Spring, 2021

지능로봇

SLAM
(SIMULTANEOUS LOCALIZATION AND MAPPING)

메카트로닉스공학과

한국산업기술대학교

PRACTICE

- SYSTEM INTERFACE
- SLAM
- SAVE MAP
- MAP

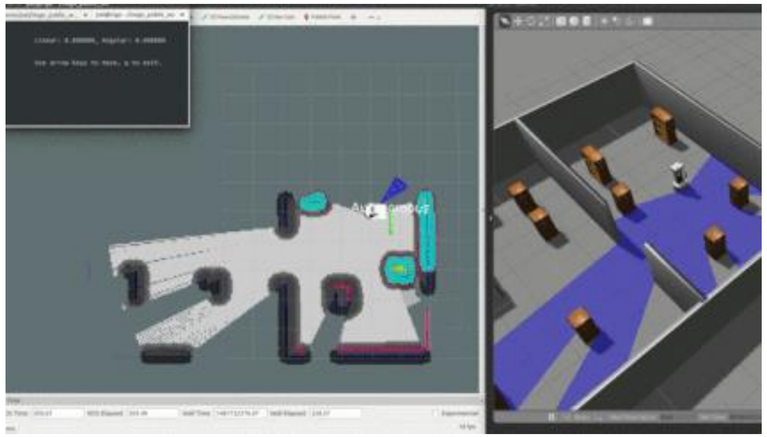
SLAM

- WHAT IS SLAM
- GMAPPING
- GRAPH OF SLAM
- TF

01. SLAM (SIMULTANEOUS LOCALIZATION AND MAPPING)

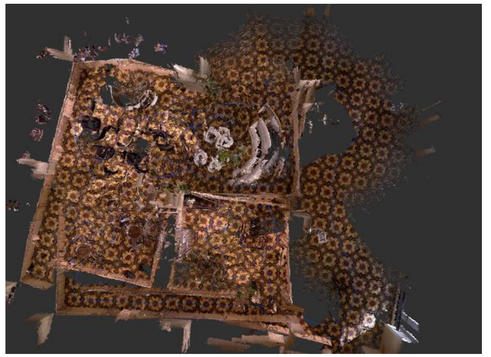
- WHAT IS SLAM

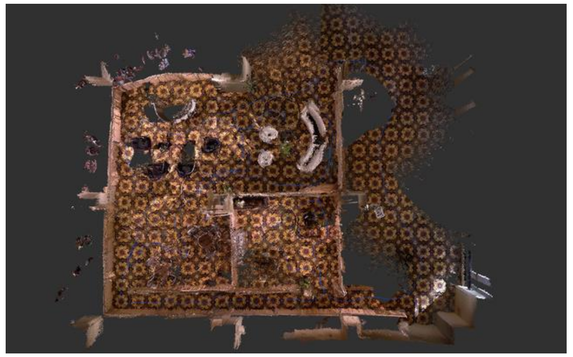
SLAM는 **동시적 위치추정 및 지도작성**으로, 임의 공간에서 이동하면서 주변을 탐색할 수 있는 로봇에 대해, 그 공간의 **지도 및 현재 위치**를 추정하는 문제이다.



- WHY SLAM?

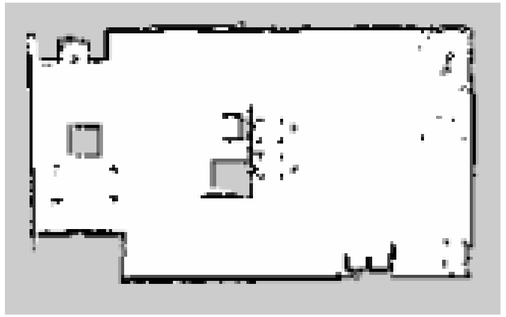
로봇의 자율주행을 위해서는 로봇의 **현재 위치를 추정**하는 것이 중요. 단순 위치추정 만으로는 센서의 오차로 인해 **정확도가 떨어짐**. 이를 해결하기 위해 시간 단위로 **맵을 그리며 위치추정 및 위치보정**을 실행함.

