

# ROS의 다양한 개발도구

**ROBOTIS**

Open Source Team

Yoonseok Pyo



# ROS의 다양한 개발 도구

- 로봇 개발에 필요한 다양한 개발 도구를 제공
- 로봇 개발의 효율성 향상

## ■ Command-Line Tools

- GUI 없이 ROS에서 제공되는 명령어로만 로봇 액세스 및 거의 모든 ROS 기능 소화

## ■ RViz

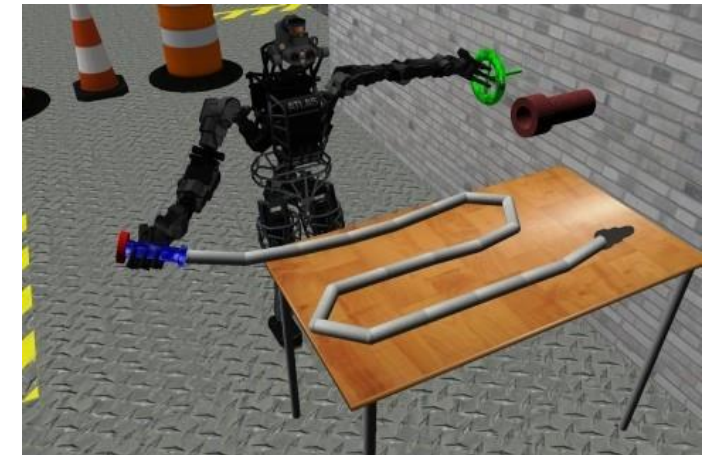
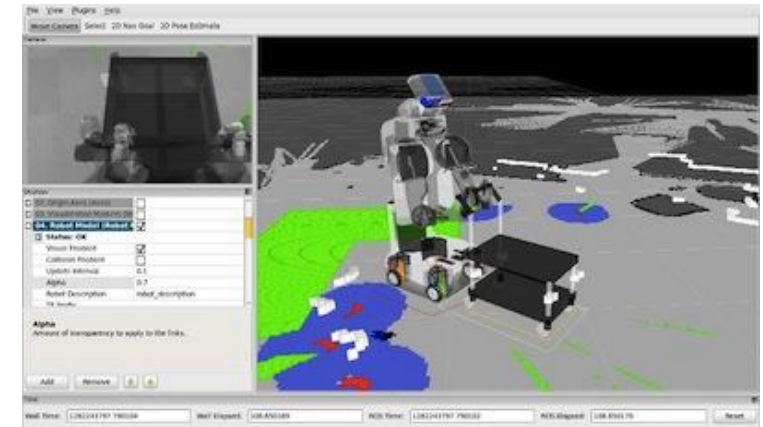
- 강력한 3D 시각화툴 제공
- 레이저, 카메라 등의 센서 데이터를 시각화
- 로봇 외형과 계획된 동작을 표현

## ■ RQT

- 그래픽 인터페이스 개발을 위한 Qt 기반 프레임 워크 제공
- 노드와 그들 사이의 연결 정보 표시(rqt\_graph)
- 인코더, 전압, 또는 시간이 지남에 따라 변화하는 숫자를 플로팅(rqt\_plot)
- 데이터를 메시지 형태로 기록하고 재생(rqt\_bag)

## ■ Gazebo

- 물리 엔진을 탑재, 로봇, 센서, 환경 모델 등을 지원, 3차원 시뮬레이터
- ROS와의 높은 호환성



# Index

---

**I. Command-Line Tools**

**II. Visualization Tool: Rviz**

**III. GUI Tool Box: RQT**

**IV. 3D Simulator: Gazebo**

# Index

---

**I. Command-Line Tools**

**II. Visualization Tool: Rviz**

**III. GUI Tool Box: RQT**

**IV. 3D Simulator: Gazebo**

# Command-Line Tools

---

아래의 **command-Line Tools** 는 ROS 강의 전반에 걸쳐서 지속적으로 사용됩니다.

rospack, roscd, rospd, rosls, rosed, roscp, rosdep,  
roswf, catkin\_create\_pkg, wstool, catkin\_make,  
roscore, rosrn, roslaunch, rosnod, rostopic,  
rosservice, rosparam, rosmg, rossrv, rosbag, tf\_echo

배워한 ROS cheatsheet 를 참고해주시고요.

[https://github.com/oroca/oroca\\_ros\\_tutorials/raw/master/ROScheatsheet\\_indigo\\_catkin.pdf](https://github.com/oroca/oroca_ros_tutorials/raw/master/ROScheatsheet_indigo_catkin.pdf)

이 부분은 생략!

# Index

---

**I. Command-Line Tools**

**II. Visualization Tool: Rviz**

**III. GUI Tool Box: RQT**

**IV. 3D Simulator: Gazebo**

# RViz (ROS Visualization Tool)

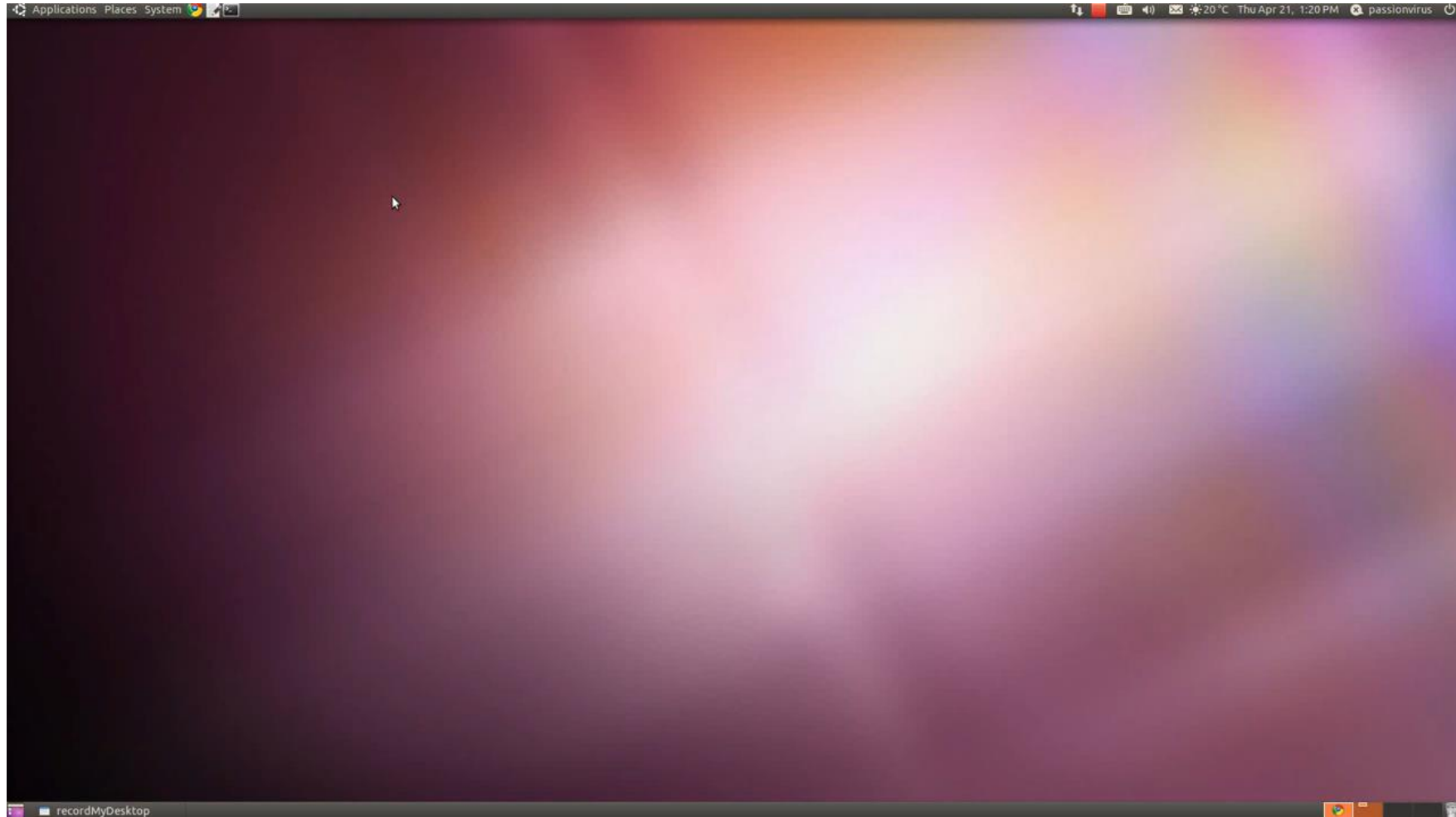
---

- ROS의 3D 시각화툴
  - 센서 데이터의 시각화
  - 레이저 거리 센서(LDS)센서의 거리 데이터
  - RealSense, Kinect, Xtion 등의 Depth Camera의 포인트 클라우드 데이터
  - 카메라의 영상 데이터
  - IMU 센서의 관성 데이터 등..
- 로봇 외형의 표시와 계획된 동작을 표현
  - URDF ( Unified Robot Description Format )
- 내비게이션
- 매니퓰레이션
- 원격 제어

