산학프로젝트 주간 보고서 ( 2019. 04. 04)

이름/팀명 : 장 환석/AP 프로젝트명 : AI planning

**1.개인별 주간 목표(상세 개발 계획서)**

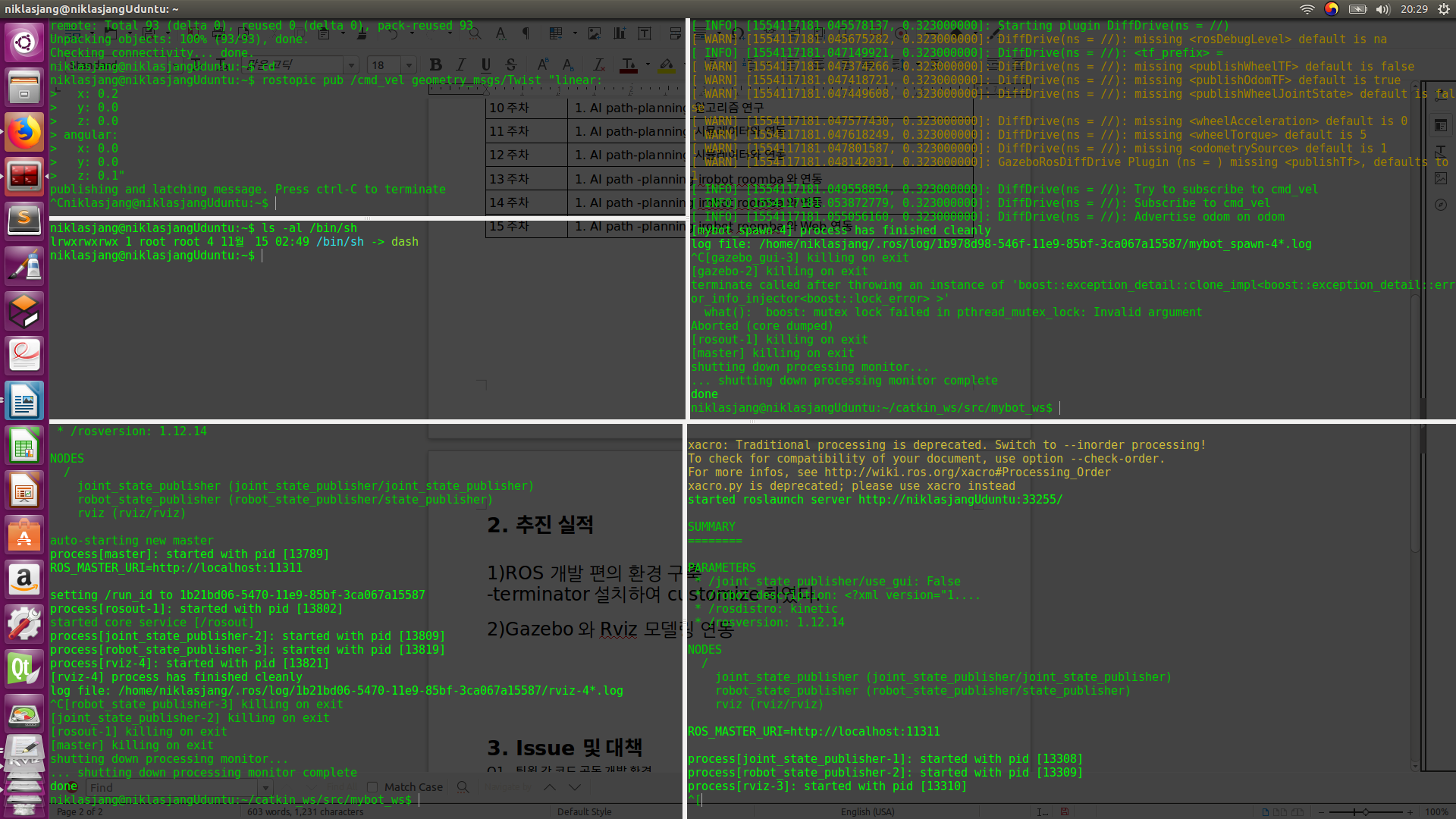
|  |  |
| --- | --- |
| 1주차 | 프로젝트 설명회 |
| 2주차 | 팀 빌딩 확정, 프로젝트 제안서 논의 |
| 3주차 | 0. 물품 구매 요청서 및 사유서 작성  1. ROS 환경 이해 및 ROS 기본 topic publish/subscribe 구현  2. Rviz, Gazebo 기본 개념 이해 |
| 4주차 | 1. ROS 개발 편의 환경 구축  2. Gazebo와 Rviz 모델링 연동 |