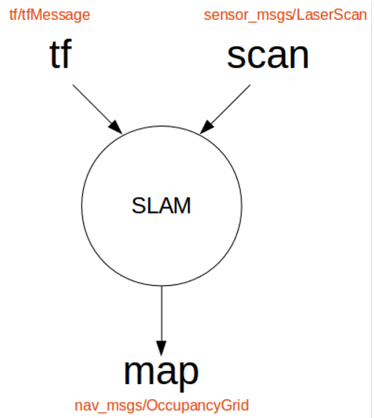




01. SLAM (SIMULTANEOUS LOCALIZATION AND MAPPING)

- GMAPPING (GRID-BASED SLAM WITH PARTICLE FILTER)





MAP

- **SLAM**의 최종목표는 **지도를 만드는 것**
- **점유 격자 지도**
  (1) 흰색 : 로봇이 이동 가능한 **자유 영역**
  (2) 검은색 : 로봇이 이동 불가능한 **점유 영역**
  (3) 회색 : 확인되지 않은 **미지 영역**

INFORMATION

- 지도를 작성하기 위해선 '**거리 값**'과 '**위치 값**'이 필요함.
- '**거리 값**'은 LRF, Depth camera 등을 이용해 X-Y 평면을 스캔한 값.
- '**위치 값**'은 센서의 위치 값, 로봇과 같이 움직이므로 로봇의 이동량인 오도메트리에 의존.
- ROS에서는 '**거리 값**'을 '**scan**', '**위치 값**'을 '**tf**'(transformation)이라 부르게 됨.

01. SLAM (SIMULTANEOUS LOCALIZATION AND MAPPING)

- GMAPPING (GRID-BASED SLAM WITH PARTICLE FILTER)

01

- **초기화(initialization)**
  - 전역 위치 추정(Global localization)면에서 처음에는 로봇 위치 및 방향을 알 수 없기 때문에 **N개의 입자 (particle_i = pose(x_i,y_i,t_i)) 를 임의로 뿌림**. 이는 가장 처음에만 수행하는 것으로 입자의 가중치는 모두 같으며(1/N) 그 합은 1이 됨.

02

- **예측(prediction)**
  - 로봇의 움직임을 기술한 시스템 모델(system model)에 기반하여 **로봇의 이동 량에 잡음(noise)을 포함하여 각 입자들을 이동시킴**.

03

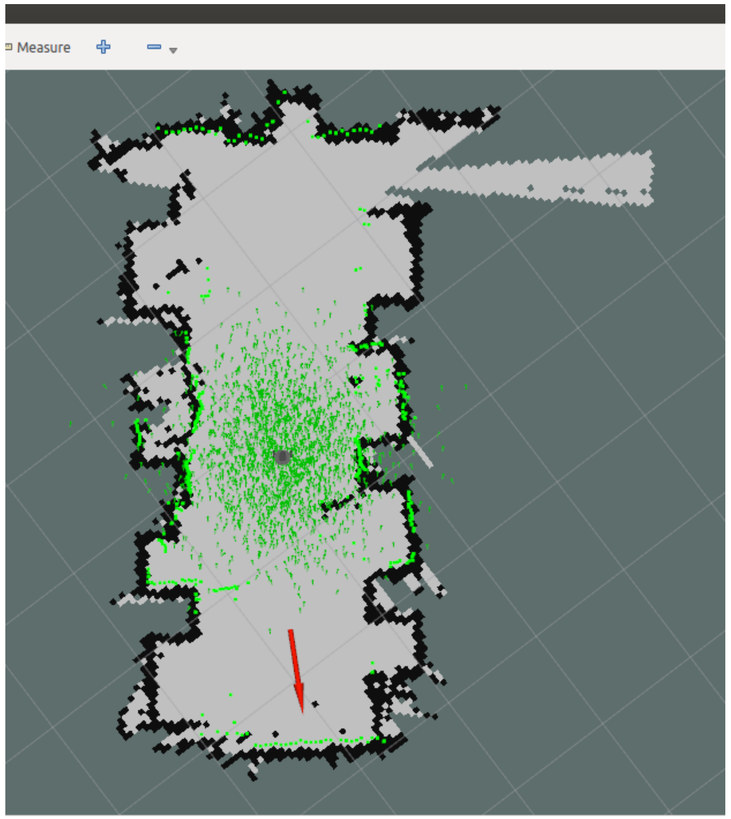
- **보정(update)**
  - 계측된 센서 정보들을 기반으로 **각 입자가 존재할 확률을 계산**하고, 이를 반영하여 각 입자의 가중치가 1이 되도록 **가중치의 값을 갱신**. 이 갱신후의 입자 값은 초기화에서 주어졌던 particle_i = pose(x_i,y_i,t_i), weight_i (i=1,...,N) 이 예측과 갱신을 거쳐 새로운 상태가 됨.

