**학부생 연구 참여 결과 보고서**

**-Turtlebot3를 이용한 SLAM과 Navigation 기술 구현-**

**2020.07.31**

**김근호, 문병필**

**목차**

1. **제목**
2. **서론**
3. **용어 정리**
4. **연구 방법**
5. **연구 결과**
6. **고찰**
7. **참고 문헌**

**제목**

* Turtlebot3 를 이용한 SLAM구현과 자율주행기술 구현

**서론**

* Turtlebot3는 ROS표준 플랫폼 로봇이다. 여기서 ROS란 Robot Operation System의 약자로, 다양한 기종(하드웨어)에 대해서도 동일한 OS를 적용하여 응용 소프트웨어를 개발할 수 있게 해주는 로봇 소프트웨어 프로그램이다. 다양한 기종에 대해 각기 다른 소프트웨어를 개발하는 것이 아니라 몇 개의 플랫폼을 개발함으로서 2차적인 응용 소프트웨어 개발에 치중할 수 있는 것이다.
* ROS는 공식 운영 체제로 Ubuntu(우분투)를 지원한다. PC에 우분투를 설치한 후 ROS를 설치하면 Setup, Bring up등의 과정을 거쳐 로봇의 수동 조작 SLAM, 그리고 길 찾기 기능까지 이용할 수 있다.
* 기존 OS(Ubuntu)위에 ROS라는 새로운 OS를 설치하는 것에서 의문이 생길 수 있다. ROS는 우리가 알던 전통적인 OS(운영체제)가 아니라 메타 운영체제라 한다. 이는 전통적인 운영체제(Ubuntu)와 로봇 하드웨어, 센서 그리고 사용자의 로봇 응용 프로그램 모두를 동시에 관리하는 종합 지원 시스템이라는 것을 의미한다.

**용어 정리**

* ROS: 로봇 응용프로그램 개발을 위한 로봇의 운영체제와 비슷한 역할을 하며, 개발 및 디버깅 도구를 제공한다.
* NODE: ROS에서 최소 단위 실행 프로세서이다. 하나의 목적에 하나의 NODE가 대응되는 실행 가능한 프로그램으로 간주해도 된다. 노드는 생성과 함께 마스터에 노드 이름, 발행자 이름, 구독자 이름, 토픽 이름, 서비스 이름, 메시지 형태, URI주소, 포트를 등록한다. 이 정보들을 기반으로 노드끼리 토픽과 서비스를 이용하여 메시지를 주고받을 수 있다.
* TOPIC & MESSAGE: 토픽은 주제이다. 발행자 노드가 어떤 주제에 대해 마스터에 토픽을 등록하고, 토픽에 대한 내용을 메시지의 형태로 발행한다.

**연구 방법**

* Ubuntu 설치
* BIOS로 들어가 부팅 우선순위를 USB로 설정하여 우분투를 설치한다.
* 영어 버전으로 설치한다. 이후에 한글 설정을 해도 된다. 영어 버전으로 설치해야 기본 설정들을 이해하고 조작하기 편하다.
* Setup
* PC Setup
  + - ROS download
    - Dependent ROS download
    - Network Configuration
* SBC Setup: SBC란 (Single Board Computer)의 약자로, 초소형 크기와 저전력의 완전한 컴퓨터이다. 대표적으로 라즈베리 파이 기판이 있다.
  + - Raspberry Pi3에 리눅스(Ubuntu)와 ROS를 설치하는 방법에 [Install Linux (Ubuntu MATE)](https://emanual.robotis.com/docs/en/platform/turtlebot3/raspberry_pi_3_setup/#install-linux-ubuntu-mate), [Install Linux (Raspbian)](https://emanual.robotis.com/docs/en/platform/turtlebot3/raspberry_pi_3_setup/#install-linux-based-on-raspbian), [Install Linux (webOS Robotics Platform)](https://github.com/ros/meta-ros/wiki/OpenEmbedded-Build-Instructions) 세 가지가 있다. 그 중에서 Raspbian을 사용하였다.
* Bring up
* PC에서 roscore을 실행한다. ROS의 모든 노드는 roscore을 구동한 후에 사용할 수 있다.
* SBC에서 turtlebot3를 실행시키기 위한 기본 패키지를 가져온다.
* Basic Operation
* Topic monitor: 터틀봇3에서 실행되는 토픽을 확인하기 위해 ROS에서 제공되는 rqt를 실행한다. Rqt를 실행하면 토픽 리스트가 나오고, 토픽의 상태를 쉽게 확인 가능하다. 배터리, 토크, 센서 정보,
* Teleoperation: 다양한 방식으로 turtlebot3를 조종할 수 있다.

아래는 RViz로 실행하는 것으로,

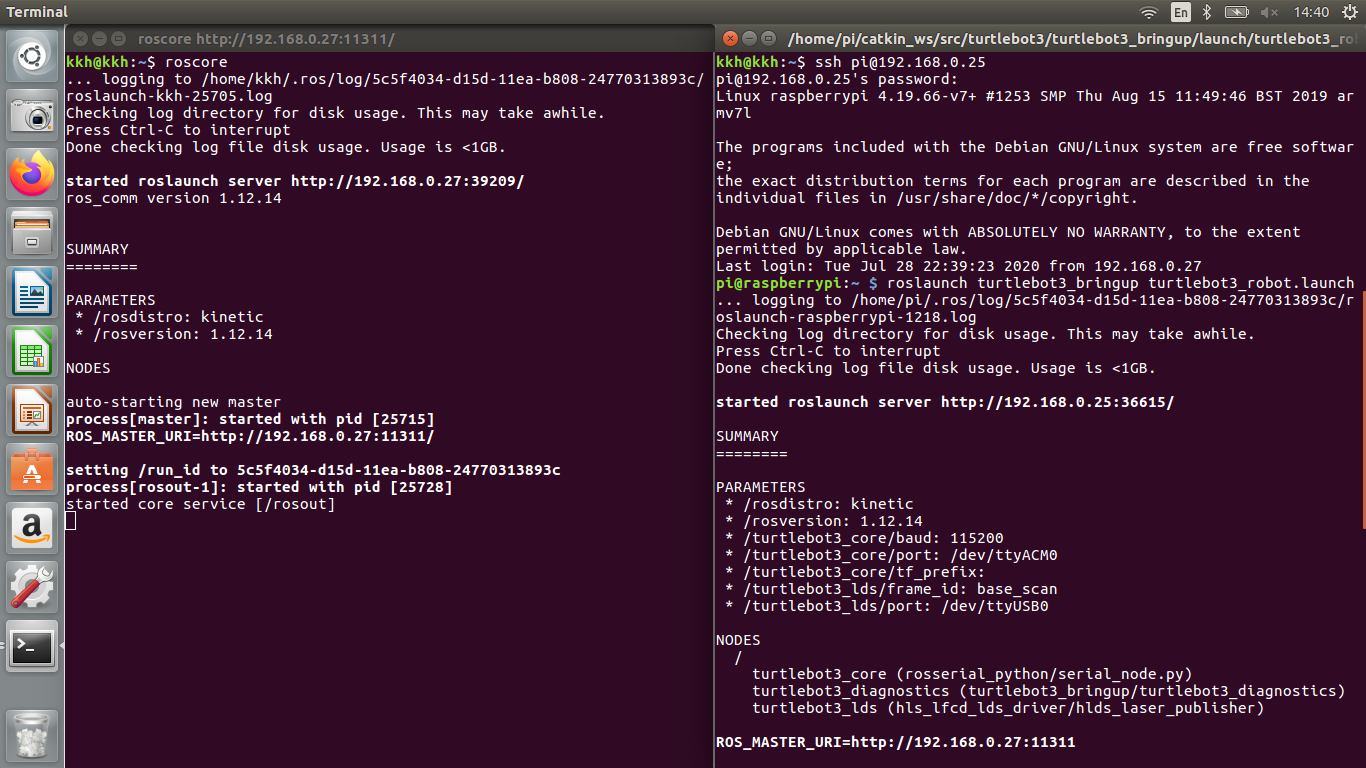
* Obstacle Detection: LDS를 이용하여 turtlebot3의 움직임으로 제어할 수 있다. 장애물이 가까우면 멈추는 기능의 패키지의 노드를 복수 실행한다(roslaunch).
* Point Operation: roslaunch를 사용하여 특정 좌표로 이동하는 기능을 RViz에서 실행한다.
* Patrol: 동일하게 RViz에서 roslaunch를 사용하여 사용자가 지정한 순찰 구역을 계속해서 순찰하도록 한다.
* SLAM
* SLAM이란 (Simultaneous Localization and Mapping)의 약자이다. 즉, 동시 현지화 및 매핑이다. 이 SLAM은 자율주행에서 핵심 기술이 되기도 하는데, 어떤 기계가 주변의 환경을 인식하여 그 공간의 지도를 작성함과 동시에 기계 자신의 위치를 파악하는 것이다.
* Roscore을 remote PC에서 실행시킨다. 이는 ros 마스터를 실행시키는 명령어이다. Remote PC와 turtlebot3가 동일 네트워크상에 연결되어 있으므로, 한 군데에서만 실행하여도 된다.
* Navigation
* 내비게이션의 정의는 익히 잘 알고 있다. 로봇에게 내비게이션이란, 로봇이 정해진 목적지까지 이동하는 것이다. 하지만, 이를 위해서는 ① 주어진 환경의 지도 ② 로봇 자신이 스스로 어디에 있는지 ③장애물을 파악하고, 피하는 것 ④다양한 경로 중 최적화된 경로 계산들이 필요하다.