**2. 추진 실적**

1)ROS 개발 편의 환경 구축

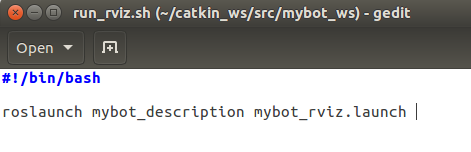
-terminator customize-

수많은 shell들을 간편하게 통제할 수 있게 되었다.



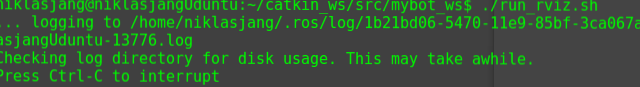
-shell script의 이해-

.sh파일을 작성하여 긴 명령어를 빠르게 칠 수 있도록 하였다.



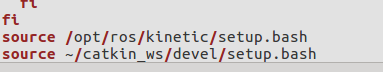
(#!/bin/bash는 \*스크립트 파일을 bash로 실행할 것을 명시해주는 역할)

(스트립트 파일(.sh) : 텍스트 형식으로 저장되는 프로그램으로서 한 줄씩 순차적으로 읽어 실행되도록 작성된 프로그램)

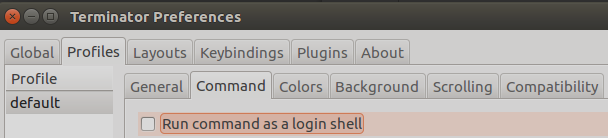


-auto source environmental variable-

~/.bashrc파일에 source ~/…/setup.bash를 추가했다. 이로 인해 새로운 terminal을 실행 후 더 이상 명시적으로 환경 변수와 path를 추가하지 않아도 자동으로 추가된다. 이유는 Non-login shell은 시작시 ~/.bashrc파일을 자동으로 실행시키기 때문이다. 



(위와 같이 명령을 추가한 뒤, 아래와 같이 login shell 설정을 uncheck하면 된다.)



**2)Gazebo와 Rviz 모델링 연동**

-Gazebo model 등 : .config, .sdf, .mesh-

~/.gazebo/models에 대한 .confi와 .sdf를 저장한다. .config파일은 model.sdf 파일을 참조하며 부가적인 메타정보를 담는다. model.sdf파일은 plugin(velodyne 등)에 대한 정보를 포함한다. ~/.gazebo/models는 가제보가 실행될 때 우선적으로 확인하는 디렉토리이다.

mkdir ~/.gazebo/models/velodyne\_hdl32

gedit ~/.gazebo/models/velodyne\_hdl32/model.config

<?xml version="1.0"?>

<model>

<name>Velodyne HDL-32</name>

<version>1.0</version>

<sdf version="1.5">model.sdf</sdf>

<author>

<name>Optional: YOUR NAME</name>

<email>Optional: YOUR EMAIL</email>

</author>

<description>

A model of a Velodyne HDL-32 LiDAR sensor.

</description>

</model>록

gedit ~/.gazebo/models/velodyne\_hdl32/model.sdf

model.sdf파일은 이미 만들어둔 velodyne.world의 내용을 복사해서 입력하면 된다. 단 , 두 개의 정보를 삭제해야한다.