04

- **위치 추정(pose estimation)**
  - N개의 모든 각 입자의 위치 (x,y,t) 와 가중치 (weight)를 곱하여 **로봇의 추정 위치를 계산**.

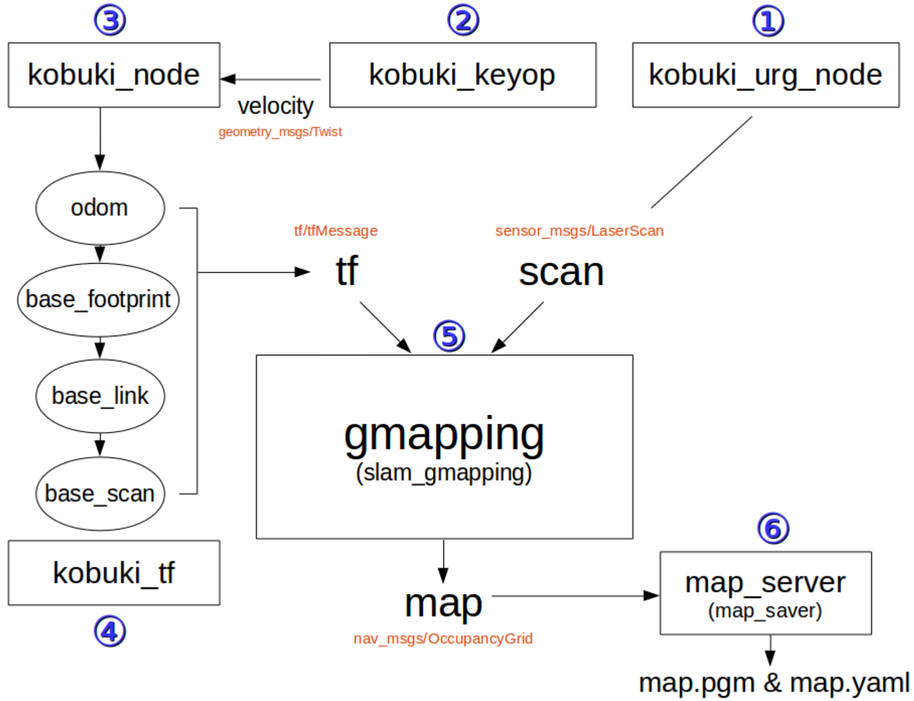
05

- **재추출(resempling)**
  - 새로운 입자를 생성하는 단계로 **가중치가 작은 입자를 없애고** 가중치가 높은 입자를 중심으로 기존의 입자의 특성인 입자의 위치정보를 물려받은 **새로운 입자를 추가로 생성**. 여기서 입자 수 N은 그대로 유지해야 함.



01. SLAM (SIMULTANEOUS LOCALIZATION AND MAPPING)

- GRAPH OF SLAM (TURTLEBOT)



01
- **kobuki_urg_node**
  - LRF 센서를 실행하여 SLAM에 필요한 **scan 정보**를 slam_gmapping 노드에게 보냄.

02
- **kobuki_keyop**
  - keyboard값을 받아서 로봇을 조종. **Velocity 정보**를 Kobuki_node로 보냄.