# RViz의 사용예 #1

---

- Kinect의 Point Cloud Data

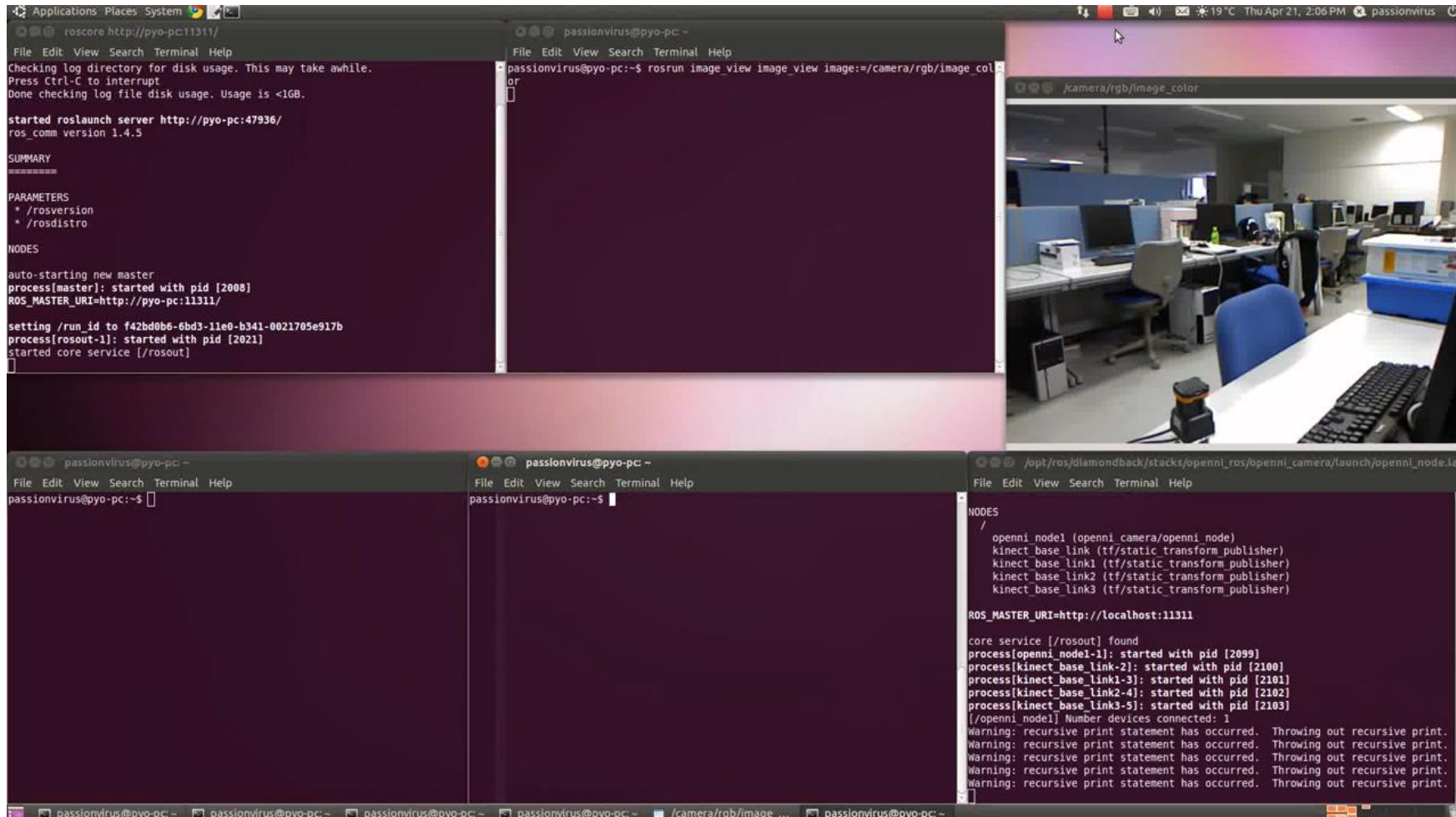


<https://youtu.be/OqOkpZBOpxY>



# RViz의 사용예 #2

- LRF의 거리 값

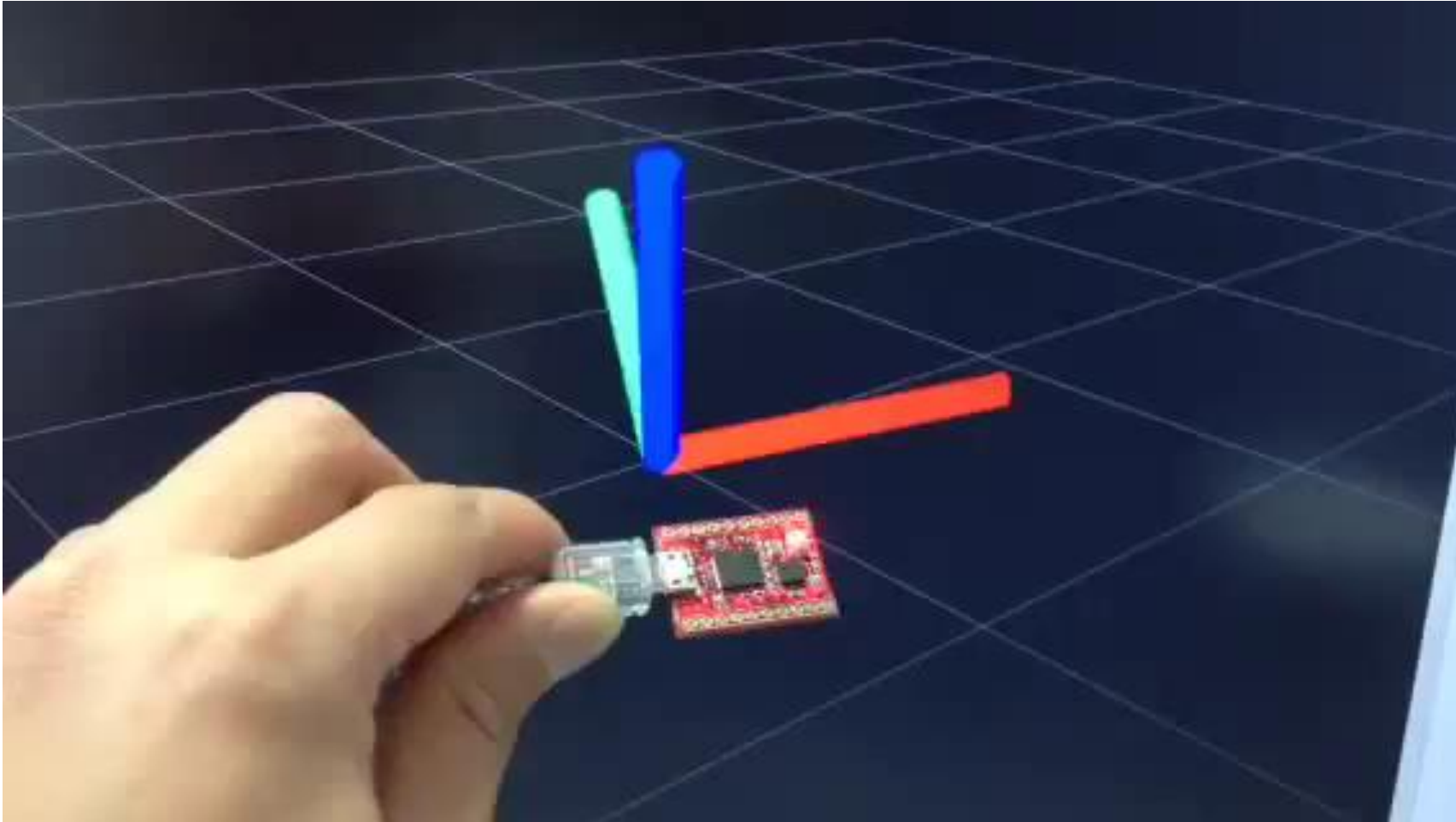


<https://youtu.be/qtoAJ1wzB6s>

# RViz의 사용예 #3

---

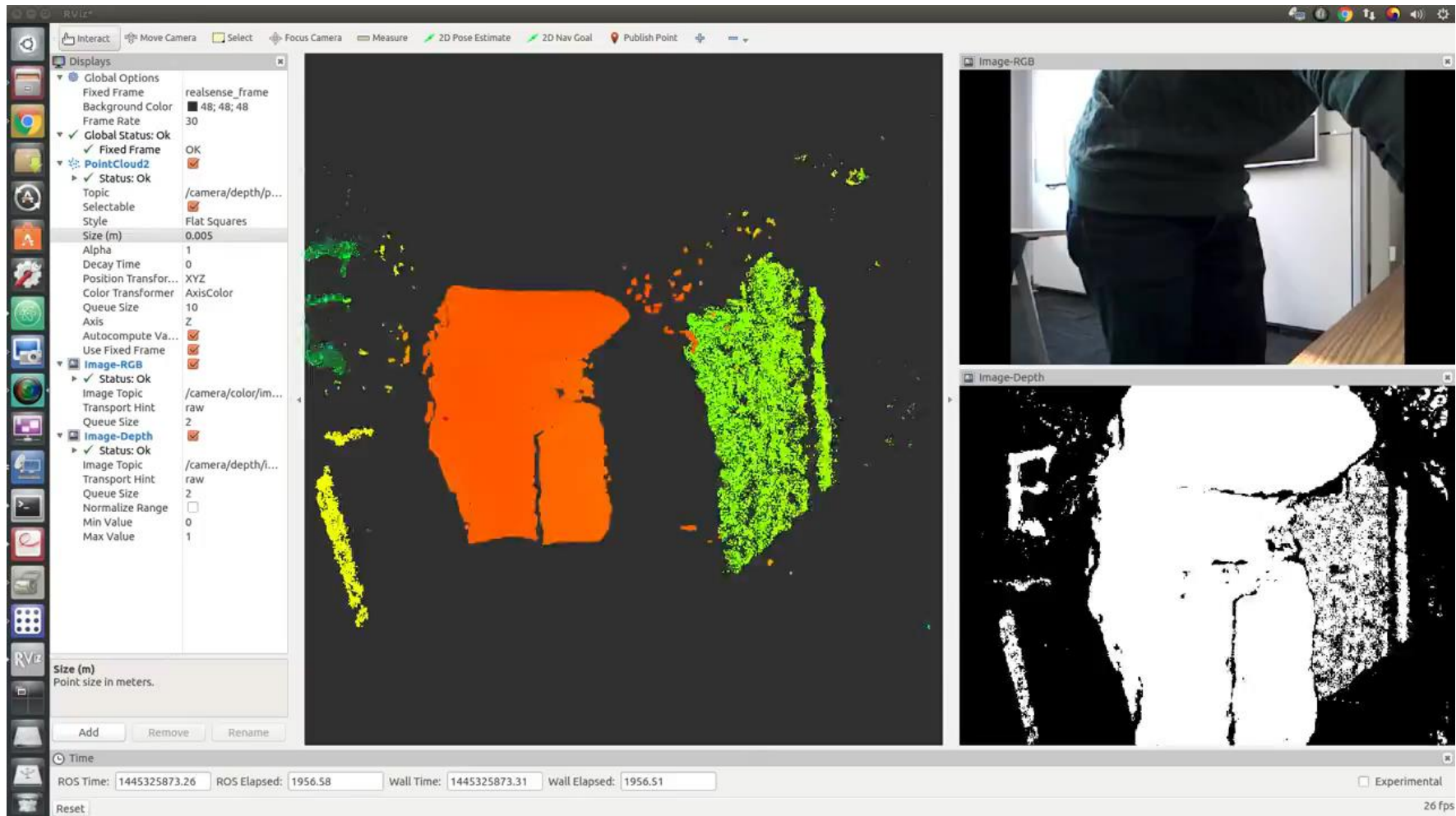
- IMU센서의 관성 값



<https://youtu.be/j5v5fKppcQo>

