RRbot을 활용하여 2자유도 기구학 관련 정보를 구현해보기로 하고, 이런 내용을 정리하여 팀장님께 보고하여 검토를 요청드렸다. 잠시 후 팀장님께서 OK하셨으며, 여기구하 알고리즘 활용하여 다양한 형태의 경로까지 생성한 결과를 만들어 다시 보고하라고 하셨다. A씨는 위 내용을 지금부터 구현해보고자 한다.

#### Ubuntu 22.04 설치



Ubuntu 22.04는 리눅스 운영체제 중 하나이다.

기 제공된 우분투 설치 가이드 참고하여 22.04버전 설치할 수 있다.

1. Ubuntu 22.04 이미지다운로드
2. 하드디스크 분할하여 별도의 드라이브에 Ubuntu 이미지 저장
3. 재부팅 후 설치

#### ROS 2 humble 설치

아래 명령어를 터미널 창에 순서대로 입력한다.

```
locale # check for UTF-8
```

```
sudo apt update && sudo apt install locales sudo locale-gen en_US en_US.UTF-8 sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8 export LANG=en_US.UTF-8
```

```
locale # verify setting
```

```
sudo apt install software-properties-common sudo add-apt-repository universe
```

```
sudo apt update && sudo apt install curl -y sudo curl -sSL
https://raw.githubusercontent.com/ros/rosdistro/master/ros.key -o /usr/share/keyrings/ros-archive-
keyring.gpg
```

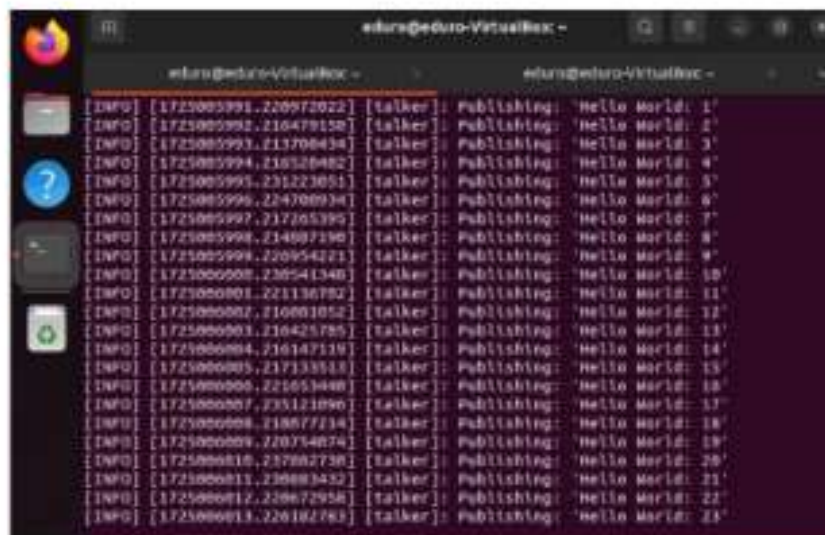
```
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/ros-archive-keyring.gpg]
http://packages.ros.org/ros2/ubuntu $(. /etc/os-release && echo $UBUNTU_CODENAME) main" | sudo
tee /etc/apt/sources.list.d/ros2.list > /dev/null
```

```
sudo apt update sudo apt upgrade
```

```
sudo apt install ros-humble-desktop sudo apt install ros-dev-tools
```