1. The opening and closing <world> tags

2. The <include> tags for sun and ground plane

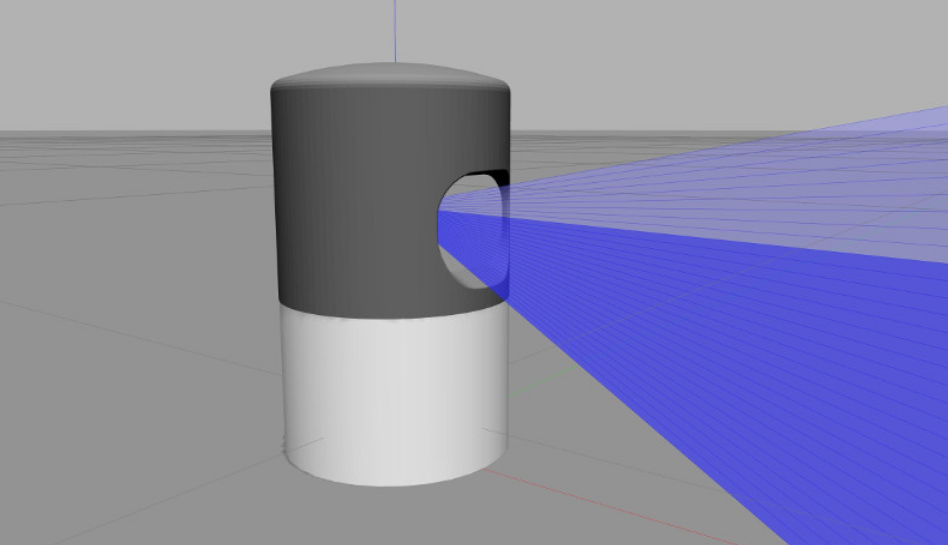
이제 gazebo –-verboss로 가제보를 실행하면 왼쪽tab > insert에 .sdf파일에 해당하는 모델이 자동으로 저장되어있다.

아래의 코드로 적절한 위치에 mesh 파일을 생성한다. 이렇게 생성된 mesh 파일은 model.sdf파일의 <visual>tag 안에서 <mesh>tag를 사용해서 적용될 수 있다.

mkdir ~/.gazebo/models/velodyne\_hdl32/meshes

cp velodyne\_base.dae ~/.gazebo/models/velodyne\_hdl32/meshes

cp velodyne\_top.dae ~/.gazebo/models/velodyne\_hdl32/meshes



<사진. Velodyne model 예제 실행 화면>

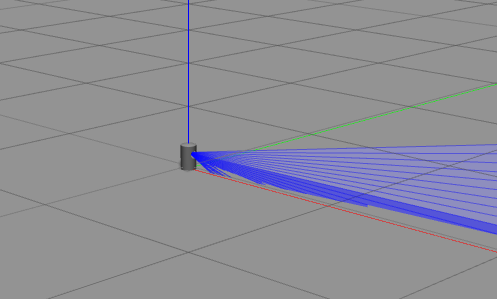
\*mesh file을 생성하는 blender사용 tutorial : <http://gazebosim.org/tutorials?cat=guided_i&tut=guided_i2>

-gazebo topic visualization-

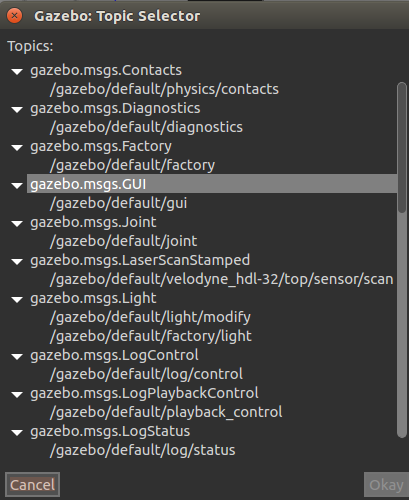
작성해놓은 .world파일을 실행한다. 이때 build 폴더에서 실행해야 ./libvelodyne\_plugin.so로 지정해둔 .so 파일을 찾을 수 있다.