1. 지도: 평소 볼 수 있는 내비게이션들은 기본적으로 지도가 탑재되어 있지만, turtlebot3와 같은 모바일 로봇들은 실내에 대한 지도가 탑재 되어있지 않다. 그렇기에, 로봇이 스스로 지도를 작성하는 기능이 필요한데, 이것이 위에 설명한 SLAM기술이다. 우리 말로는 동시적 위치 추정 및 지도 작성이다. 로봇이 공간을 이동하면서(인간의 도움을 받아) 센서(초음파, 레이저 등)들을 사용하여 주변 공간의 모습을 추정하는 것이다.
2. 로봇 자신이 어디에 있는지: 평소 보는 자동차의 내비게이션들은 GPS를 사용하지만, 이런 모바일 로봇들은 GPS를 사용하기엔 너무 정밀한 위치 파악이 필요하다. 그래서 추측 항법을 사용하여 로봇이 스스로의 위치를 특정하게 된다. 추측 항법이란, 로봇의 모터 회전량을 통해 로봇이 어디로 얼마나 이동했는지 파악하는 기술이다. 문제는 오차가 많이 발생한다는 것인데, 이는 IMU센서 등으로 관성정보를 취득하여 위치보상을 통해 오차를 줄여준다.
3. 장애물을 파악하고 피하기 위해서는 센서가 필요하다. 사용되는 센서의 종류에는 LRF(Lasor Range Finder), 초음파센서, 적외선 거리센서 등이 사용되었는데, turtlebot3에서는 LRF가 사용되었다.
4. 다양한 경로 중 최적화된 경로를 계산하는 것을 내비게이션(Navigation)이라고 한다. 이 기술에는 A\* 알고리즘, 전자자기장 알고리즘, 파티클, 그래프 방식 등 여러 방식들이 있다.

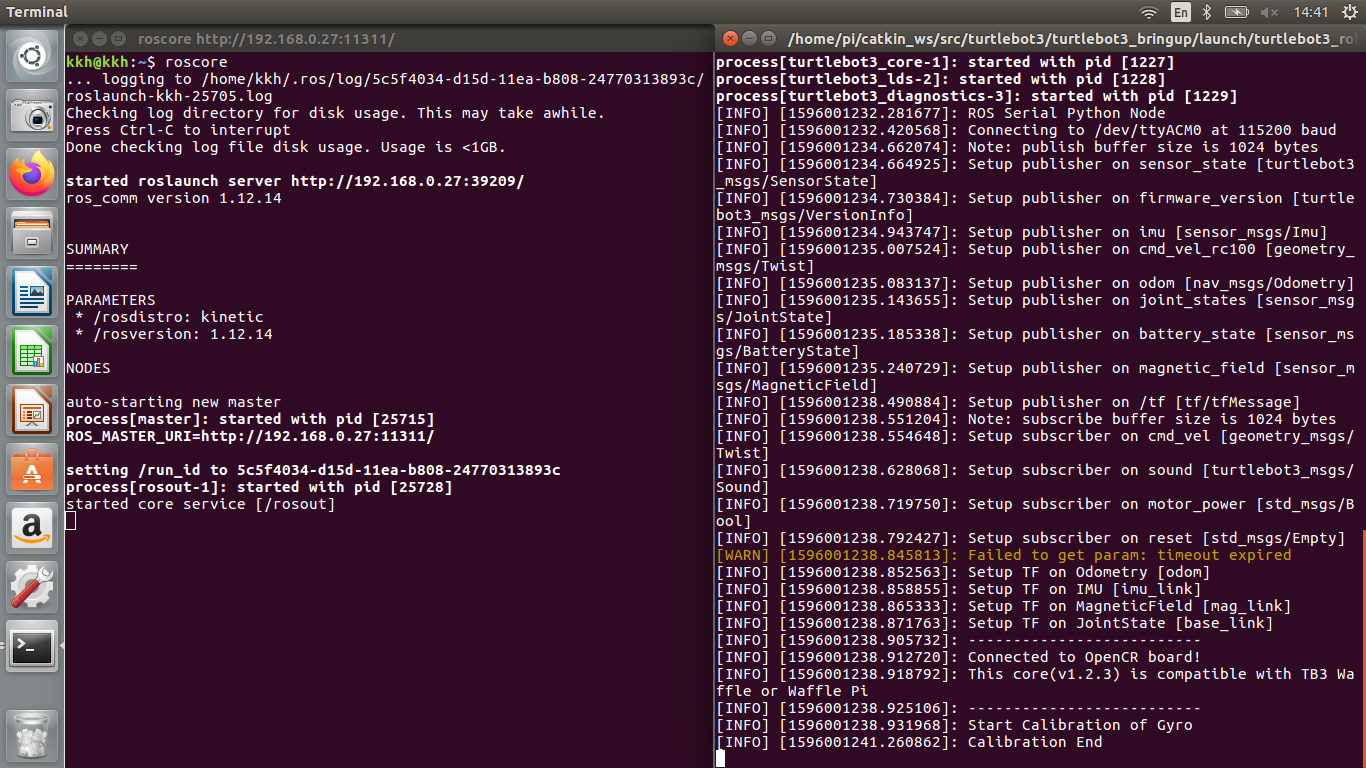
* Simulation
* 시뮬레이션은 가짜 노드라 불리는 fake node를 실행함으로서 구동된다. 하지만, 센서를 사용할 수 없으므로, SLAM과 같은 기능을 수행하는 데에 부족함이 있다. 그래서 IMU, LDS, 카메라 같은 센서를 사용할 수 있는 Gazebo를 주로 활용하였다.