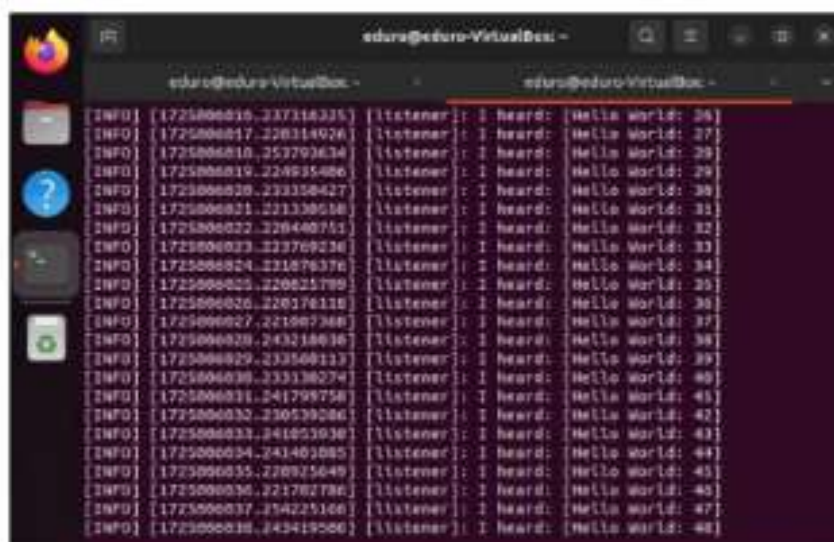
## ROS 정상 동작 여부 확인

터미널 창 2개를 열어 아래 명령어로 talker, listener 예제를 통해 topic을 이용한 데이터 송수신 결과를 확인한다.



```
eduro@eduro-VirtualBox: ~
[INFO] [1725005991.220972022] [talker]: Publishing: 'Hello World: 1'
[INFO] [1725005992.216479158] [talker]: Publishing: 'Hello World: 2'
[INFO] [1725005993.213706434] [talker]: Publishing: 'Hello World: 3'
[INFO] [1725005994.218520482] [talker]: Publishing: 'Hello World: 4'
[INFO] [1725005995.231223051] [talker]: Publishing: 'Hello World: 5'
[INFO] [1725005996.224700934] [talker]: Publishing: 'Hello World: 6'
[INFO] [1725005997.217265395] [talker]: Publishing: 'Hello World: 7'
[INFO] [1725005998.214897190] [talker]: Publishing: 'Hello World: 8'
[INFO] [1725005999.220954221] [talker]: Publishing: 'Hello World: 9'
[INFO] [1725006000.230541340] [talker]: Publishing: 'Hello World: 10'
[INFO] [1725006001.221136782] [talker]: Publishing: 'Hello World: 11'
[INFO] [1725006002.210001052] [talker]: Publishing: 'Hello World: 12'
[INFO] [1725006003.216425785] [talker]: Publishing: 'Hello World: 13'
[INFO] [1725006004.216147119] [talker]: Publishing: 'Hello World: 14'
[INFO] [1725006005.217133518] [talker]: Publishing: 'Hello World: 15'
[INFO] [1725006006.221651448] [talker]: Publishing: 'Hello World: 16'
[INFO] [1725006007.235121090] [talker]: Publishing: 'Hello World: 17'
[INFO] [1725006008.218677234] [talker]: Publishing: 'Hello World: 18'
[INFO] [1725006009.220754074] [talker]: Publishing: 'Hello World: 19'
[INFO] [1725006010.237982738] [talker]: Publishing: 'Hello World: 20'
[INFO] [1725006011.230883432] [talker]: Publishing: 'Hello World: 21'
[INFO] [1725006012.228672958] [talker]: Publishing: 'Hello World: 22'
[INFO] [1725006013.226182783] [talker]: Publishing: 'Hello World: 23'
```