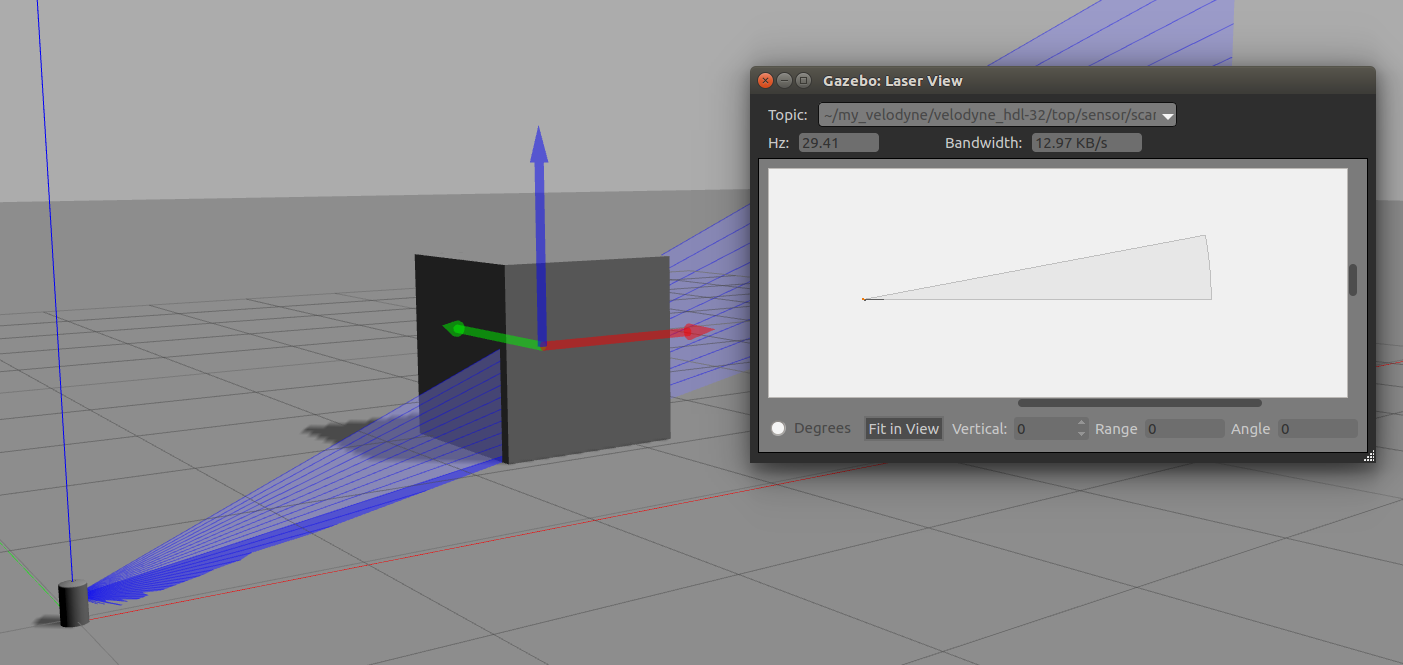
cd ~/velodyne\_plugin/build

gazebo --verbose ../velodyne.world



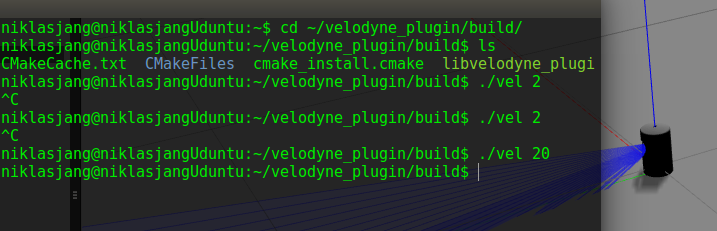
+gazebo에서 ctrl+t를 누르면 topic selector를 열 수 있다.





가장 기본적으로 gazebo의 내장 viewer를 사용해서 데이터를 시각화할 수 있다.