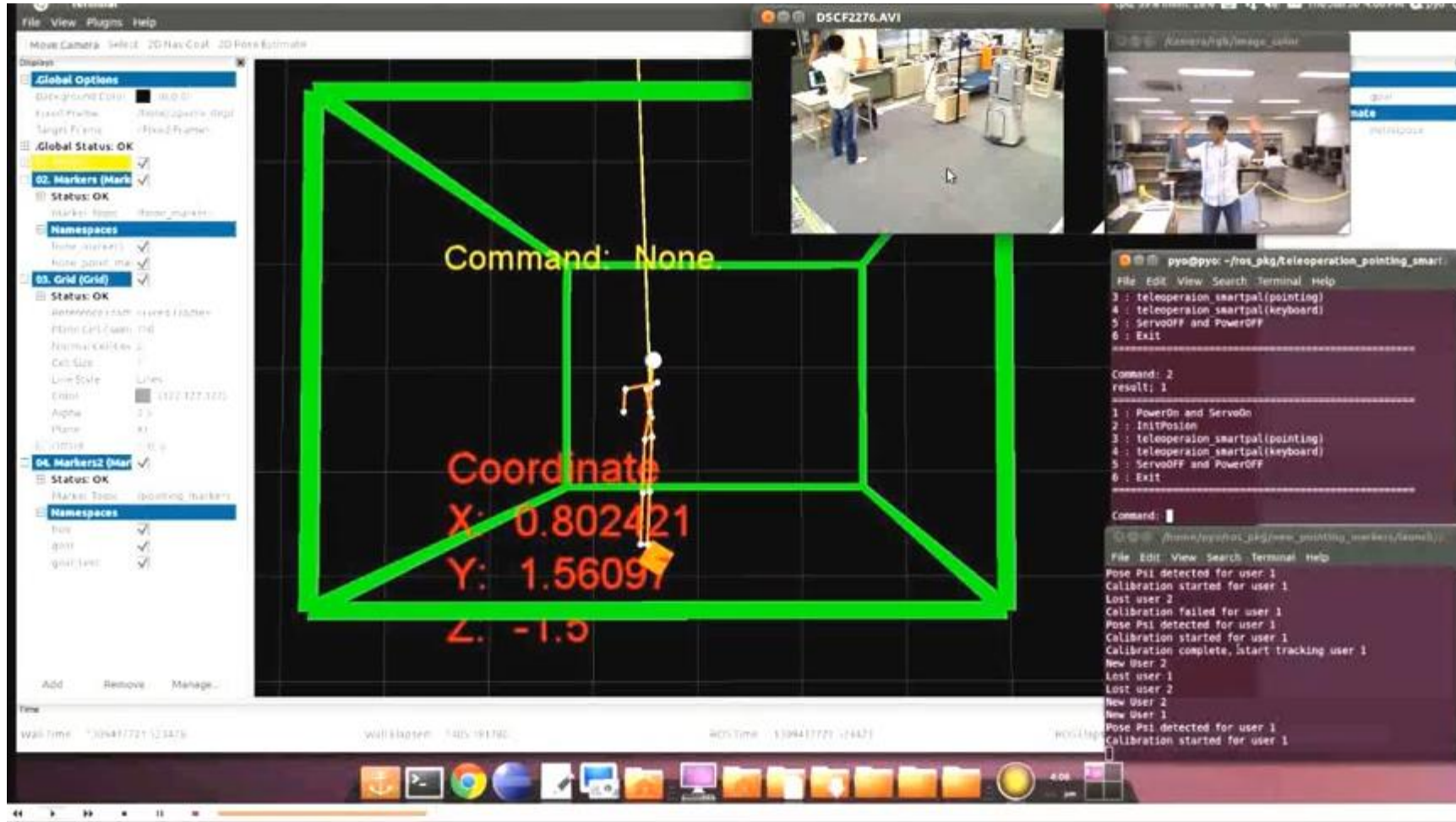
# RViz의 사용예 #4

- RealSense의 Point Cloud와 Color, Depth 영상



# RViz의 사용예 #5

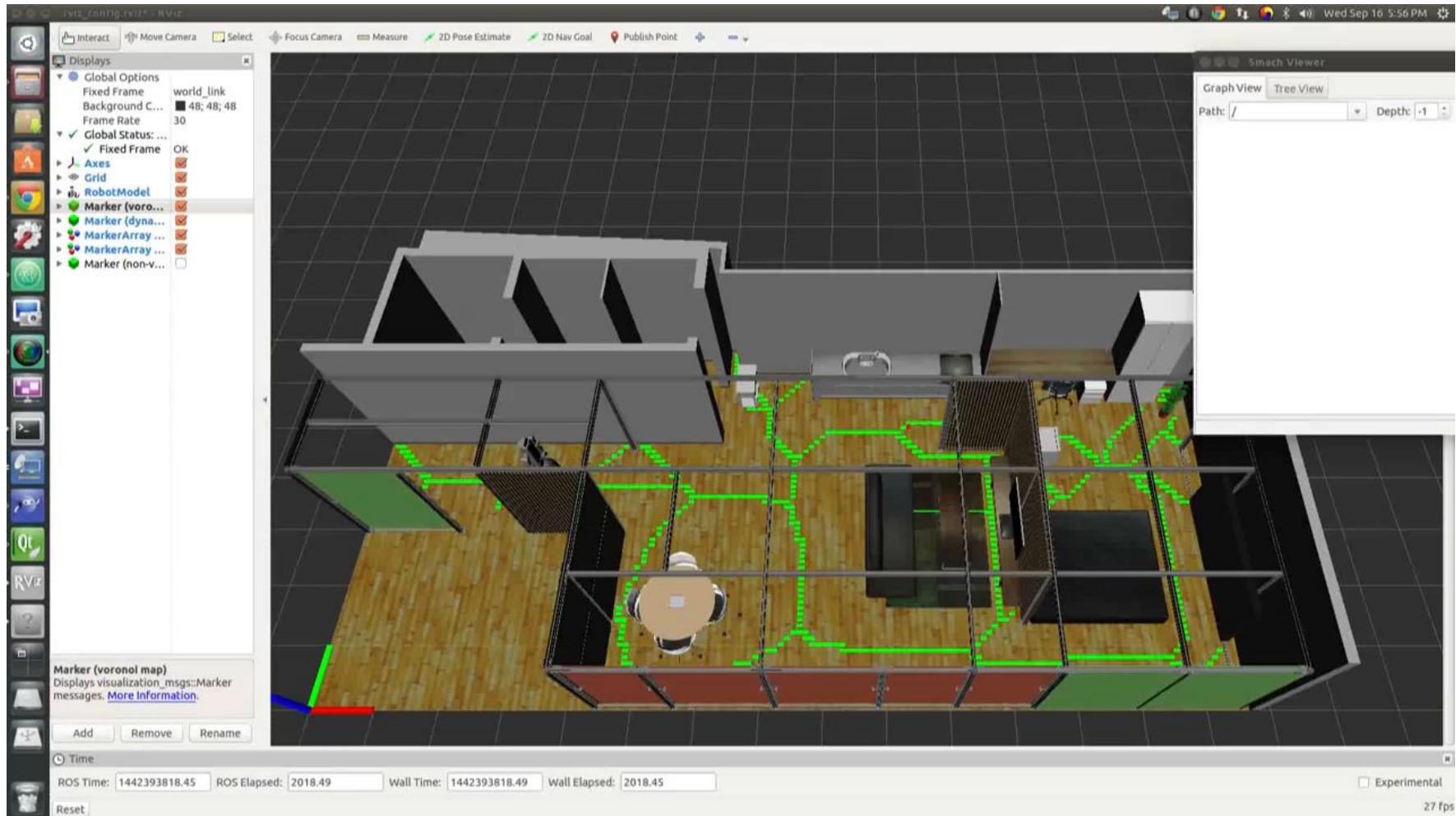
- 사람의 골격과 지시 방향 표시





# RViz의 사용예 #6

- 로봇 및 환경 모델, 경로까지



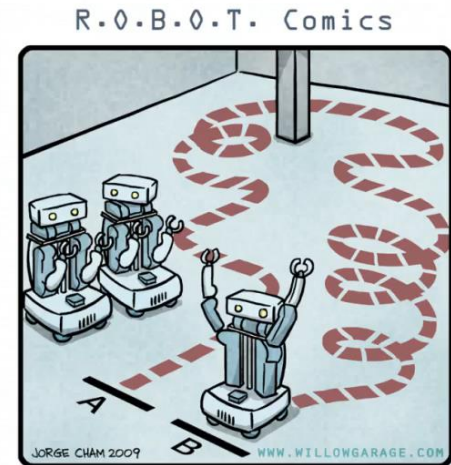
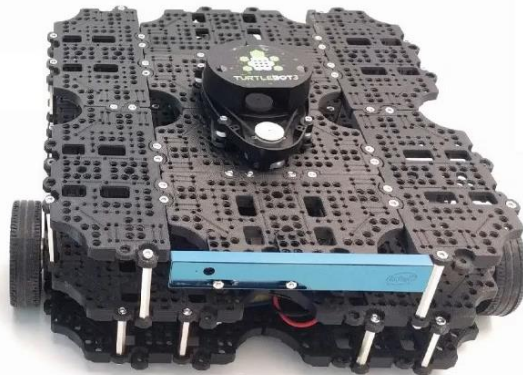
# RViz의 사용예 #7