**연구 결과**

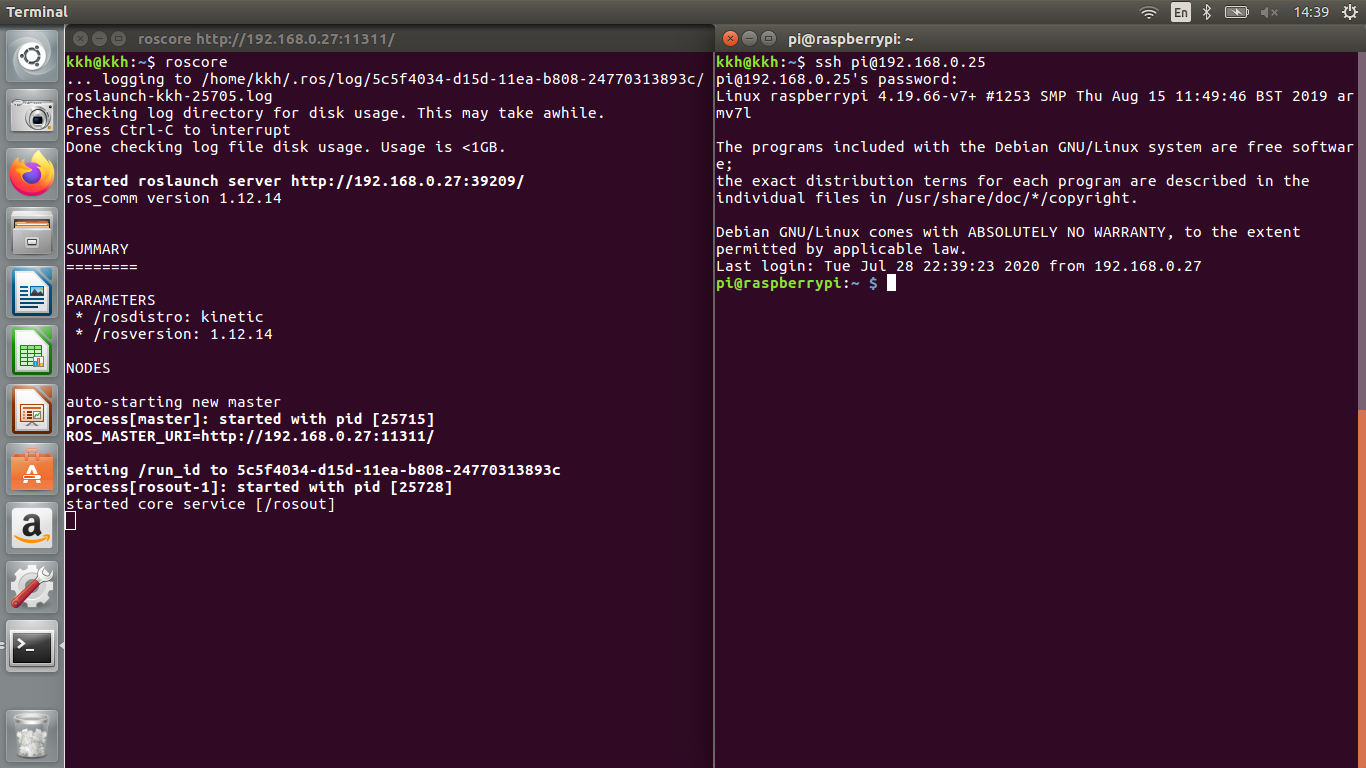
* Bring up



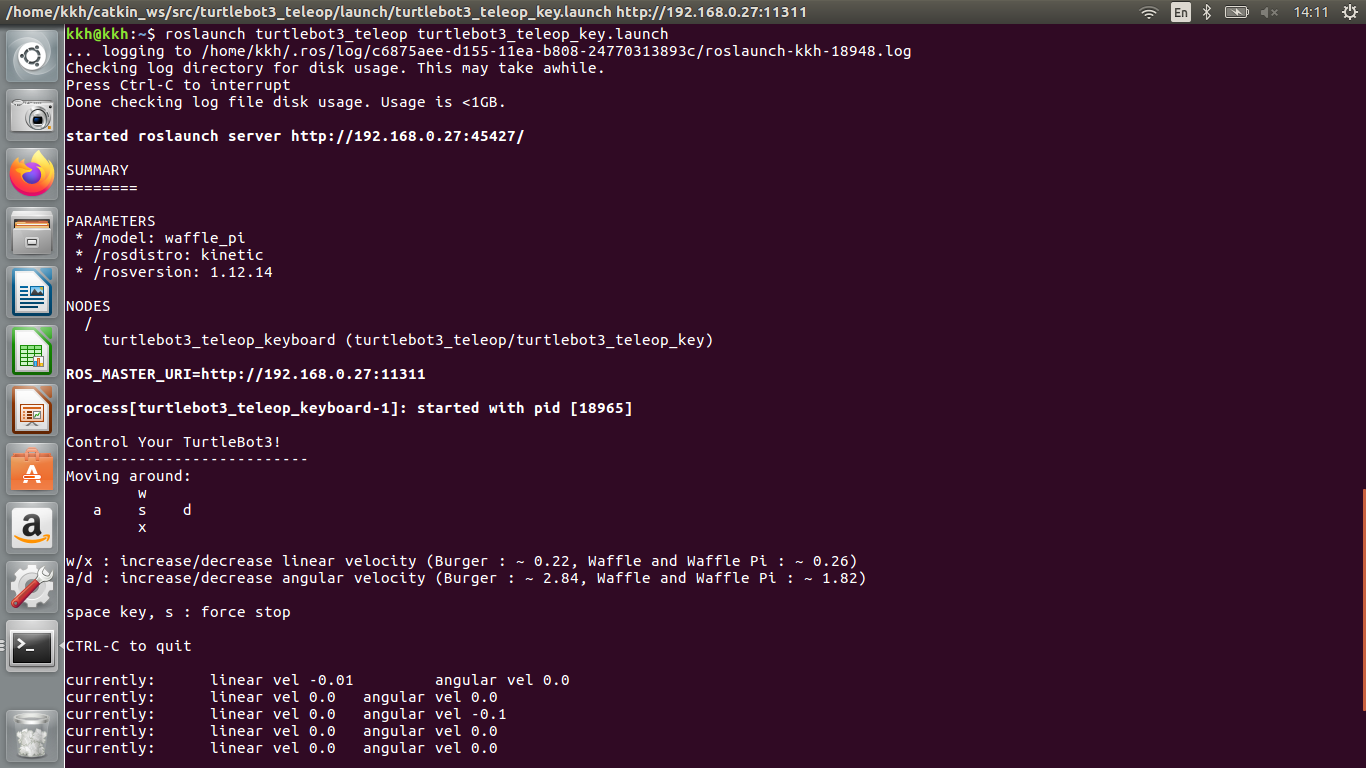
* Bring up 1



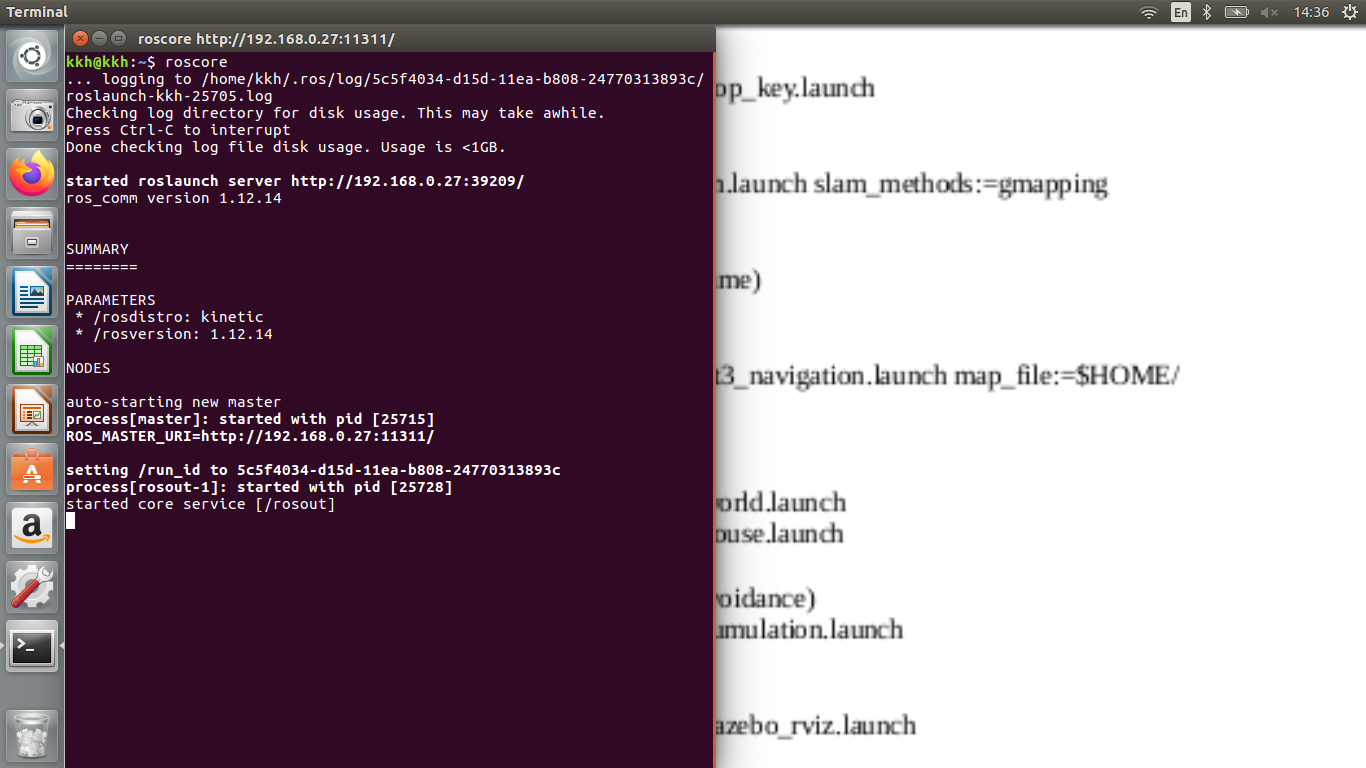
* Bring Up 2