03
- **kobuki_node**
  - 유저의 명령을 받고 이동을 하게 됨.
  이동 후에 자신의 위치를 계측/측정한 위치 정보인 **odom 정보**를 전송함.

04
- **kobuki_tf**
  - 센서의 위치인 base_scan을 SLAM에게 넘기기 위하여 **odom** -> **base_footprint** -> **base_link** -> **base_scan** 의 변환 후, **tf** 형태로 내보냄

05
- **slam_gmapping**
  - '**거리 값**'인 '**scan**' 정보와 '**위치 값**'인 '**tf**' 정보를 기반으로 지도를 작성.

06
- **map_server**
  - **map 정보**를 저장 가능한 map.pgm 파일과 이에 대한 정보 파일인 map.yaml 파일로 생성.

01. SLAM (SIMULTANEOUS LOCALIZATION AND MAPPING)

- TF (TRANSFORMATION OF TURTLEBOT)

01

- **odom**
  - **자신의 위치**를 계측/측정한 위치 정보.

02

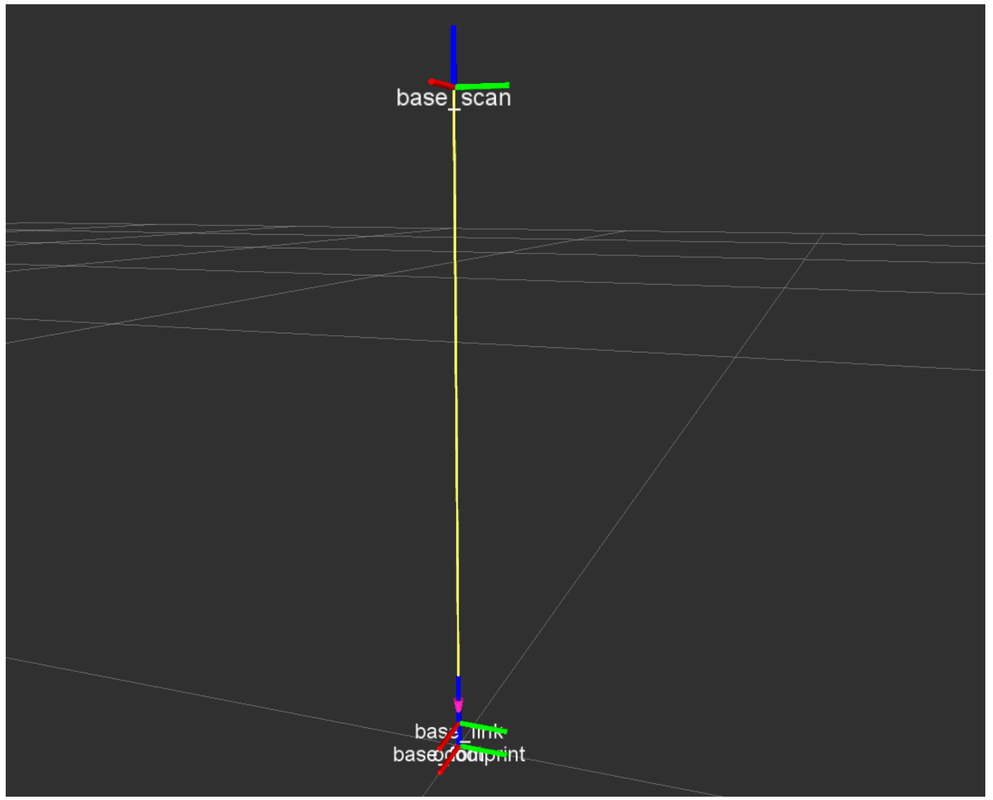
- **base_foorprint**
  - 바닥으로 부터 **로봇 밑바닥** 까지의 거리에 대한 odom로부터 변환된 좌표

03

- **base_link**
  - 바닥으로부터 **로봇의 하드웨어적 중심** 까지의 거리에 대한 base_footprint로부터 변환된 좌표

04

- **base_scan**
  - 바닥으로부터 **로봇의 센서** 까지의 거리에 대한 base_link로부터 변환된 좌표



02. PRACTICE

- SYSTEM INTERFACE (TURTLEBOT)