- 지도 표시, 내비게이션, 목적지 지정

TurtleBot3  
BURGER ↺



TurtleBot3  
WAFFLE ↻

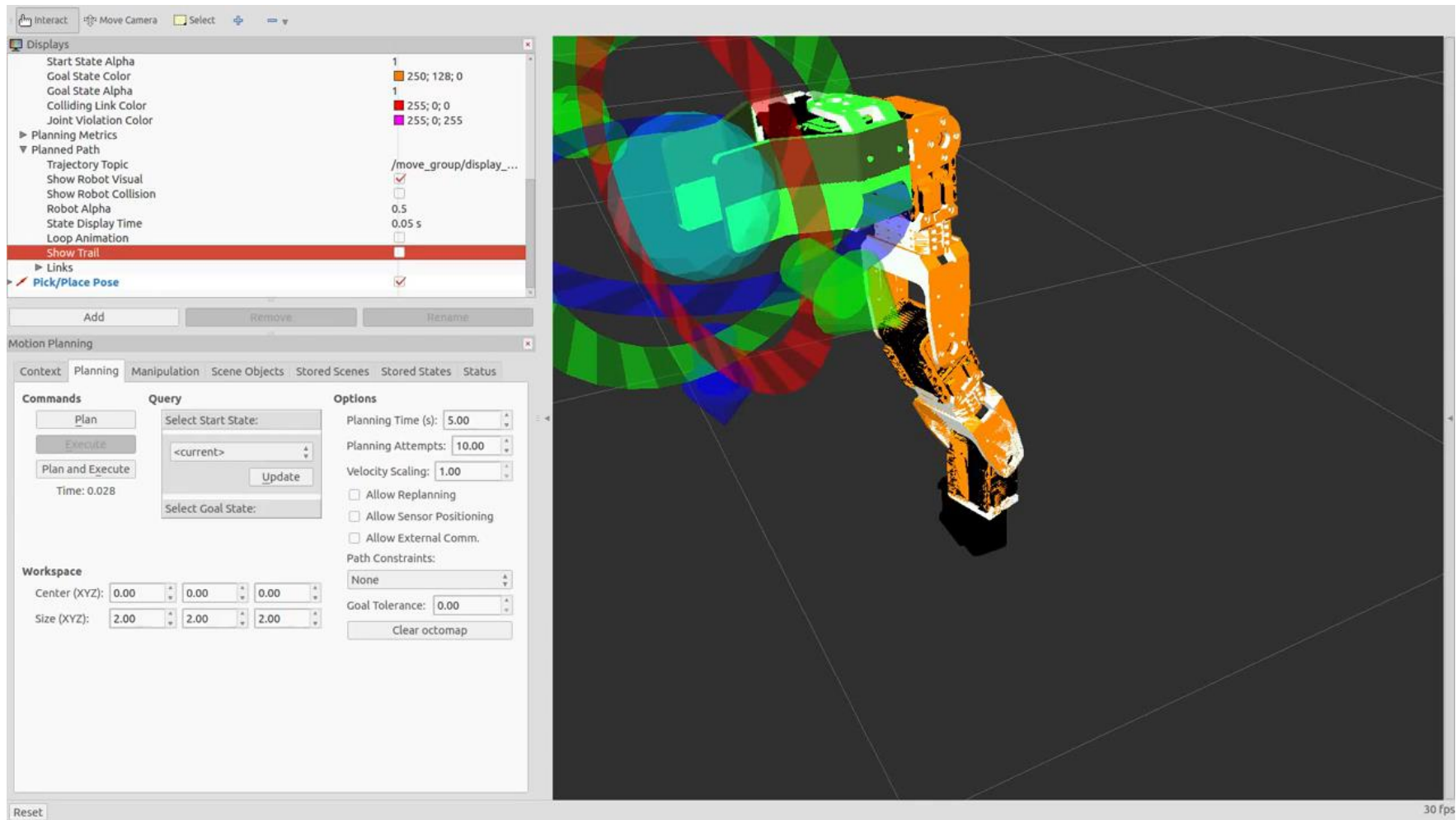


"HIS PATH-PLANNING MAY BE  
SUB-OPTIMAL, BUT IT'S GOT FLAIR."

**Navigation Demo**

# RViz의 사용예 #8

- 인터랙티브 마커를 이용한 IK 목표 위치 지정 및 경로 표시





# RViz의 사용예 #9

- 재난구조로봇의 경우 (2015 DARPA Robotics Challenge)