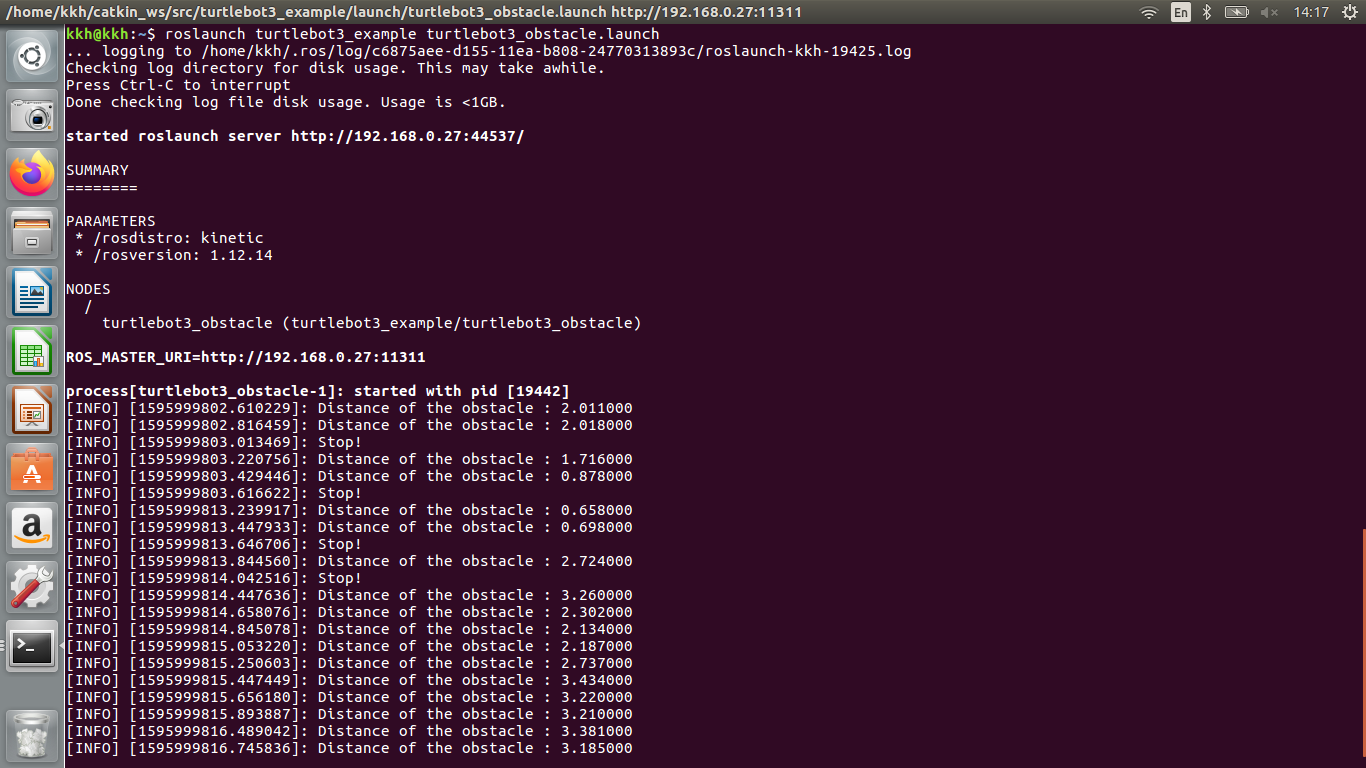
* Ssh\_pi



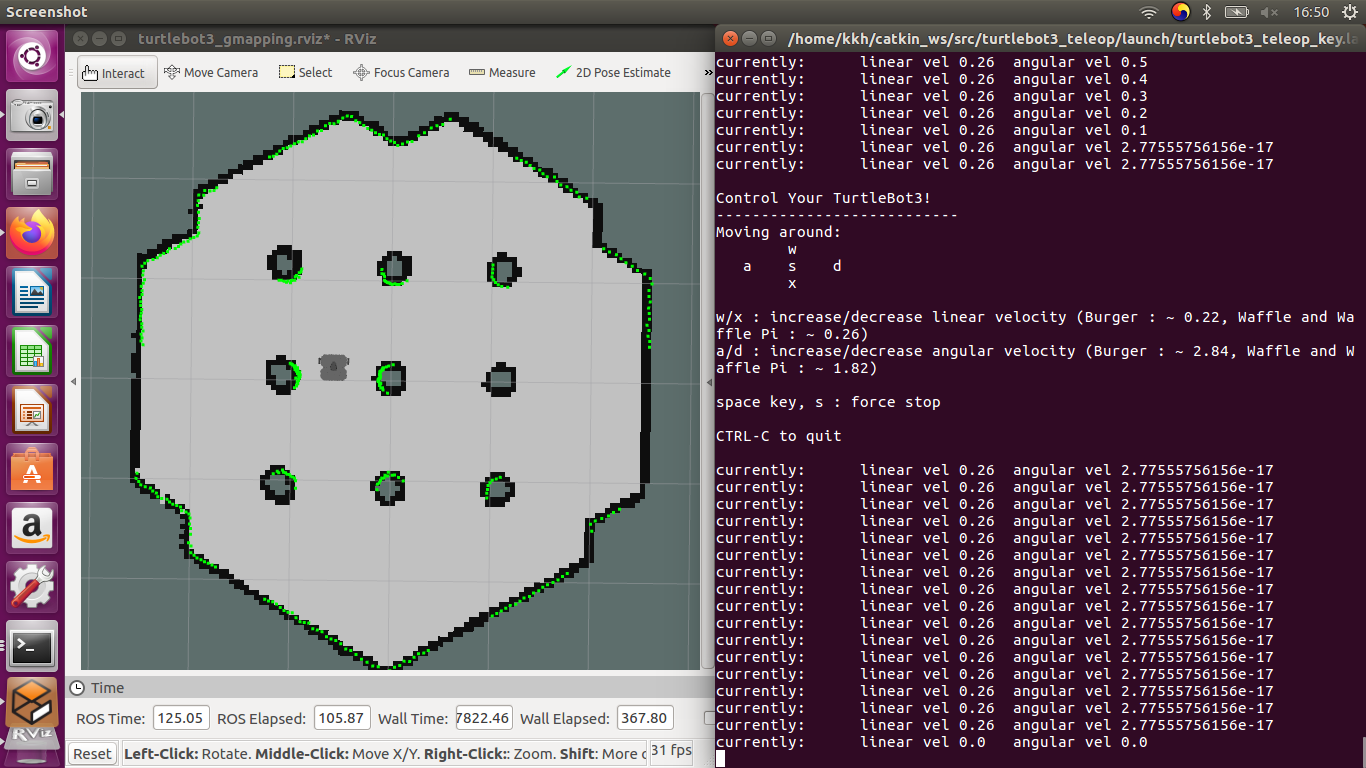
* Teleoperation\_keyboard



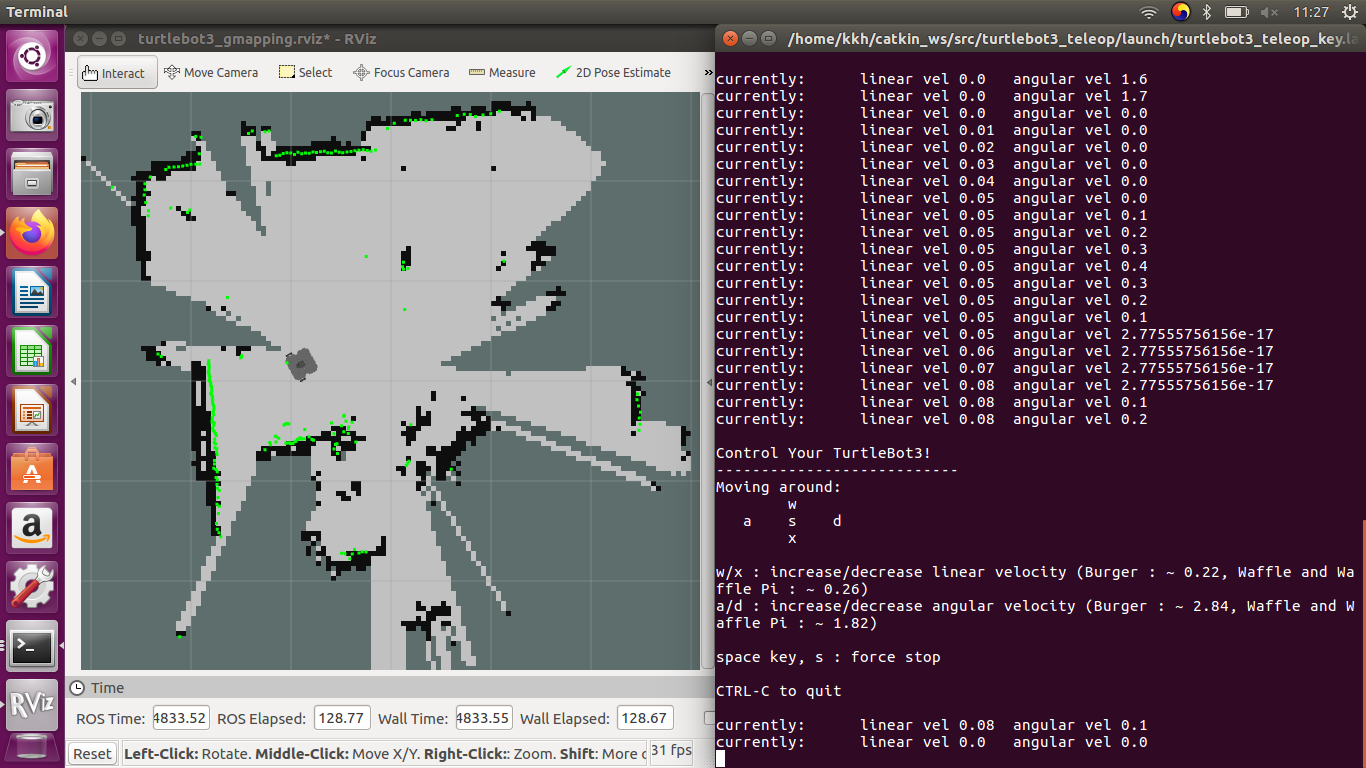
* roscore
* Basic Operation