-gazebo node와 ros노드의 연동-



http://gazebosim.org/tutorials?tut=ros\_depth\_camera&cat=connect\_ros

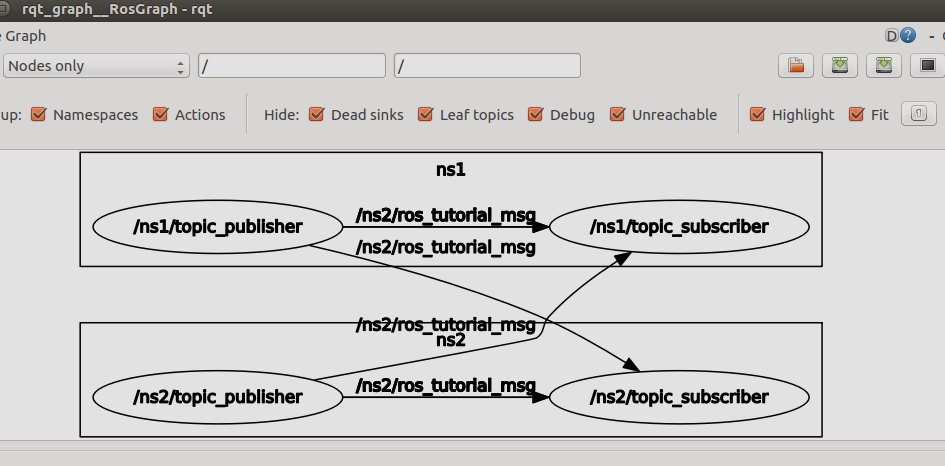
rostopic type /turtle1/cmd\_vel

You should get:

geometry\_msgs/Twist

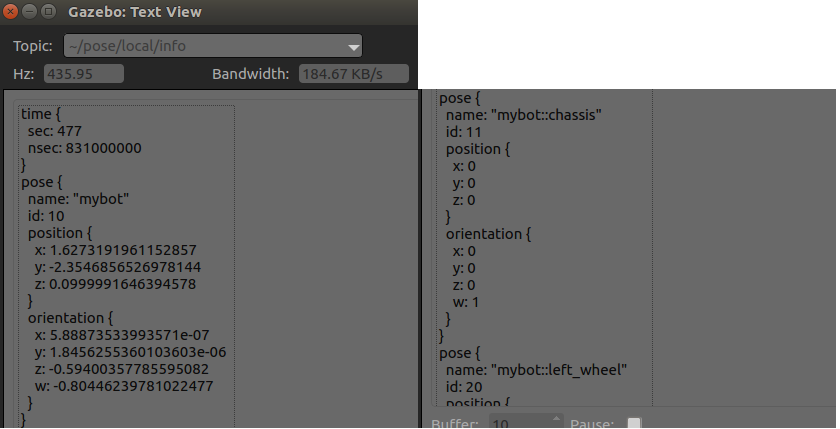
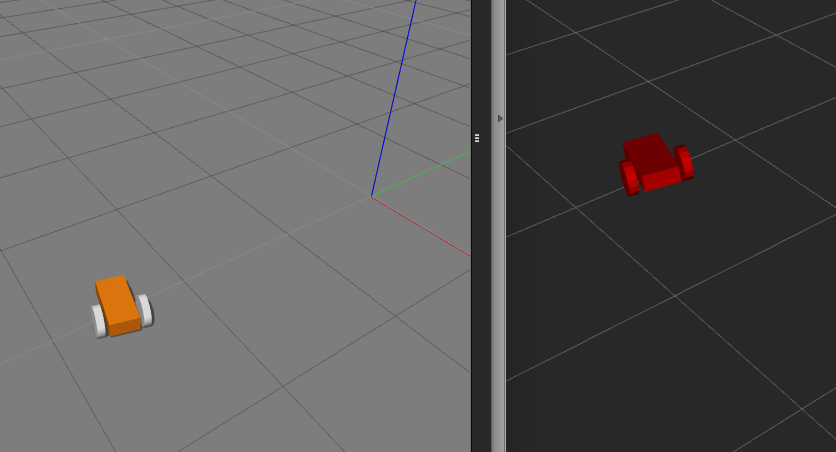
rosnode가 publish한 Topic을 gazebo의 노드가 subscribe하여 축의 속도를 제어하는 과정을 진행했다.

-namespace가 다른 노드 간 pub/sub-



launch 파일에서 하나의 노드를 namespace를 달리하여 여러번 실행시킬 수 있다. 이를 이용해서 namespace가 다른 노드간의 데이터 전송을 진행했다.

-Gazebo와 Rviz의 연동-



gazebo에서 publish되는 pose/local/info정보를Rviz에서 subscribe하여 시각화하는 과정을 진행했다.

**3. Issue 및 대책**

Q1. Gazebo에서 원하는 데이터를 publish하기 위한 과정이 복잡하다.

A1. plugin이 어떤 방식으로 topic을 publish하는지 자세한 분석이 요구된다.

Q2. Topic message만 활용하는 등 제한된 상황에서만 테스트 되었다.

A2. 보다 정확한 시뮬레이션 환경을 위해서 service와 action의 활용이 필요하다.