- Remote PC와 Robot PC가 같은 WIFI 환경을 갖도록 하여 **서로 통신이 가능하도록** 함

WIFI 공유기

LRF sensor

- Laser Range Finder
- **거리센서 역할**

SERIAL

Robot PC (raspbetty pi)

- LRF, Encoder, 3축 자이로 등의 **센서 값을 받거나 모터를 동작** 시킴으로써 **SLAM 또는 Navigation을 진행**

WIFI

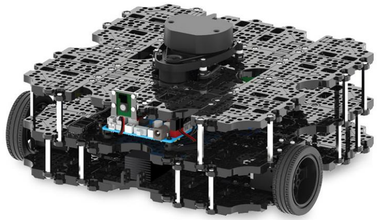
WIFI

- **ROS MASTER** 역할
- keyop 노드를 이용해 로봇을 **원격조종**
- **Rviz**를 이용해 Robot PC에서 받은 정보들을 **Display**

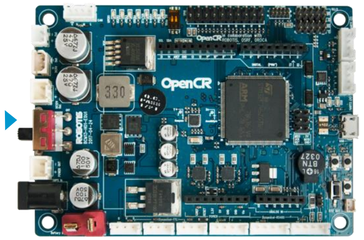
Remote PC

SERIAL

Turtlebot3



Embedded Board



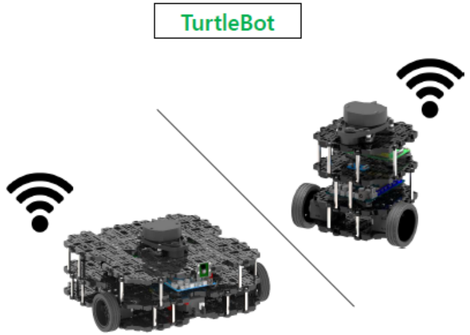
- Robot PC에 **센서 값들을 전달** (3축 자이로, 바퀴 엔코더 등)
- Robot PC로 부터 명령을 받아 **모터를 구동**

02. PRACTICE

- Turtulebot3 개발환경 (네트워크)

- **Laptop 네트워크 설정**

1. Turtlebot3와 같은 wifi로 설정
2. Terminal에 ‘ifconfig’를 입력하여 ip address를 획득
3. Terminal에 ‘cd ~’를 입력하여 home으로 이동
4. Terminal에 ‘sudo nano .bashrc’를 입력
   (.bashrc : Terminal이 켜질 때 작동될 내용들)
5. 맨 밑으로 이동하여 다음과 같이 입력 후 저장 (‘ctrl’+’x’)
   ROS_MASTER_URI=http://IP_OF_REMOTE_PC:11311
   ROS_HOSTNAME = IP_OF_REMOTE_PC
6. source ~/.bashrc (.bashrc에 갱신된 내용을 실행시킴)



ROS_MASTER_URI = http://IP_OF_REMOTE_PC:11311
ROS_HOSTNAME = IP_OF_TURTLEBOT

ROS_MASTER_URI = http://IP_OF_REMOTE_PC:11311
ROS_HOSTNAME = IP_OF_REMOTE_PC

```
# labtop setting
export ROS_MASTER_URI=http://192.168.0.101:11311
export ROS_HOSTNAME=192.168.0.101
```

02. PRACTICE

- Turtulebot3 개발환경 (네트워크)