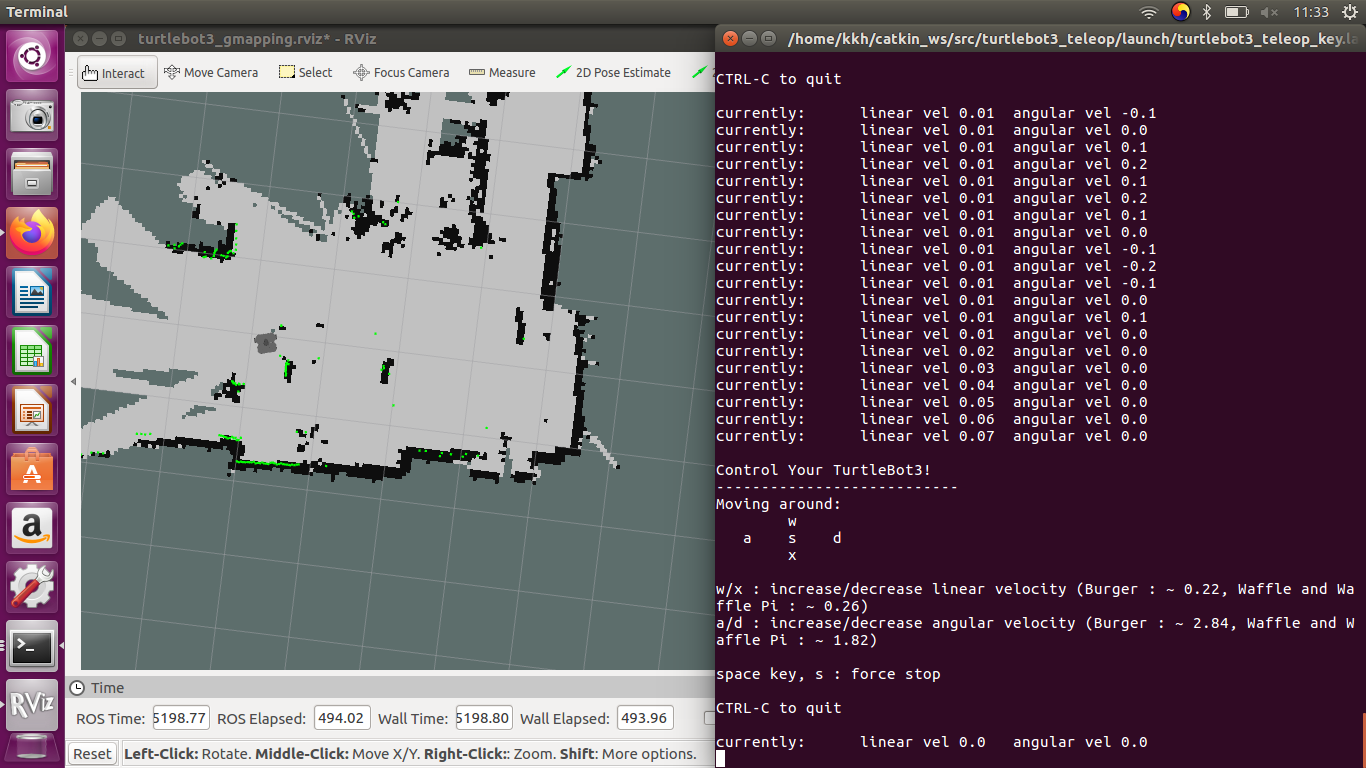
* SLAM



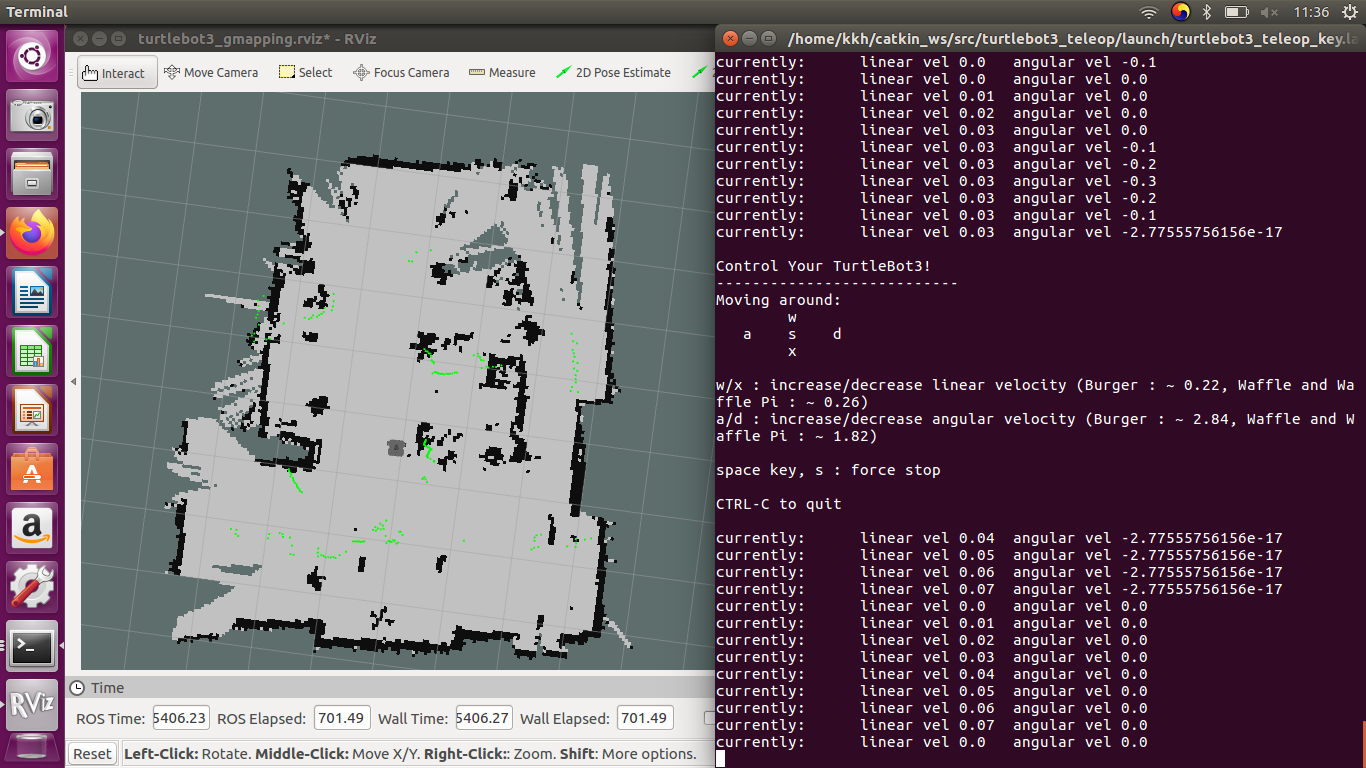
* G-mapping turtlebot world



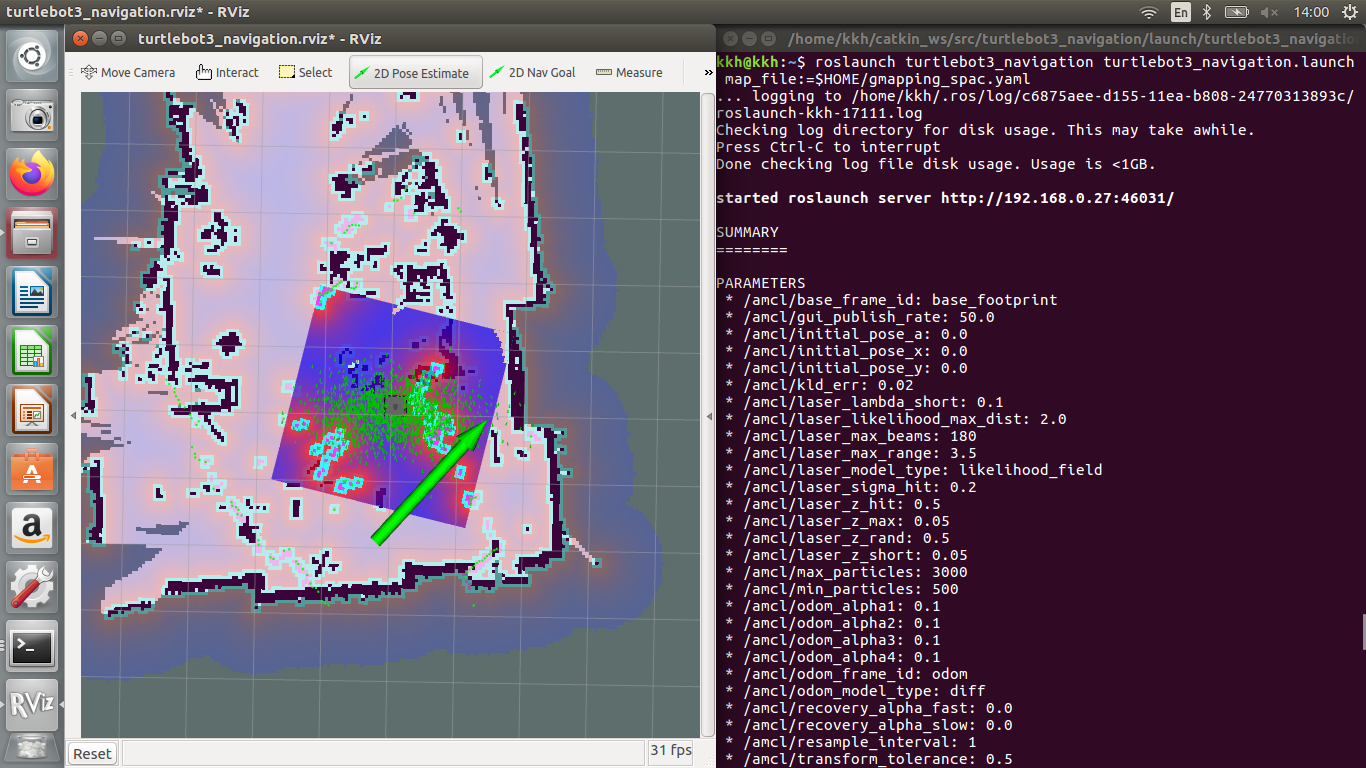