# RViz 설치 및 실행

---

- RViz 설치

```
$ sudo apt-get install ros-kinetic-rviz
```

\* ros-kinetic-desktop-full를 설치하였다면 기본 설치됨

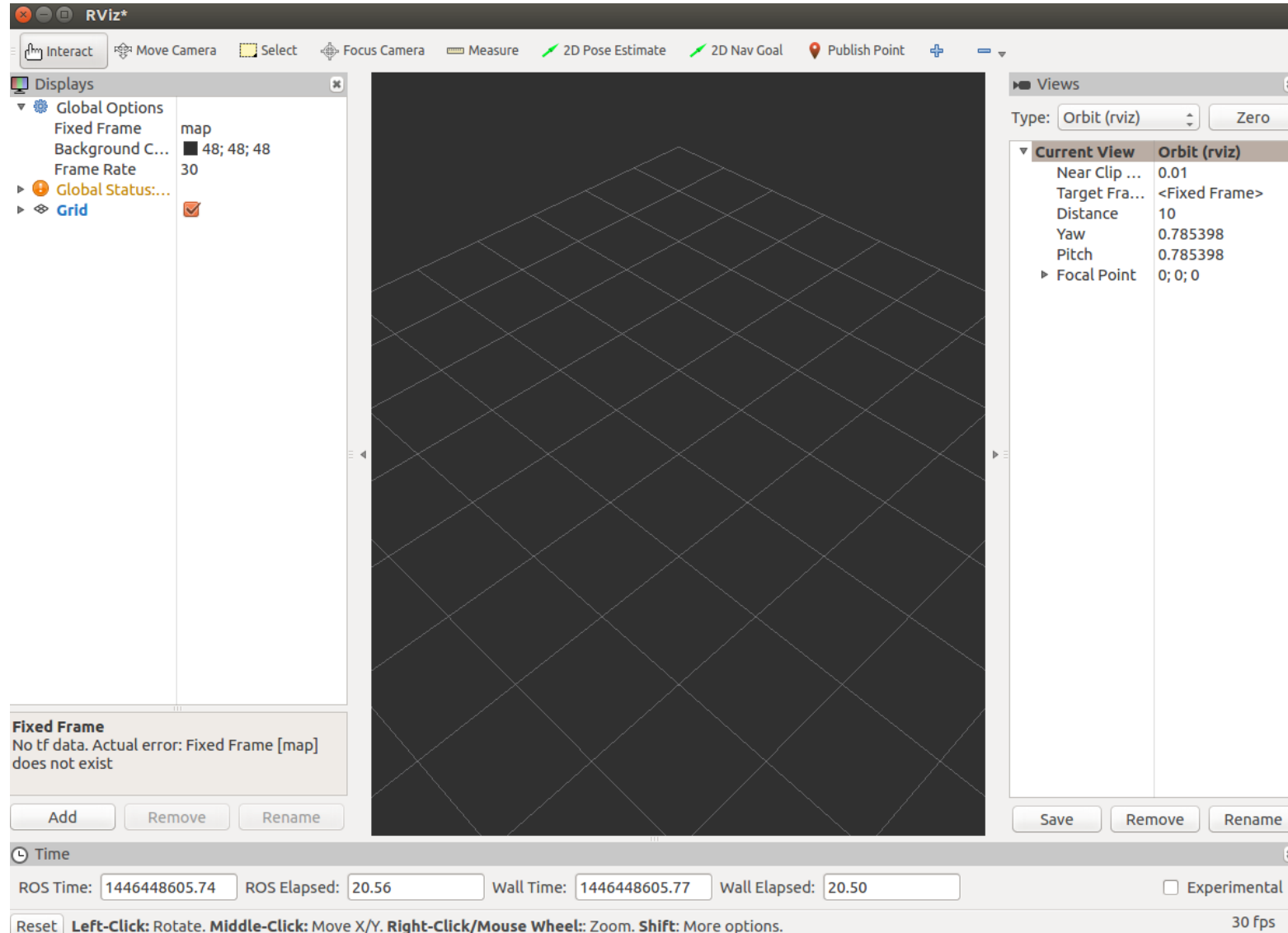
- RViz 실행

```
$ rosrun rviz rviz
```

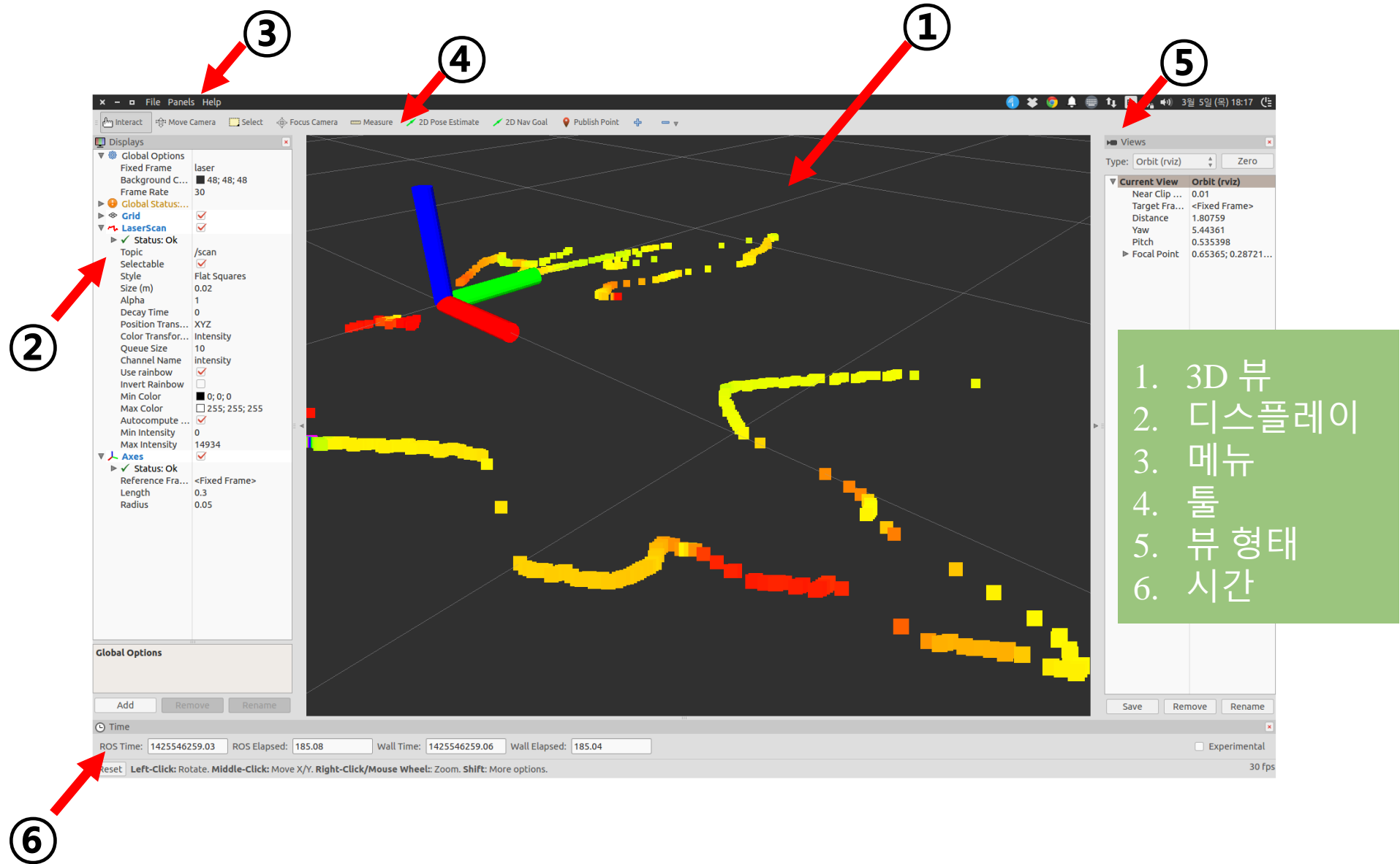
- 또는

```
$ rviz
```

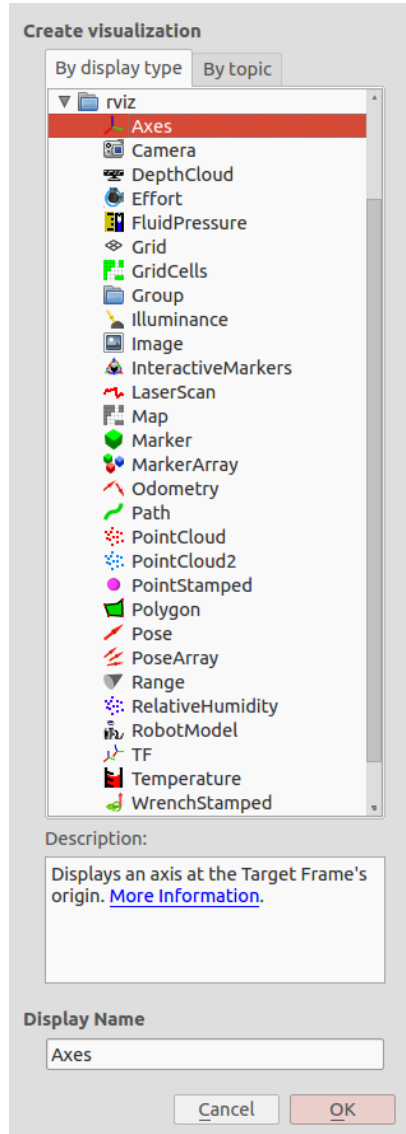
# RViz 초기 모습 (미설정)



# RViz 의 화면 구성 (LDS의 경우)



# RViz의 디스플레이 종류 (②의 'ADD'를 클릭)



	xyz 축		경로
	카메라 영상 오버레이		포인트 클라우드
	거리영상에 카메라 영상을 입힘		포인트 클라우드2
	회전 관절의 힘		점
	유체 압력		폴리곤
	그리드		포즈
	그리드 셀 (지도에 이용)		포즈 배열
	그룹		범위
	조도		상대 온도
	영상		로봇 모델
	인터랙티브 마커		좌표 변환 값 (TF)
	레이저 스캔		상대 습도
	지도		쥐어 돌림
	마커		
	마커 배열		
	오도메트리		

실습 시간

‘LRF, IMU, USB camera,  
Depth camera, Robot Model  
을 RViz를 통해 확인해보자.’

나눠 드린 센서들을

각자 PC의 RViz를 이용해 데이터를 확인해보세요.

# 오늘의 실습 준비물



# RViz 실습 #1 (LDS)

```
$ cs
$ git clone https://github.com/ROBOTIS-GIT/hls_lfcd_lds_driver.git
$ cd
$ sudo chmod a+rw /dev/ttyUSB0
$ roslaunch hls_lfcd_lds_driver view_hlds_laser.launch
```

(LDS의 경우)

```
$ cs
$ git clone https://github.com/robopeak/rplidar_ros.git
$ cd
$ sudo chmod a+rw /dev/ttyUSB0
$ roslaunch rplidar_ros rplidar.launch
```

(RPLiDAR의 경우)

```
$ sudo apt-get install ros-kinetic-urg-node
$ sudo chmod a+rw /dev/ttyACM0
$ rosrn urg_node urg_node
```

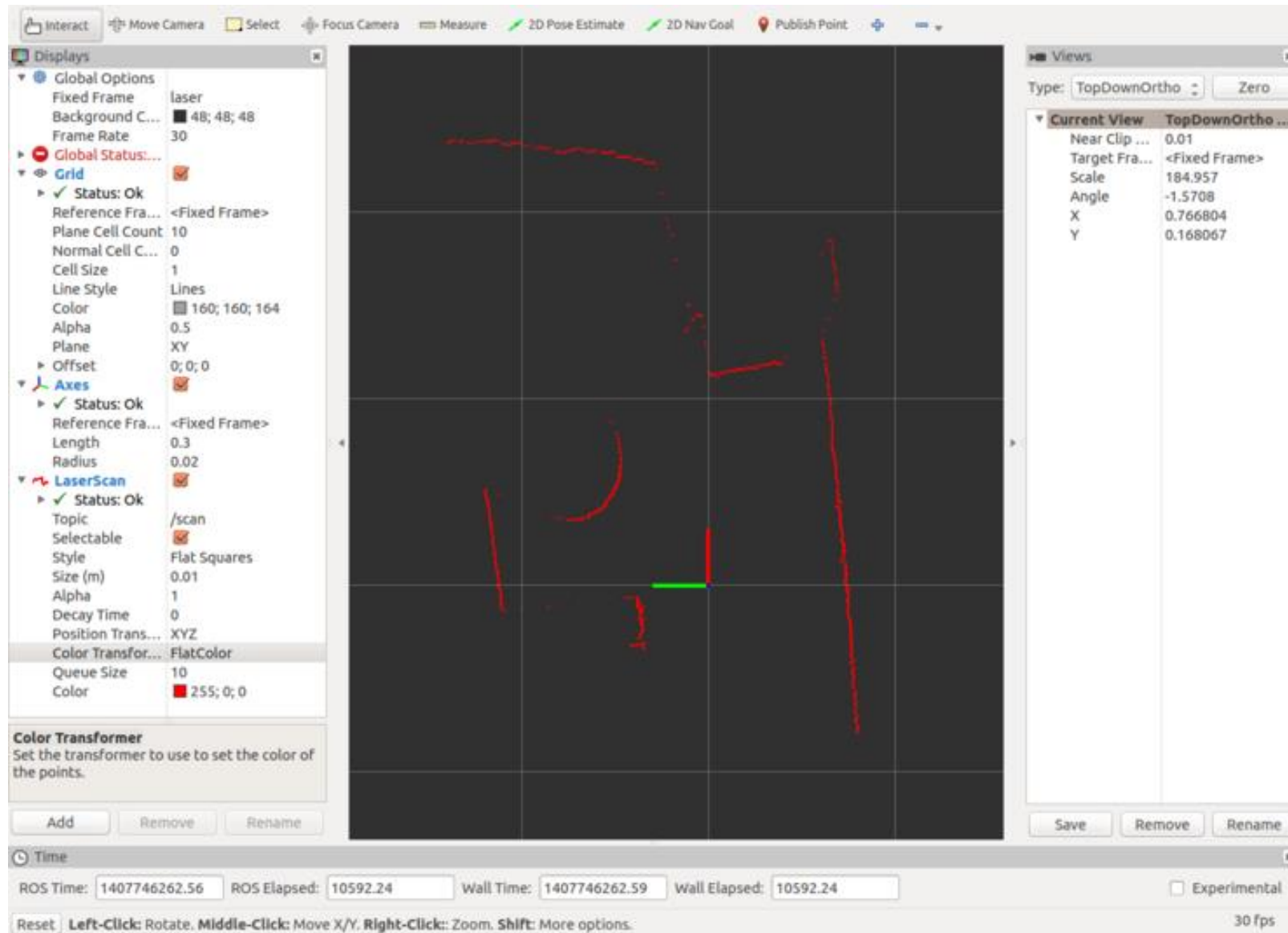
(HOKUYO의 경우)

\* RViz의 Displays 옵션 변경

- 1) Fixed Frame 변경: Global Options > **Fixed Frame = laser**
- 2) Axes 추가 및 설정: rviz 좌측 하단의 Add 클릭한 후, **Axes** 선택하여 추가한다. (Length 및 Radius 변경은 옵션)
- 3) LaserScan 추가 및 설정: rviz 좌측 하단의 Add 클릭한 후, **LaserScan** 선택하여 추가한다.  
(**Topic** 지정은 필수, Color Transformer, Color 등은 옵션)