- **Turtlebot3 네트워크 설정**

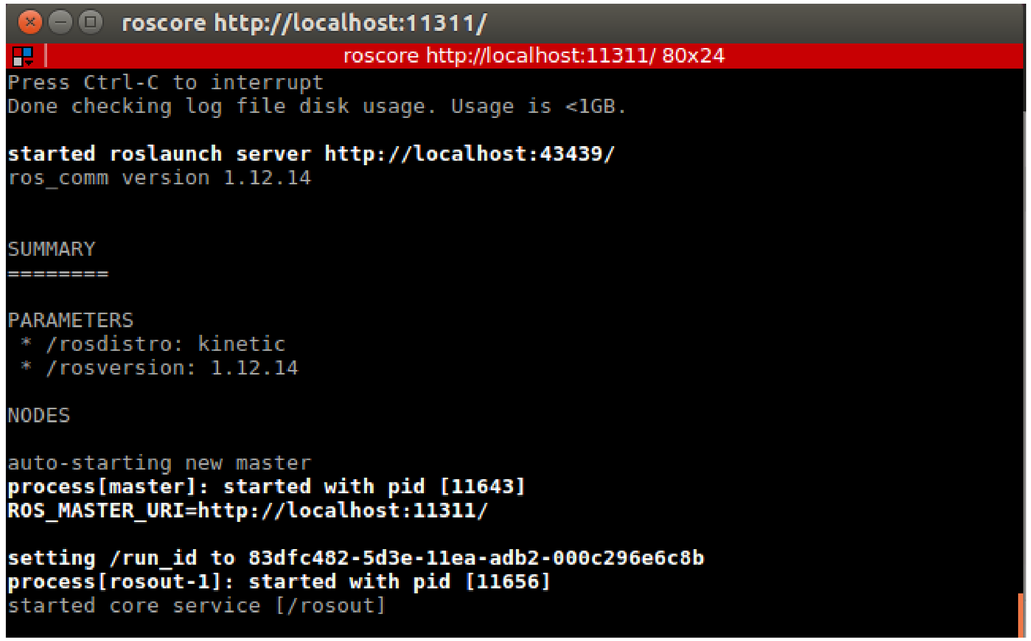
1. Laptop에서 ssh pi@IP_OF_TURTLEBOT 입력
   (Turtlebot3의 라즈베리파이에 원격접속)
2. Password 입력 : turtlebot
3. Terminal에 ‘ifconfig’를 입력하여 ip address를 획득
4. Terminal에 ‘cd ~’를 입력하여 home으로 이동
5. Terminal에 ‘sudo nano .bashrc’를 입력
   (.bashrc : Terminal이 켜질 때 작동될 내용들)
6. 맨 밑으로 이동하여 다음과 같이 입력 후 저장
   ROS_MASTER_URI=http://IP_OF_REMOTE_PC:11311
   ROS_HOSTNAME = IP_OF_TURTLEBOT
7. source ~/.bashrc (.bashrc에 갱신된 내용을 실행시킴)

```
#turtlebot3 setting
export ROS_MASTER_URI=http://192.168.0.101:11311
export ROS_HOSTNAME=192.168.0.115
```

02. PRACTICE

- SLAM [학생]

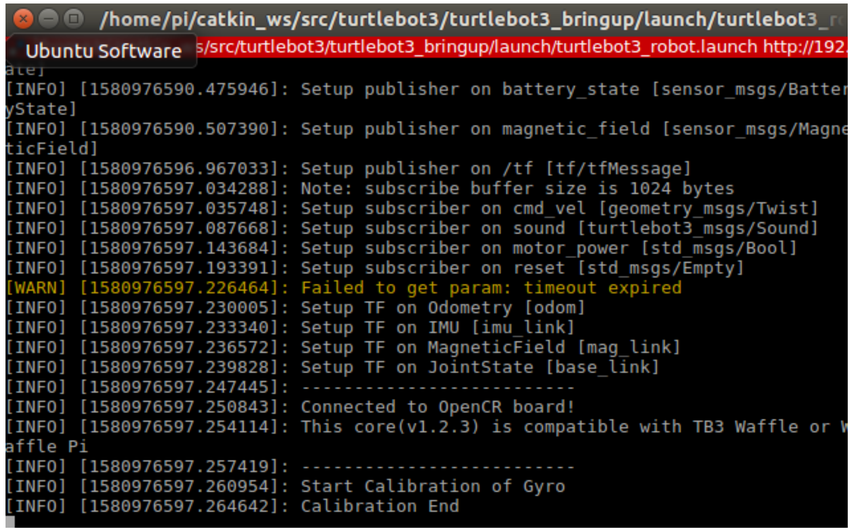
- [Remote PC] ROS Master를 실행
  $ roscore