* G-mapping Spac 1



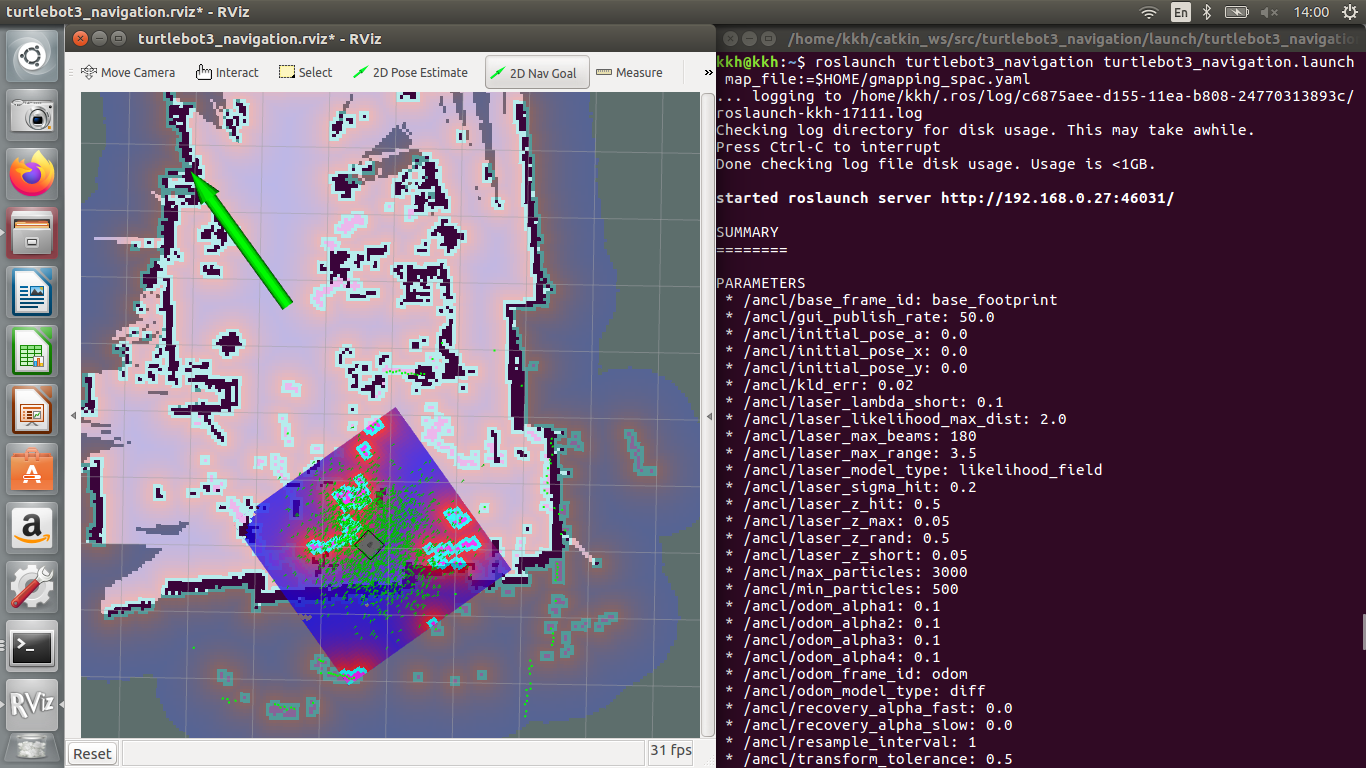
* G-mapping Spac 2



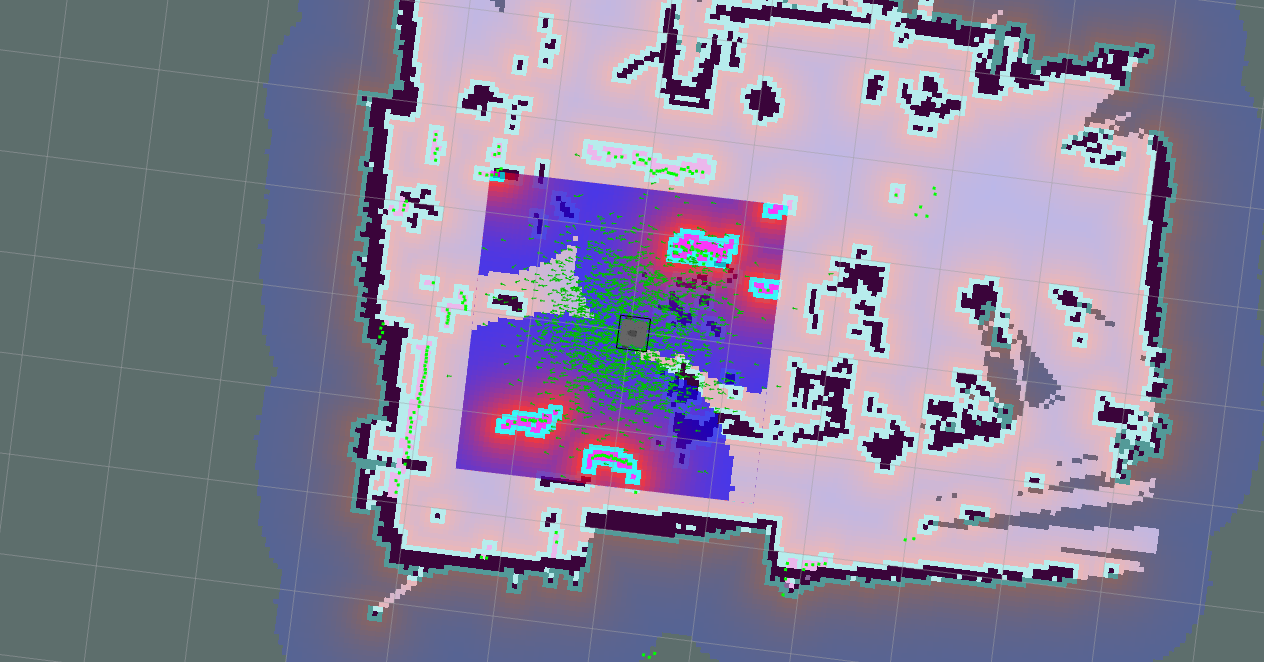
* G-mapping Spac 3
* Navigation



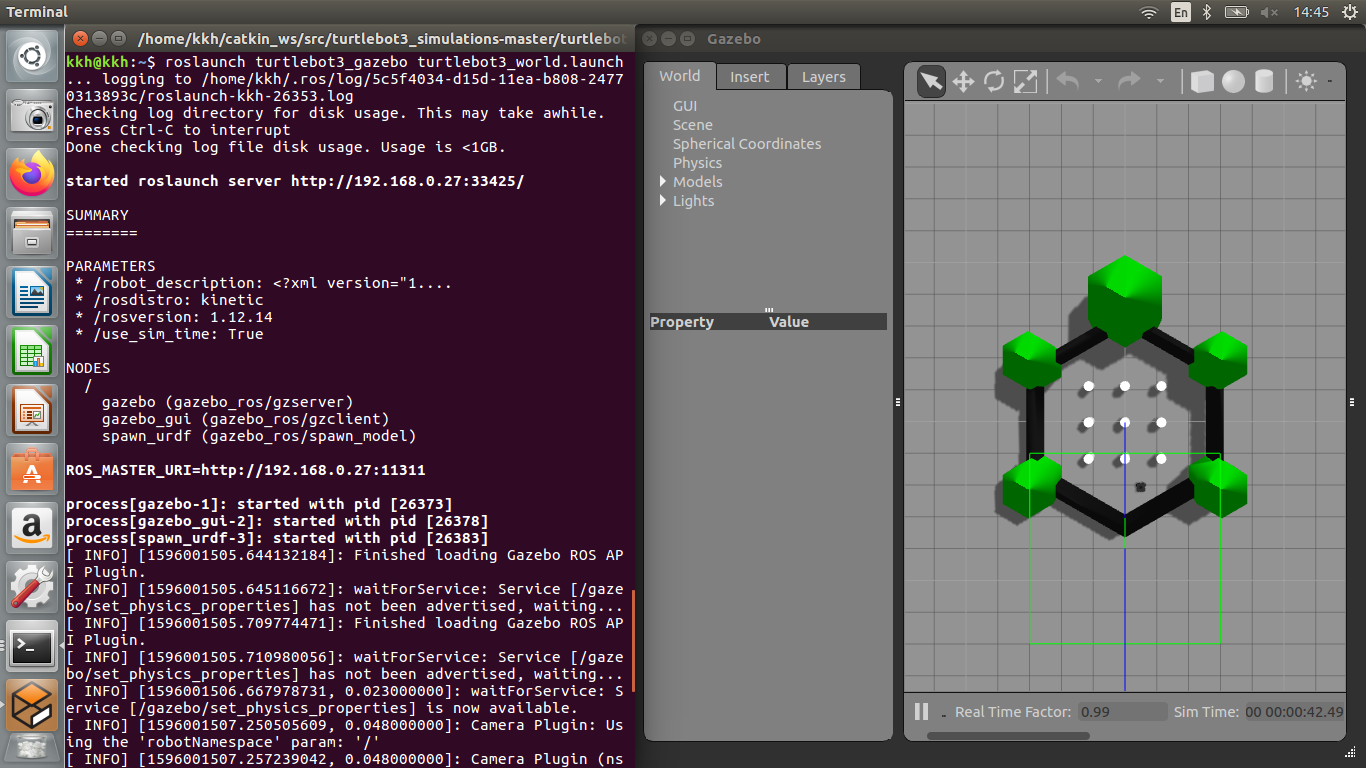