# RViz 실습 #1 (LDS)

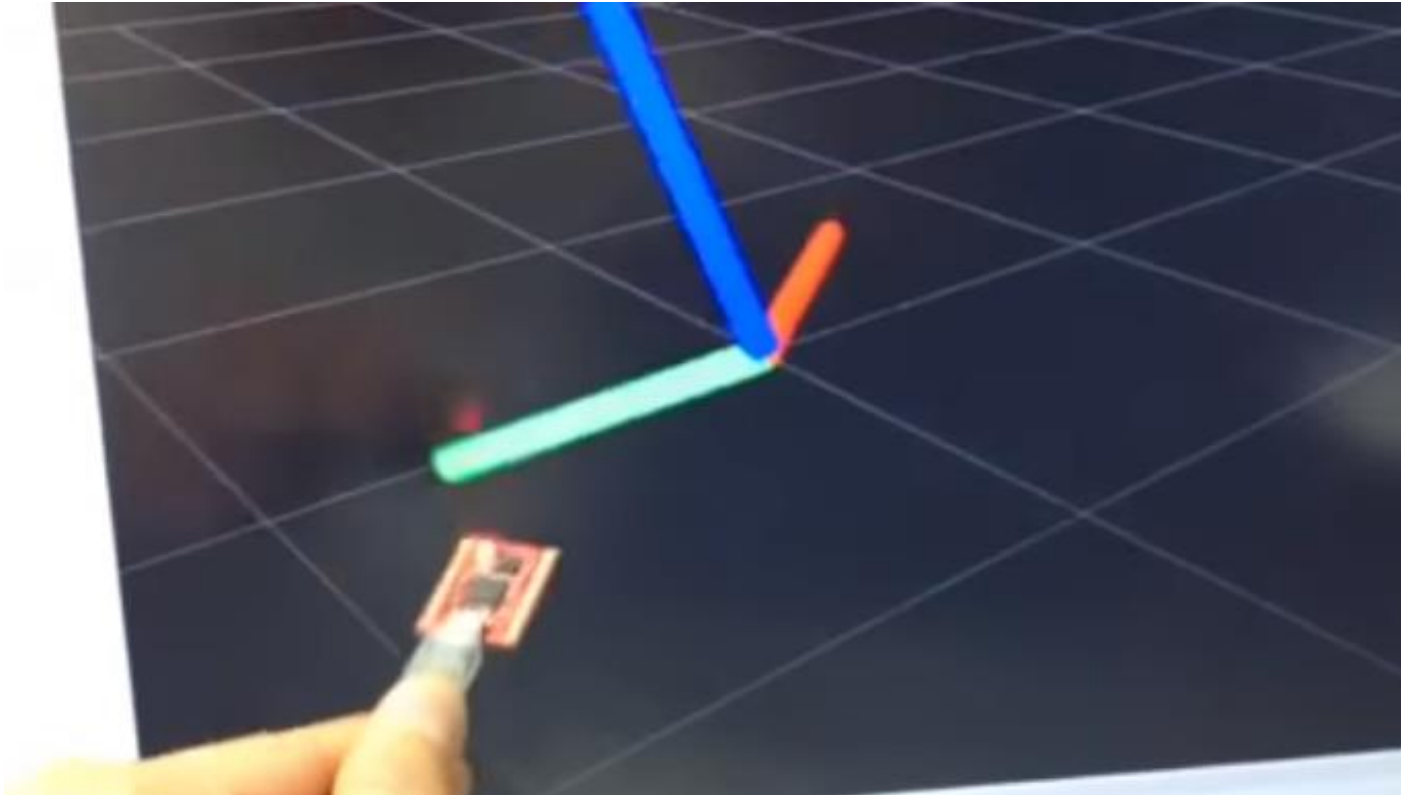




# RViz 실습 #2 (IMU)

```
$ cs  
$ git clone https://github.com/robotpilot/myahrs_driver.git  
$ cd  
$ sudo chmod a+rw /dev/ttyACM0  
$ roslaunch myahrs_driver myahrs_driver.launch
```

(withrobot사의 myAHRS+)



# RViz 실습 #3 (USB Camera)

```
$ sudo apt-get install ros-kinetic-udev-camera
```

```
$ roslaunch uvc_camera uvc_camera_node _device:=/dev/video?
```

```
$ rqt_image_view
```

카메라가 2대 이상일 경우,  
물음표 대신 사용하기 원하는  
디바이스 번호를 입력 (특히, 노트북의 경우)

\* RViz의 Displays 옵션 변경

1) Fixed Frame 변경

Global Options > Fixed Frame = camera

2) 이미지 디스플레이 추가

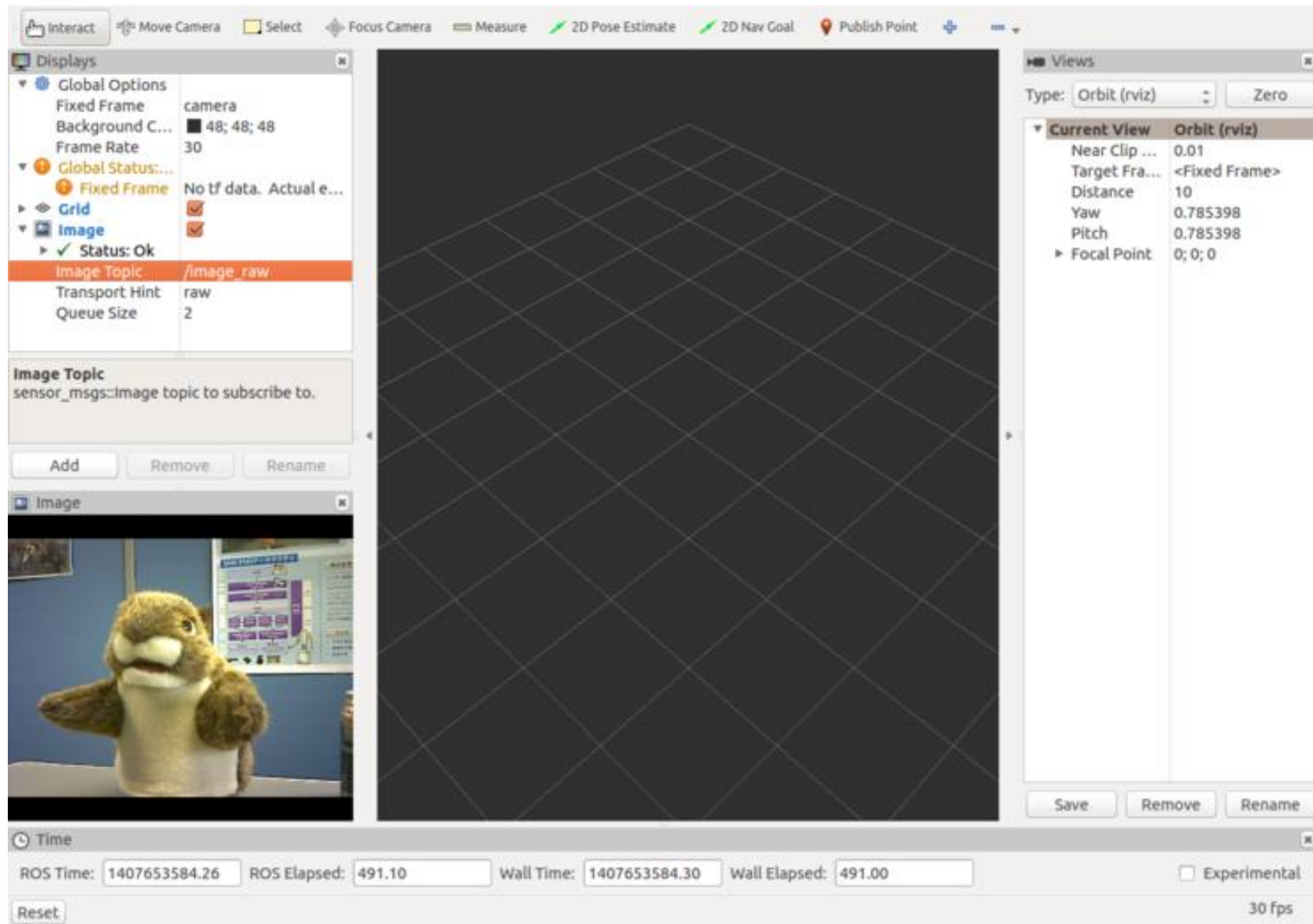
rviz 좌측 하단의 Add 클릭한 후, Image 선택하여 추가한다.

(Add > by display > rviz > Image)

3) 토픽 값 변경

Image > Image Topic 의 값을 "/image\_raw" 로 변경한다.

# RViz 실습 #3 (USB Camera)



# RViz 실습 #4 (Depth Camera)

```
$ sudo apt-get install ros-kinetic-openni2-camera ros-kinetic-openni2-launch (ASUS사의 Xtion의 경우)  
$ tar -xvf Sensor-Bin-Linux-x64-v5.1.0.41.tar.bz2 (*Xtion 구매시 CD안에 있음 또는 http://cafe.naver.com/openrt/6070)  
$ cd Sensor-Bin-Linux-x64-v5.1.0.41/  
$ sudo sh install.sh  
$ roslaunch oppeni2_launch oppeni2.launch
```

```
$ sudo apt-get install ros-kinetic-astra-camera ros-kinetic-astra-launch (ASTRA의 경우)  
$ wget https://raw.githubusercontent.com/tfoote/ros_astra_camera/master/orbbec-usb.rules  
$ wget https://raw.githubusercontent.com/tfoote/ros_astra_camera/master/install.sh  
$ sudo ./install.sh  
$ roslaunch astra_launch astra.launch
```

\* RViz의 Displays 옵션 변경

1) Fixed Frame 변경

Global Options > Fixed Frame 을 "[camera\\_depth\\_frame](#)" 로 변경한다.

2) PointCloud2 추가 및 설정

rviz 좌측 하단의 Add 클릭한 후, [PointCloud2](#)를 선택하여 추가한다.