talker node: publishing data

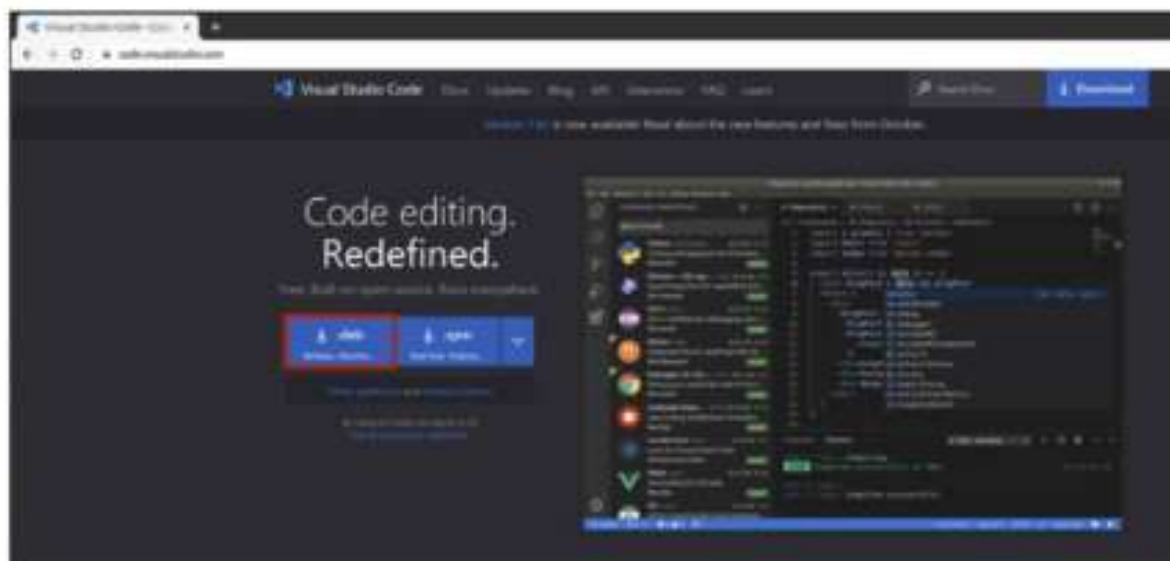


```
eduro@eduro-VirtualBox: ~
[INFO] [1725006016.237116335] [listener]: I heard: 'Hello World: 26'
[INFO] [1725006017.228114026] [listener]: I heard: 'Hello World: 27'
[INFO] [1725006018.253793634] [listener]: I heard: 'Hello World: 28'
[INFO] [1725006019.224935486] [listener]: I heard: 'Hello World: 29'
[INFO] [1725006020.233358427] [listener]: I heard: 'Hello world: 30'
[INFO] [1725006021.221330518] [listener]: I heard: 'Hello World: 31'
[INFO] [1725006022.228440751] [listener]: I heard: 'Hello World: 32'
[INFO] [1725006023.222769236] [listener]: I heard: 'Hello World: 33'
[INFO] [1725006024.231879376] [listener]: I heard: 'Hello World: 34'
[INFO] [1725006025.220825789] [listener]: I heard: 'Hello World: 35'
[INFO] [1725006026.228176118] [listener]: I heard: 'Hello World: 36'
[INFO] [1725006027.221987368] [listener]: I heard: 'Hello world: 37'
[INFO] [1725006028.243218636] [listener]: I heard: 'Hello World: 38'
[INFO] [1725006029.233508113] [listener]: I heard: 'Hello World: 39'
[INFO] [1725006030.233338274] [listener]: I heard: 'Hello World: 40'
[INFO] [1725006031.241799758] [listener]: I heard: 'Hello World: 41'
[INFO] [1725006032.230539286] [listener]: I heard: 'Hello World: 42'
[INFO] [1725006033.241805930] [listener]: I heard: 'Hello World: 43'
[INFO] [1725006034.241401085] [listener]: I heard: 'Hello World: 44'
[INFO] [1725006035.228925649] [listener]: I heard: 'Hello World: 45'
[INFO] [1725006036.221782780] [listener]: I heard: 'Hello World: 46'
[INFO] [1725006037.234225160] [listener]: I heard: 'Hello world: 47'
[INFO] [1725006038.243419586] [listener]: I heard: 'Hello world: 48'
```