* Navigation\_start



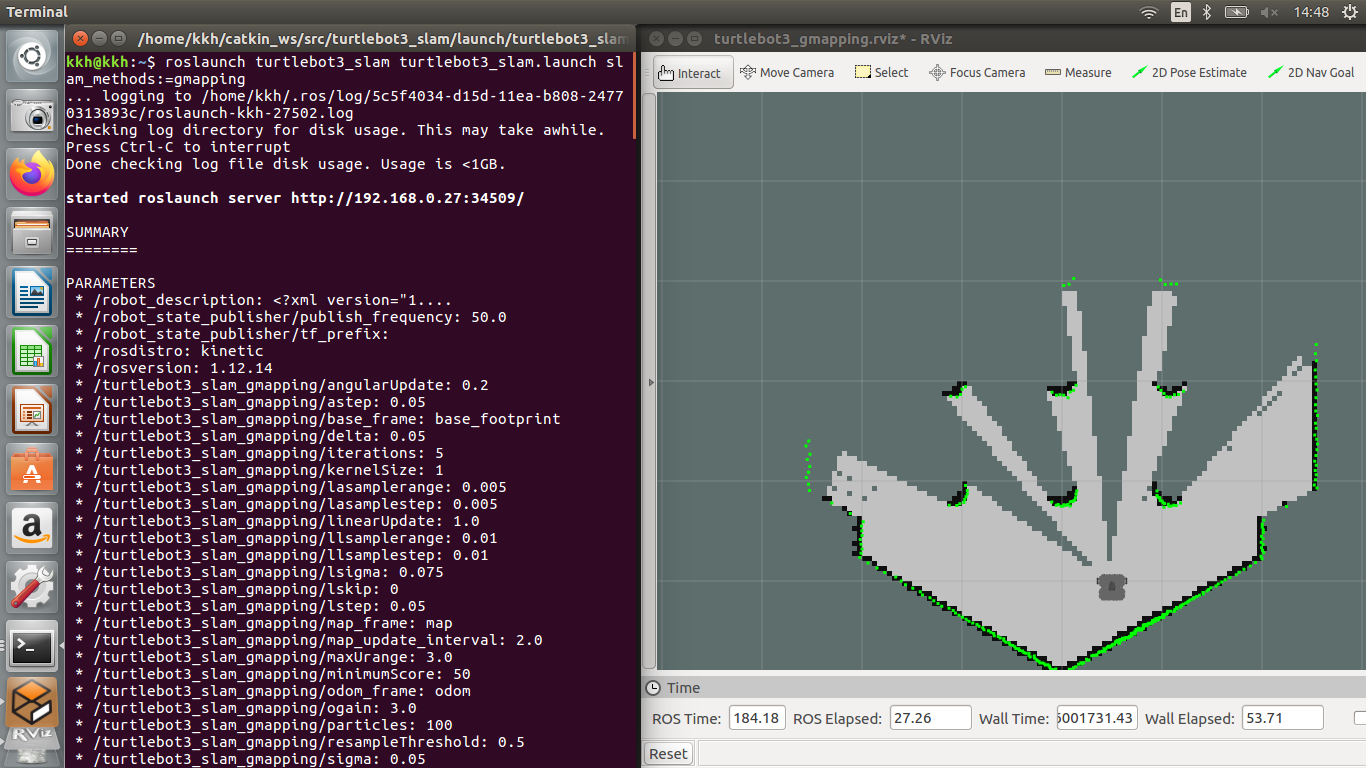
* Navigation\_end



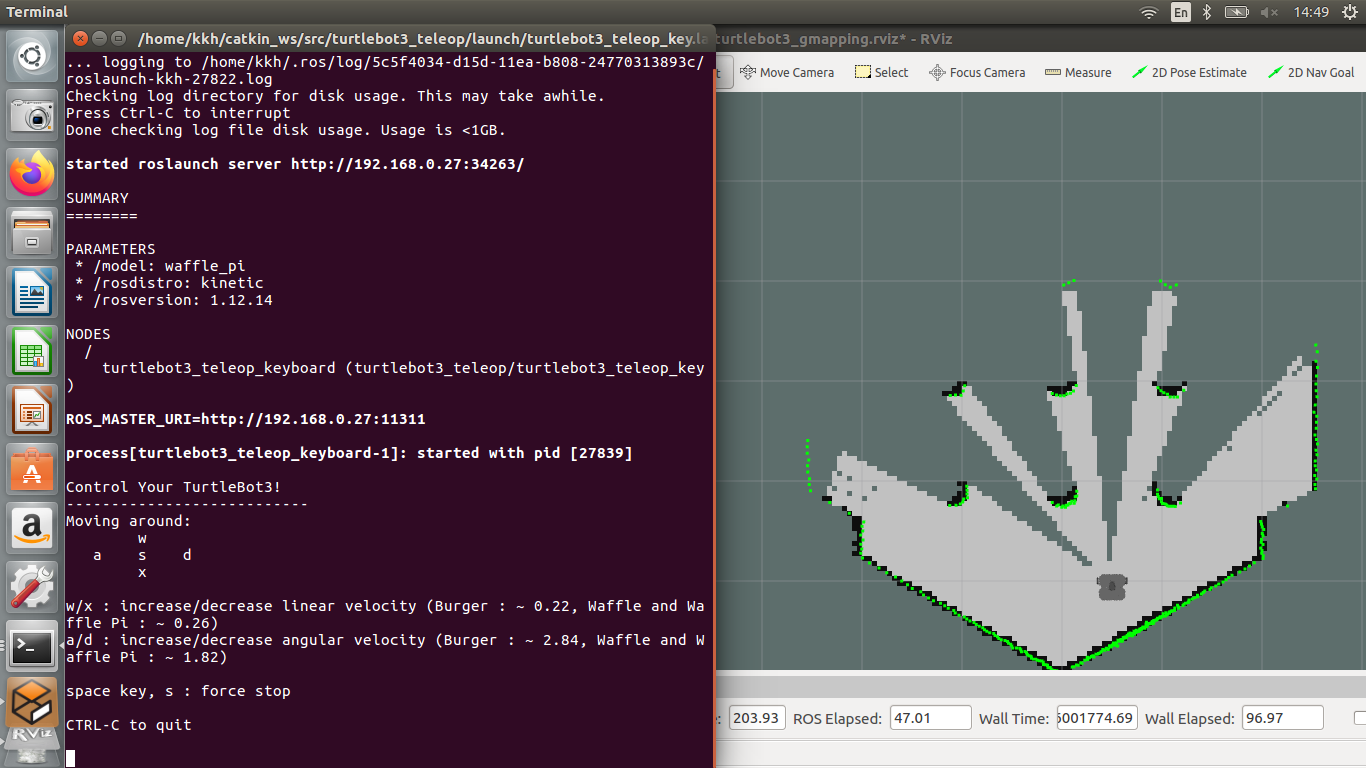
* Navigation\_G-mapping\_Spac
* Simualtion