02. PRACTICE

- SLAM [학생]

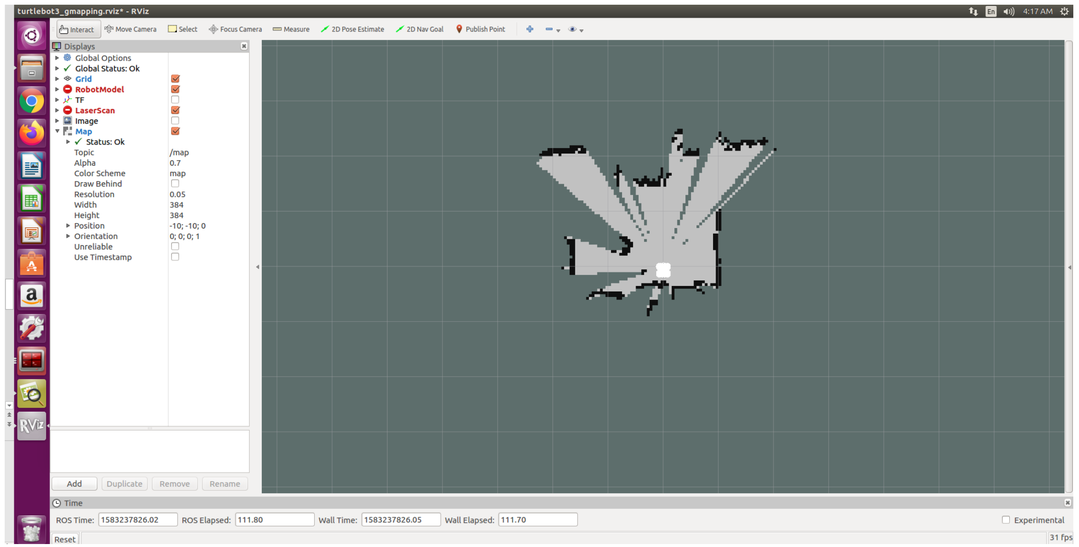
- [TurtleBot] 터틀봇에 원격으로 접속하여 운동을 위한 노드를 실행시킴
  $ ssh pi@IP_OF_TURTLEBOT
  (password : turtlebot)
  $ roslaunch turtlebot3_bringup turtlebot3_robot.launch --screen



02. PRACTICE

- SLAM [학생]

- [Remote PC] 로봇모델을 설정하고 gmapping slam을 실행
  $ export TURTLEBOT3_MODEL=waffle_pi
  $ roslaunch turtlebot3_slam turtlebot3_slam.launch slam_method:=gmapping



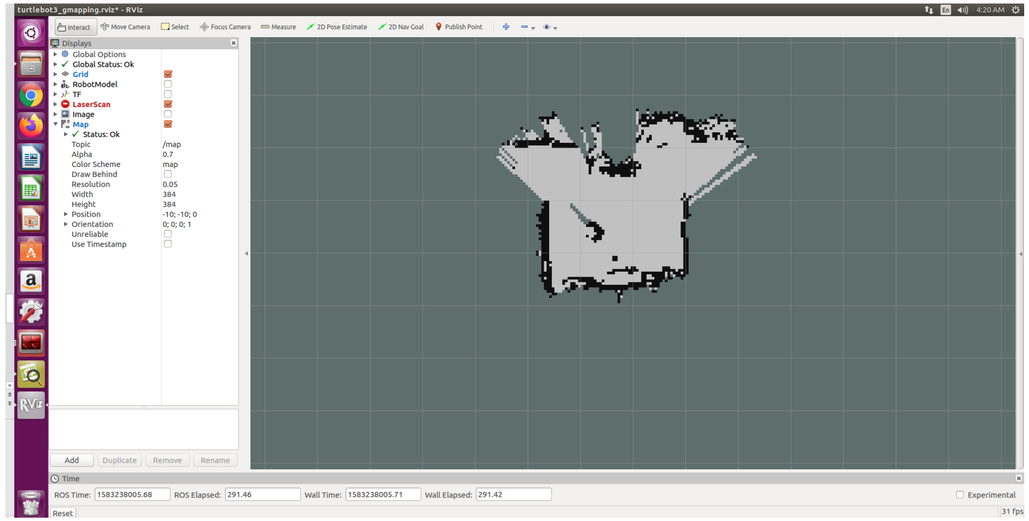
02. PRACTICE

- SLAM [학생]