listener node: subscribing data

## Visual studio code 설치

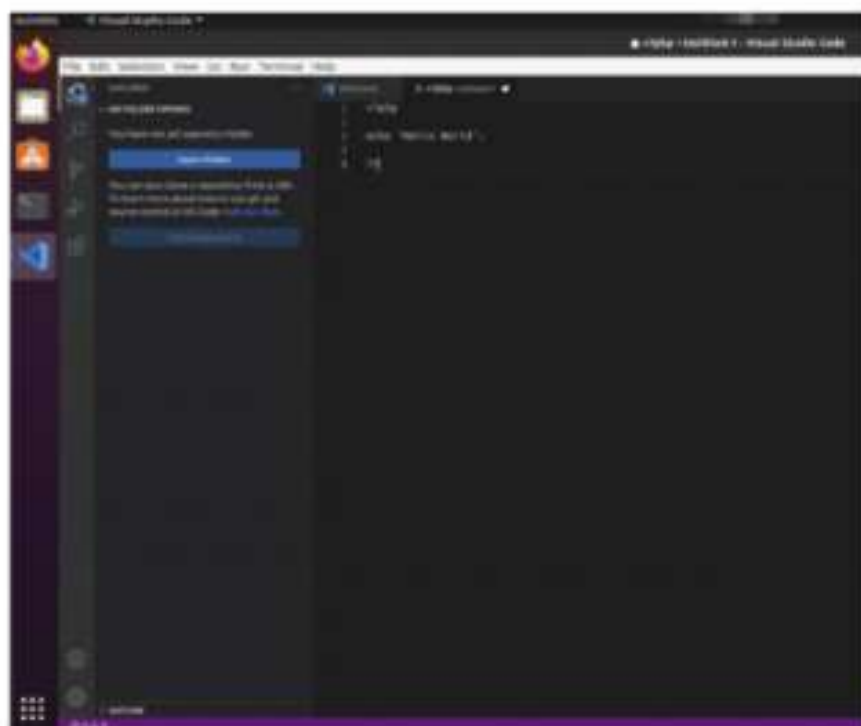
공식 홈페이지(<https://code.visualstudio.com/>)



## Visual Studio Code 다운로드 사이트

다운로드 받은 후 마우스 우클릭 후 아래 순서대로 설치한다.

1. Open With Other Application
2. Software Install 선택 후 Select
3. Install



## Visual Studio Code 실행

## 2장. RRbot 설치하기

RRbot을 실행하기 위해서는 아래 순차대로 진행하여야 합니다.

### 1. 워크 스페이스 생성

터미널

```
mkdir -p ~/ros2_ws/src
```

```
cd ~/ros2_ws/src
```

```
cd ..
```

```
colcon build
```

위 명령어를 실행하면 ros2\_ws 폴더 구조는 다음과 같다.

```
edurobot@edurobot:~/ros2_ws$ ls
build  install  log  src
```

### 2. 환경변수 설정: bashrc 설정

터미널

```
code ~/.bashrc
```

.bashrc

제일 하단에 아래 코드 2줄 추가

```
source /opt/ros/humble/setup.bash
```

```
source ~/ros2_ws/install/local_setup.bash
```

터미널

```
source ~/.bashrc
```

위 명령어를 터미널 창과 visual studio code를 활용하여 입력하고 저장한다.



```

$ bash
some@edurot:~$ . bashrc
96. # alias ls="ls -la"
97. alias alert="notify-send --urgency=low -i \"$@\" \"$@\" \"[ $(cat /dev/urandom | tr -dc 'a-z' | fold -w 64 | xargs printf '%30s' | sed 's/ /\\n/g' | grep ^$ | sed 's/^$//g' | head -n 1]\" \"$@\""
98.
99. # Alias definitions.
100. # You may want to put all your additions into a separate file like
101. # ~/.bash_aliases, instead of adding them here directly.
102. # See /usr/share/doc/bash-doc/examples in the bash-doc package.
103.
104. if [ -f ~/.bash_aliases ]; then
105. . ~/.bash_aliases
106. fi
107.
108. # enable programmable completion features (you don't need to enable
109. # this, if it's already enabled in /etc/bash.bashrc and /etc/profile
110. # sources /etc/bash.bashrc)
111. if ! shopt -o posix; then
112. if [ -f /usr/share/bash-completion/bash_completion ]; then
113. . /usr/share/bash-completion/bash_completion
114. elif [ -f /etc/bash_completion ]; then
115. . /etc/bash_completion
116. fi
117. fi
118.
119. source /opt/ros/humble/setup.bash
120. source ~/ros2_ws/install/local_setup.bash