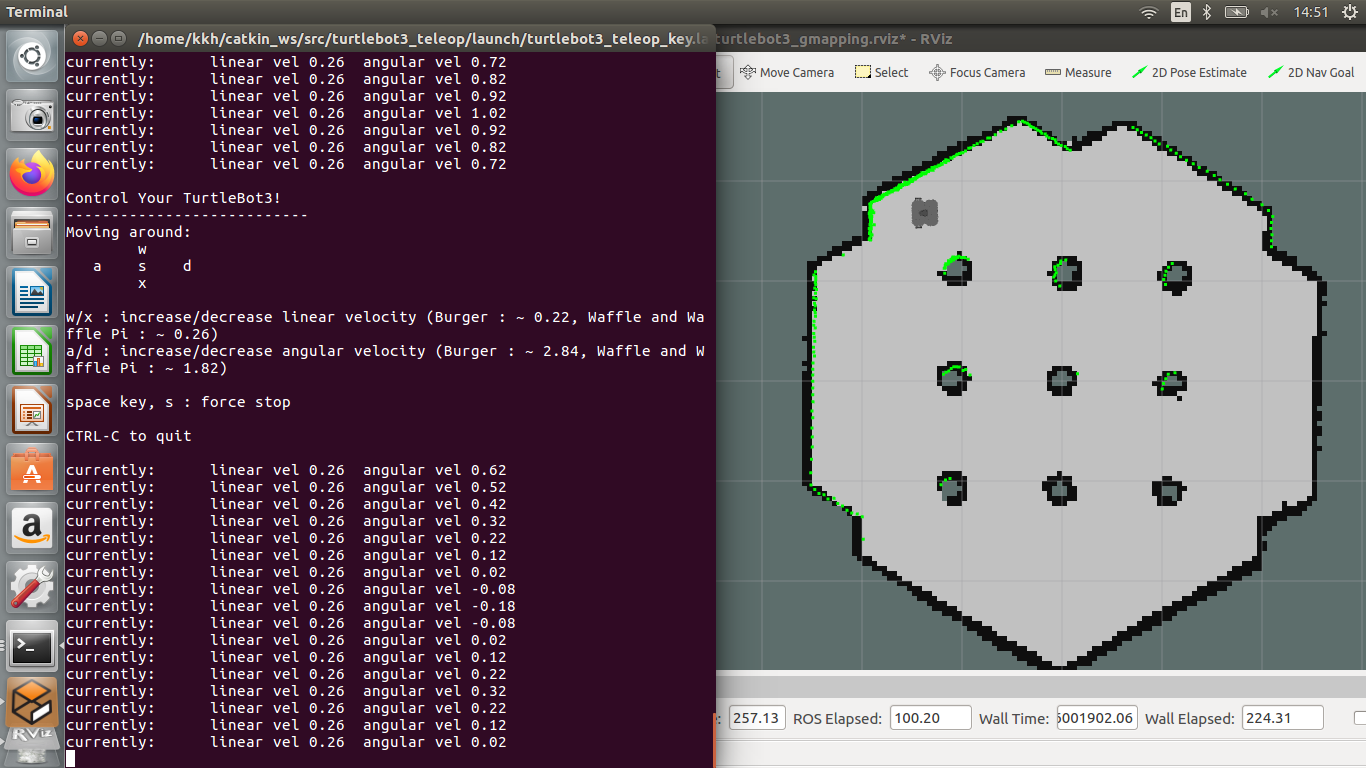
* Simulation world 1



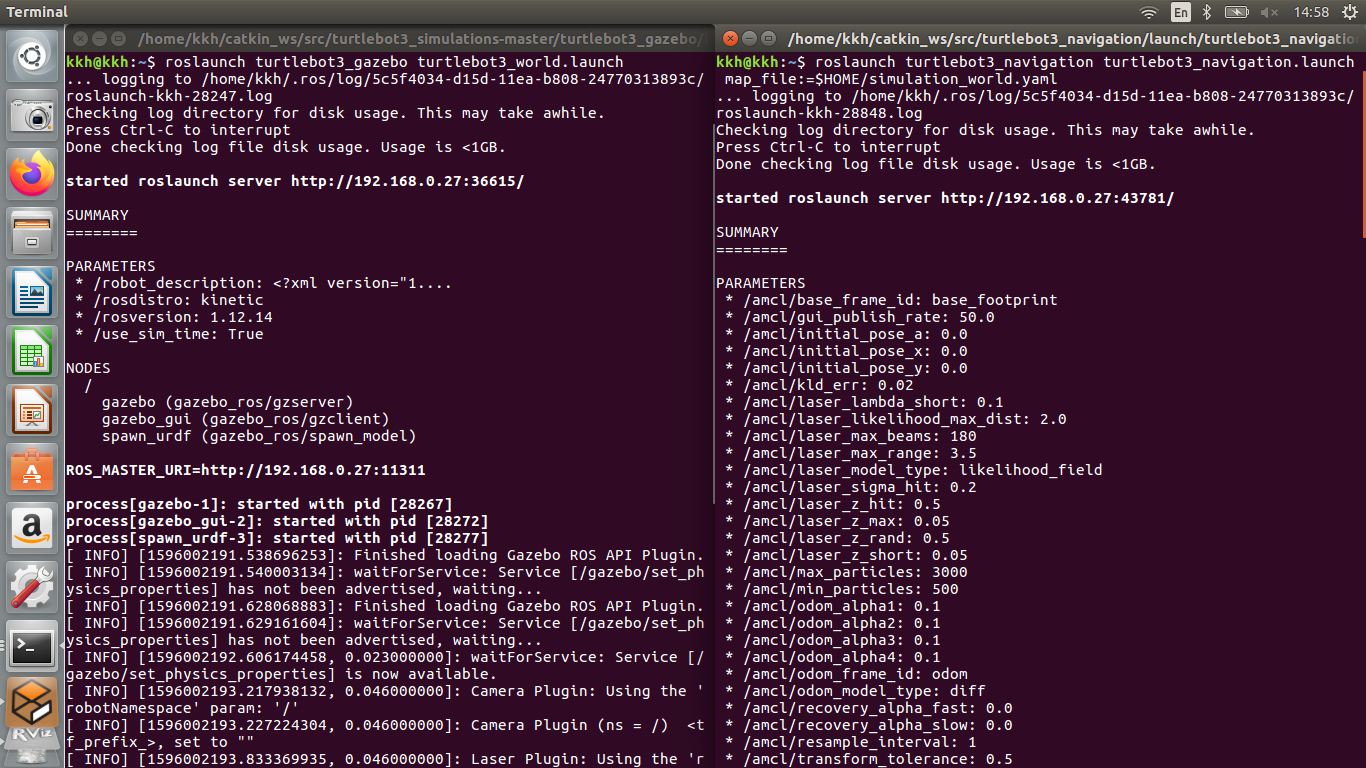
* Simulation world 2



* Simulation world 3



* Simulation world 4



* Simulation Navigation

**고찰**

* 이번 연구참여를 통해 우분투의 설치와 사용해 보았다. 우분투와 같은 리눅스 사용은 앞으로도 여러 개발 환경을 활용하는 데에 도움이 될 것으로 생각된다. 터미널을 활용하는 리눅스 os의 특성상 여러 커맨드를 암기해야 했는데, 숙련된 커맨드 사용을 위해 노력한 결과 많은 발전이 있었다. 거북이 로봇과 wifi로 동기화하여 원격 접속하는 과정도 매우 흥미로웠고, 이를 통해 원격 조종과, 실시간 데이터 확인하는 과정까지 직접 써넣기도 하고 오류를 해결하기 위해 우분투를 공부하는 등 스스로 공부하며 발전하는 계기가 되었다.
* 터틀봇의 bring up과 같은 기본 설정들을 설치하는 데 여러 시행착오들이 있었다. 주로 문제는 1. 우분투에 기본 파일들을 설치할 때 폴더의 설정의 문제가 있었다. 이는 현재 파일을 다운로드할 폴더의 위치가 잘못된 경우가 많아, 삭제 후 올바른 위치에 재설치 하면 된다. 2. 다운로드 시 몇몇 소스가 불완전한 경우가 있었다. 이런 경우 인터넷에 검색하여, 다른 사이트의 소스를 사용하는 등의 방법을 통해 해결하였다. 3. 데스크탑의 와이파이가 잡히지 않는 문제를 해결하기 위해서 Wi-Fi동글을 구매하였다. 하지만, iptime의 홈페이지에 구 버전의 리눅스용 소프트웨어만 소개되어 있어 인터넷에서 검색하여 시도해 보았지만, 결국 실패했다. Wi-Fi가 되는 노트북으로 진행했다.
* P.s. 연구실 분위기도 너무 좋고, 교수님도 너무 많이 배려해 주셔서 대학원에 대한 너무 좋은 인식을 갖게 되었습니다. 처음부터 끝까지 먼저 적극적으로 도와주신 형들과 같이 운동 재미있게 해주신 형들, 편하게 대해 주신 형 누나들 모두 감사드립니다. 교수님도 저희가 연참생인데도 돈 지원해 주시고, 부담 없게 참여할 수 있는 환경 조성해 주셔서 감사했습니다.

**참고 문헌**

* 로봇 프로그래밍, ROS로 시작하자! - 표윤석 지음
* <http://wiki.ros.org/> - 참고 사이트

**학부생 연구 참여 결과 보고서**

**-Turtlebot3를 이용한 SLAM과 Navigation 기술 구현-**

2020.07.31 김근호, 문병필

============아래 내용은 연구 핵심 결과를 한 페이지에 정리한 것임. ============