3) Topic 이름 및 세부 설정 변경

# RViz 실습 #4 (Depth Camera)

(RealSense의 경우)

```
$ sudo apt-get install ros-kinetic-librealsense ros-kinetic-realsense-camera  
$ roslaunch realsense_camera r200_nodelet_default.launch  
$ rosrn rviz rviz -d rviz/realsenseRvizConfiguration1.rviz
```

\* RViz의 Displays 옵션 변경

1) Fixed Frame 변경

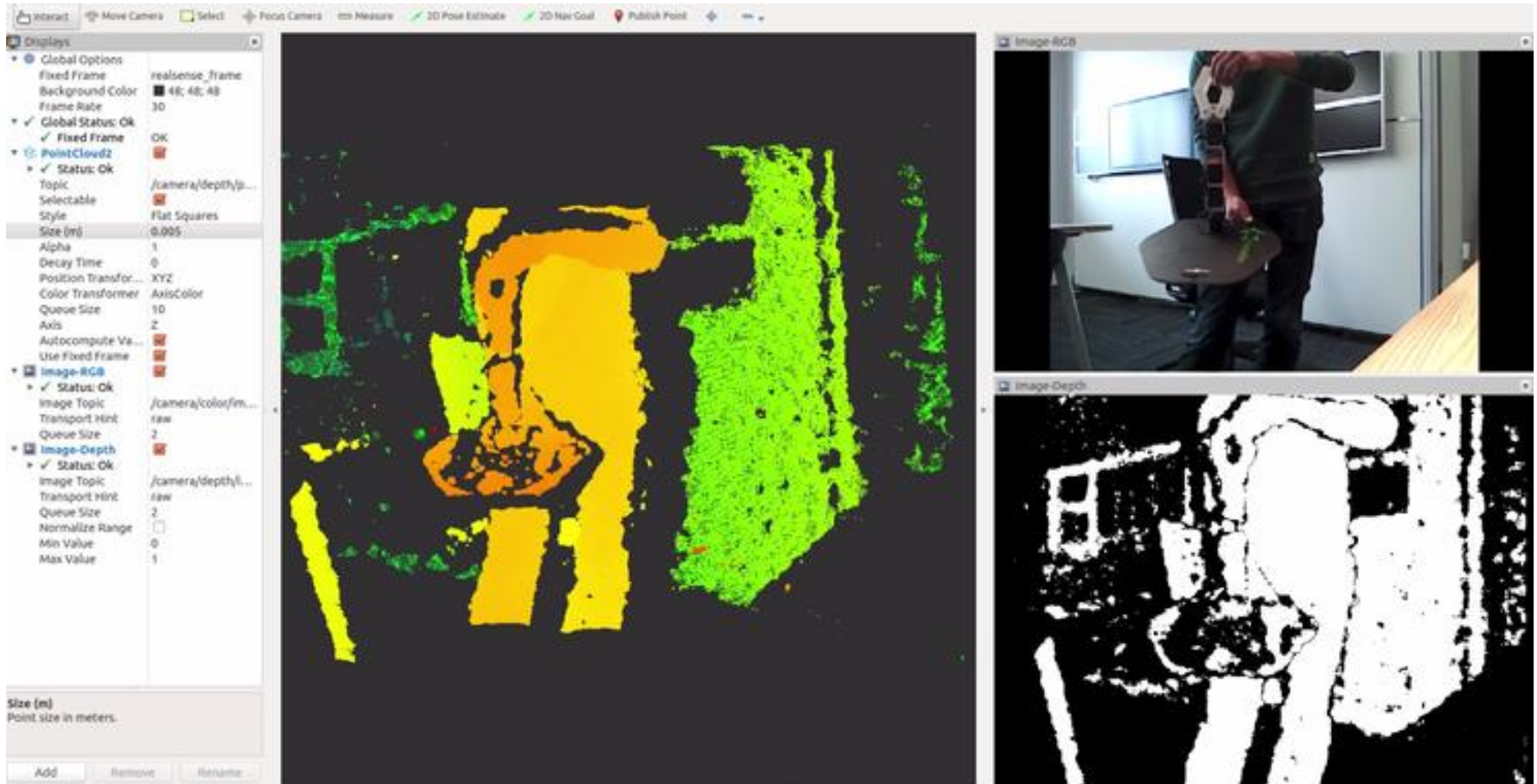
Global Options > Fixed Frame 을 "[camera\\_depth\\_frame](#)" 로 변경한다.

2) PointCloud2 추가 및 설정

rviz 좌측 하단의 Add 클릭한 후, [PointCloud2](#)를 선택하여 추가한다.

3) Topic 이름 및 세부 설정 변경

# RViz 실습 #4 (Depth Camera)

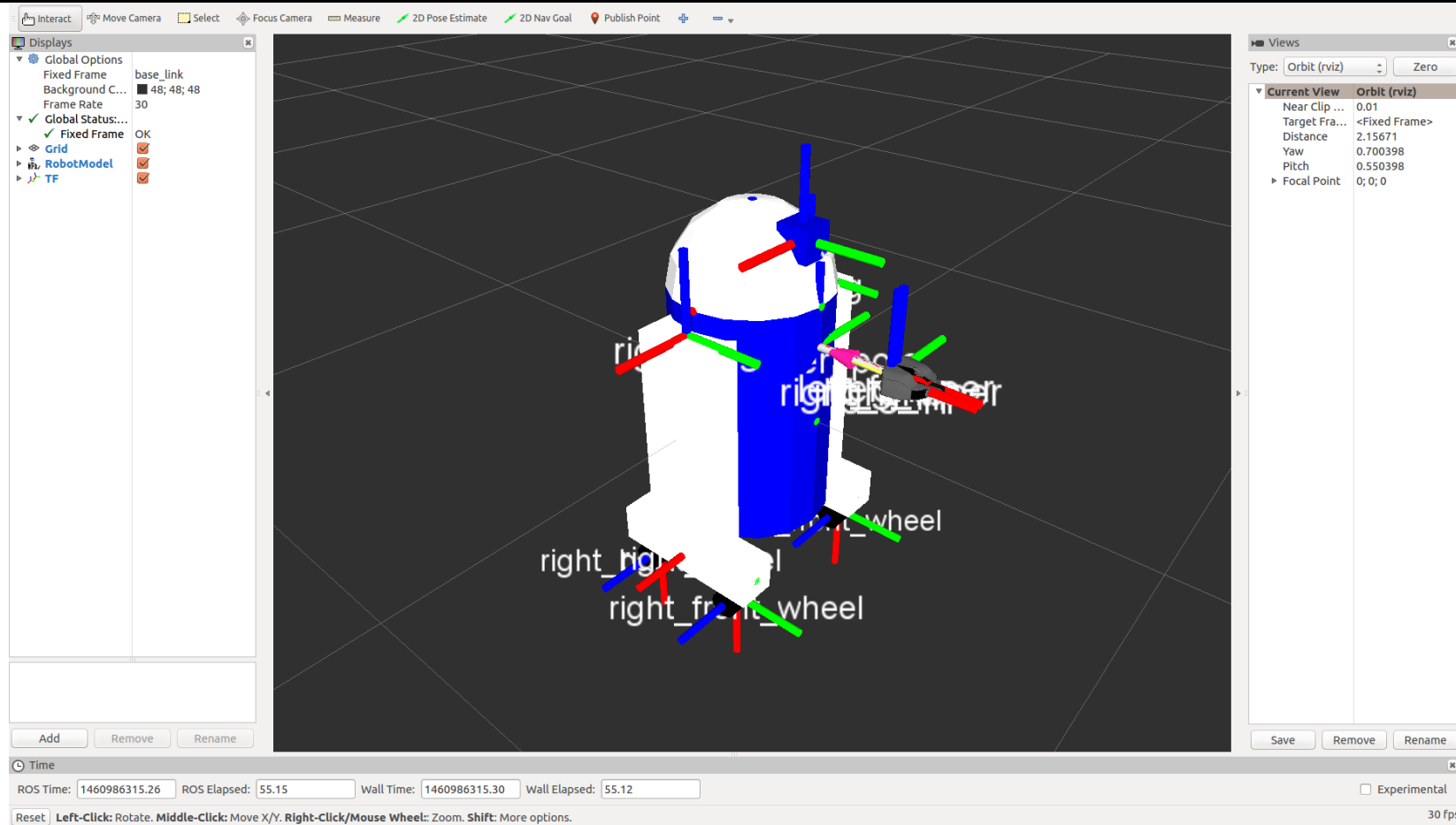


# RViz 실습 #5 (Robot Model)

- R2-D2 모델

```
$ sudo apt-get install ros-kinetic-urdf-tutorial
```

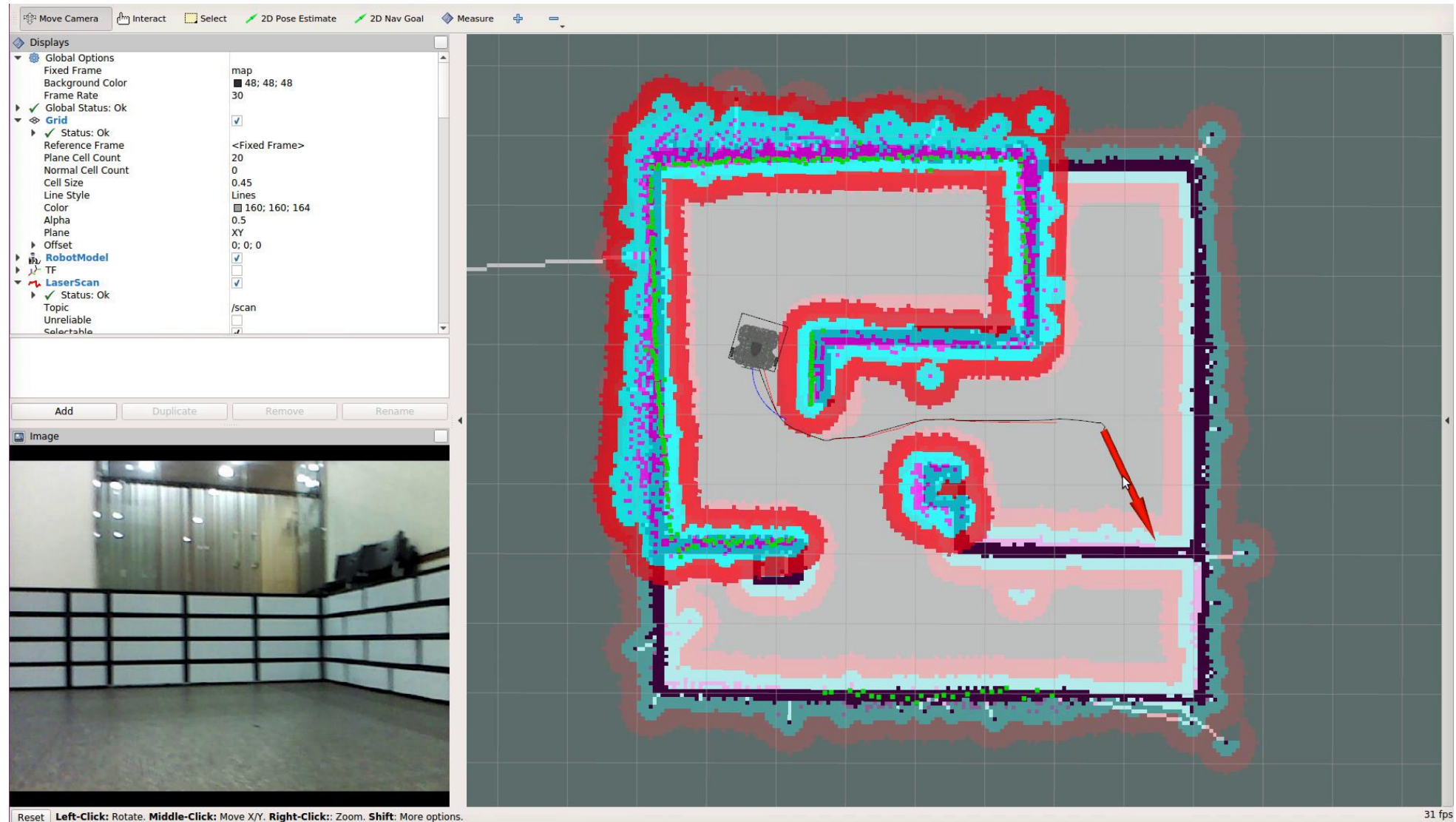
```
$ roslaunch urdf_tutorial display.launch model:='$(find urdf_tutorial)'urdf/05-visual.urdf
```