```

### 3. ros2\_control\_demos 다운로드

#### 터미널

```

git clone https://github.com/ros-controls/ros2_control_demos -b humble
cd ~/ros2_ws/

```

위 명령어를 터미널에서 입력하여 실행하면 다음과 같은 결과를 얻을 수 있다.

```

edurot@edurot:~/ros2_ws$ git clone https://github.com/ros-controls/ros2_control_demos -b humble
Cloning into 'ros2_control_demos'...
remote: Enumerating objects: 5163, done.
remote: Counting objects: 100% (512/512), done.
remote: Compressing objects: 100% (76/76), done.
remote: Total 5163 (delta 47), reused 88 (delta 35), pack-reused 5051 (from 1)
Receiving objects: 100% (5163/5163), 3.97 MiB | 25.11 MiB/s, done.
Resolving deltas: 100% (3264/3264), done.

```

### 4. rrbot 관련 라이브러리 설치

#### 터미널

```

sudo rosdep init
rosdep update
sudo apt-get update
rosdep install --from-paths ./ -i -y --rosdistro humble

```

위 명령어를 순차적으로 입력하여 관련 라이브러리를 설치한다. (설치 시간 5분 이상)

## 5. 전체 패키지 빌드

### 터미널

```
cd ~/ros2_ws/  
  
source ~/.bashrc  
  
colcon build --merge-install
```

위 명령어를 순차적으로 입력하면 전체 패키지를 빌드하게 되며 아래 그림과 같이 별다른 에러없이 완료되어야 한다.

(만약, 에러 발생할 경우 위에서 설명한 내용 중 어떤 부분을 놓쳤거나 이전 설정에서 문제가 발생할 가능성이 높음 - 문제 해결 불가할 경우 포맷 후 재설치 추천)

```
edurobot@edurobot:~/ros2_ws$ colcon build --merge-install --parallel-workers 8  
Starting >>> ros2_control_demo_description  
Starting >>> ros2_control_demo_testing  
Finished <<< ros2_control_demo_description [0.57s]  
Finished <<< ros2_control_demo_testing [0.57s]  
Starting >>> ros2_control_demo_example_1  
Starting >>> ros2_control_demo_example_5  
Starting >>> ros2_control_demo_example_10  
Starting >>> ros2_control_demo_example_11  
Starting >>> ros2_control_demo_example_12  
Starting >>> ros2_control_demo_example_14  
Starting >>> ros2_control_demo_example_2  
Starting >>> ros2_control_demo_example_3  
Finished <<< ros2_control_demo_example_14 [4.39s]  
Starting >>> ros2_control_demo_example_4  
Finished <<< ros2_control_demo_example_2 [4.45s]  
Starting >>> ros2_control_demo_example_6  
Finished <<< ros2_control_demo_example_11 [5.20s]  
Starting >>> ros2_control_demo_example_7  
Finished <<< ros2_control_demo_example_3 [5.27s]  
Starting >>> ros2_control_demo_example_8  
Finished <<< ros2_control_demo_example_5 [5.38s]  
Starting >>> ros2_control_demo_example_9  
Finished <<< ros2_control_demo_example_1 [5.91s]  
Starting >>> ros2_control_demo_example_15  
Finished <<< ros2_control_demo_example_15 [1.00s]  
Finished <<< ros2_control_demo_example_10 [8.21s]  
Finished <<< ros2_control_demo_example_8 [3.18s]  
Finished <<< ros2_control_demo_example_4 [4.39s]  
Finished <<< ros2_control_demo_example_6 [4.35s]  
Finished <<< ros2_control_demo_example_9 [4.87s]  
Finished <<< ros2_control_demo_example_12 [10.7s]  
Finished <<< ros2_control_demo_example_7 [8.00s]  
Starting >>> ros2_control_demos  
Finished <<< ros2_control_demos [0.76s]  
  
Summary: 17 packages finished [14.7s]
```

## 6. rrbot 실행

### 터미널

```
ros2 launch ros2_control_demo_example_1 view_robot.launch.py
```

위 명령어를 입력하면 아래 그림과 같이 RViz가 실행되고 RRbot을 확인할 수 있다.

좌측에 있는 Joint\_state\_controller\_gui를 이용해서 rrbot의 움직여볼 수 있다.