- [Remote PC] teleop_key 노드를 이용해 키보드로 터틀봇을 조작하며 지도를 작성

  $ export TURTLEBOT3_MODEL=waffle_pi

  $ roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch --screen



02. PRACTICE

- Save Map [학생]

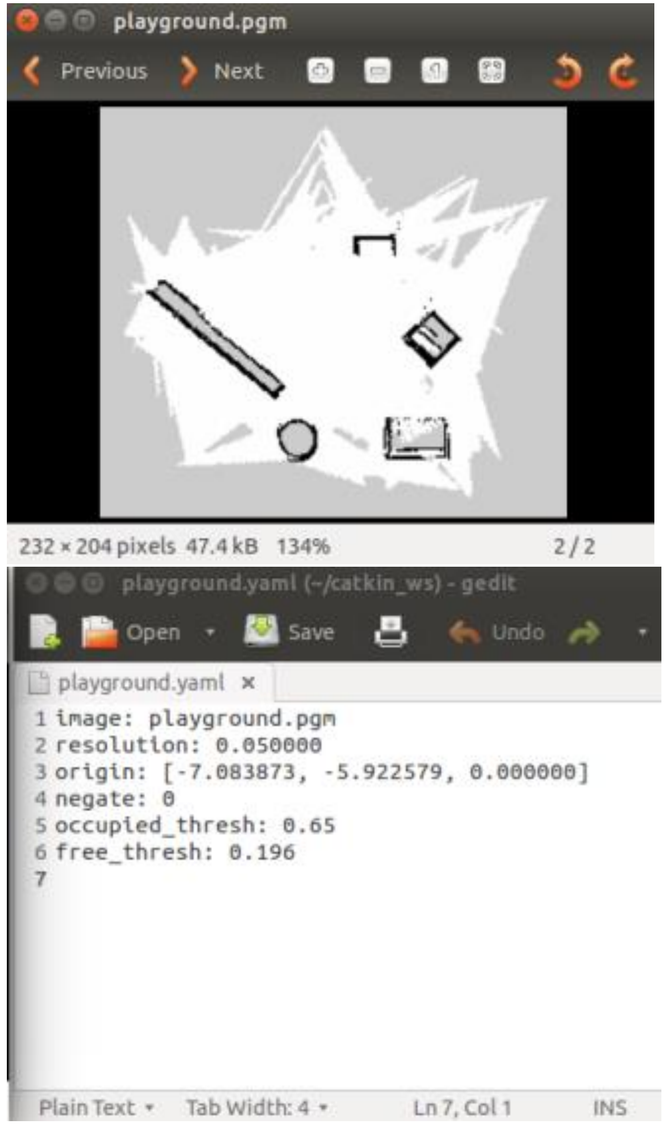
- [Remote PC] Save Map to Home : home folder에 map이라는 파일 명으로 저장
  $ rosrun map_server map_saver –f ~/map

- Map

- Map을 저장하면 2가지 파일이 저장됨.

1. Occupancy Grid Map 형태의 .pgm 파일
2. map에 대한 정보를 지니는 .yaml 파일

- 차후에 Navigation 실행 시 .yaml 파일을 호출하여 Navigation에 이용함.



02. PRACTICE

- SLAM [학생]

- 모든 작업이 끝나면 ‘ctrl’ + ‘c’ 를 눌러 모든 노드를 종료시킨다.

THANK YOU

SOURCES

[1] 로봇 운영체제 ROS 강좌, Pyo yunseok , 2015.3.2,
https://cafe.naver.com/openrt/2360

[2] SLAM, jacobyu, 2018, https://steemit.com/kr-dev/@jacobyu/54qama