# RViz 실습 #6 (Navigation)

- 네비게이션 강의에서 진행





# RViz 실습 #7 (Interactive Marker)

---

- 매니퓰레이션 강의에서 진행

Rviz를 이용하면  
센서 및 로봇 관련  
데이터 시각화가 매우 간단!

# Index

---

**I. Command-Line Tools**

**II. Visualization Tool: Rviz**

**III. GUI Tool Box: RQT**

**IV. 3D Simulator: Gazebo**

# RQT: 플러그인 방식의 ROS의 종합 GUI 툴

---

- ROS Fuerte 버전부터는 rqt 라는 이름으로 기존의 rxbag, rxplot, rxgraph 등이 통합되어 rqt\_bag, rqt\_plot, rqt\_graph 등을 플러그인으로 하는 **ROS의 종합 GUI 툴**로써 사용 가능해졌다.
- rqt는 Qt로 개발되어 있기 때문에 유저들이 자유롭게 **플러그인**을 개발하여 추가할 수도 있다.
- rqt의 대표적인 플러그인인 **rqt\_image\_view, rqt\_graph, rqt\_plot, rqt\_bag**에 대해서 알아보도록 하자.
- 참고로, 그 이외에도
- rqt\_action, rqt\_gui, rqt\_plot, rqt\_runtime\_monitor, rqt\_bag, rqt\_gui\_cpp, rqt\_pose\_view, rqt\_rviz, rqt\_bag\_plugins, rqt\_gui\_py, rqt\_publisher, rqt\_service\_caller, rqt\_capabilities, rqt\_image\_view, rqt\_py\_common, rqt\_shell, rqt\_console, rqt\_launch, rqt\_py\_console, rqt\_srv, rqt\_controller\_manager, rqt\_logger\_level, rqt\_reconfigure, rqt\_tf\_tree, rqt\_dep, rqt\_moveit, rqt\_robot\_dashboard, rqt\_top, rqt\_ez\_publisher, rqt\_msg, rqt\_robot\_monitor, rqt\_topic, rqt\_graph, rqt\_nav\_view, rqt\_robot\_steering, rqt\_web
- 등의 플러그인이 존재한다. (헐 ——;;)

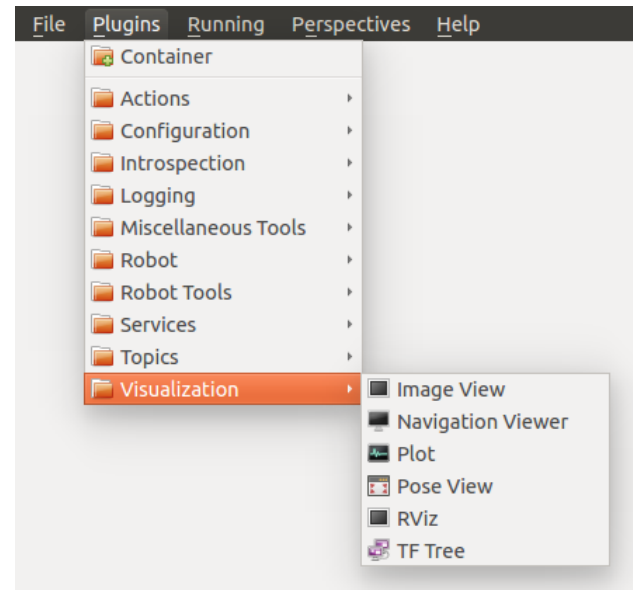
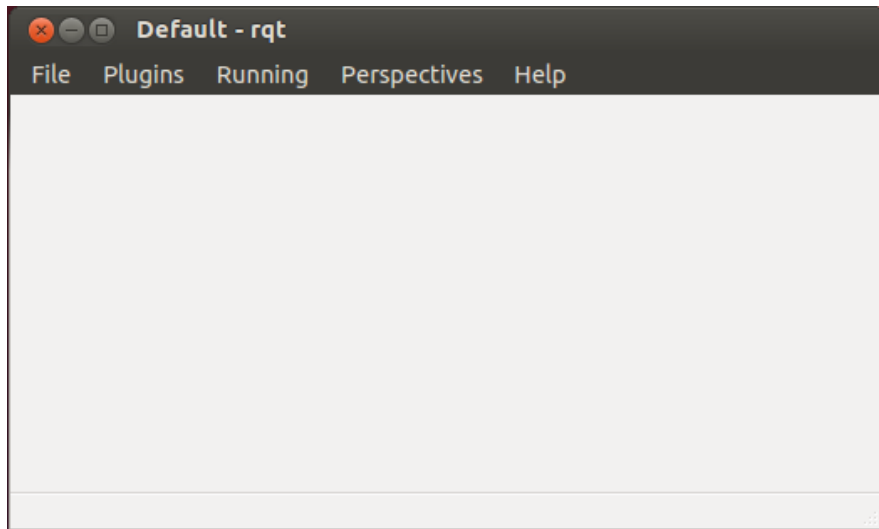
# RQT 설치 및 실행

- RQT 설치

```
$ sudo apt-get install ros-kinetic-rqt ros-kinetic-rqt-common-plugins
```

- RQT 실행

```
$ rqt
```



# RQT 플러그인 #1

---

## 1. 액션 (Action)

- **Action Type Browser** | Action 타입의 데이터 구조를 확인

## 2. 구성 (Configuration)

- **Dynamic Reconfigure** | 노드들에서 제공하는 설정값 변경을 위한 GUI 설정값 변경
- **Launch** | roslaunch 의 GUI 버전

## 3. 내성 (Introspection)

- **Node Graph** | 구동중인 노드들의 관계도 및 메시지의 흐름을 확인 가능한 그래프 뷰
- **Package Graph** | 노드의 의존 관계를 표시하는 그래프 뷰
- **Process Monitor** | 실행중인 노드들의 CPU사용률, 메모리사용률, 스레드수 등을 확인

## 4. 로깅 (Logging)

- **Bag** | ROS 데이터 로깅
- **Console** | 노드들에서 발생하는 경고(Warning), 에러(Error) 등의 메시지를 확인
- **Logger Level** | ROS의 Debug, Info, Warn, Error, Fatal 로거 정보를 선택하여 표시

# RQT 플러그인 #2

---

## 5. 다양한 툴 (Miscellaneous Tools)

- Python Console | 파이썬 콘솔 화면
- Shell | 셸(shell)을 구동
- Web | 웹 브라우저를 구동

## 6. 로봇 (Robot)

- 사용하는 로봇에 따라 계기판(dashboard) 등의 플러그인을 이곳에 추가

## 7. 로봇툴 (Robot Tools)

- Controller Manager | 컨트롤러 제어에 필요한 플러그인
- Diagnostic Viewer | 로봇 디바이스 및 에러 확인
- Moveit! Monitor | 로봇 팔 계획에 사용되는 Moveit! 데이터를 확인
- Robot Steering | 로봇 조정 GUI 툴, 원격 조정에서 이 GUI 툴을 이용하여 로봇 조종
- Runtime Monitor | 실시간으로 노드들에서 발생하는 에러 및 경고를 확인

# RQT 플러그인 #3

---

## 8. 서비스 (Services)

- **Service Caller** | 구동중인 서비스 서버에 접속하여 서비스를 요청
- **Service Type Browser** | 서비스 타입의 데이터 구조를 확인

## 9. 토픽 (Topics)

- **Easy Message Publisher** | 토픽을 GUI 환경에서 발행
- **Topic Publisher** | 토픽을 생성하여 발행
- **Topic Type Browser** | 토픽 타입의 데이터 구조 확인
- **Topic Monitor** | 사용자가 선택한 토픽의 정보를 확인

## 10. 시각화 (Visualization)

- **Image View** | 카메라의 영상 데이터를 확인
- **Navigation Viewer** | 로봇 네비게이션의 위치 및 목표지점 확인
- **Plot** | 2차원 데이터 플롯 GUI 플러그인, 2차원 데이터의 도식화
- **Pose View** | 현재 TF의 위치 및 모델의 위치 표시
- **RViz** | 3차원 시각화 툴인 RViz 플러그인
- **TF Tree** | tf 관계를 트리로 나타내는 그래프 뷰



# RQT의 사용 예시

The screenshot displays the RQT interface with several panels:

- Web Panel:** Shows the ROS.org website with the "Documentation" tab selected.
- Publisher Panel:** Displays a table of topics being published.
- Robot Steering Panel:** Features a vertical slider for controlling a robot's velocity.
- Logger Level Panel:** Shows a list of loggers and their configured levels.
- Console Panel:** Displays a log of messages from the ROS system.
- Plot Panel:** Shows a graph of two sine waves,  $\cos(i/20)*20$  (red) and  $\sin(i/20)*10$  (blue).

topic	type	rate	enabled	expression
/cmd_vel2	std_msgs/Float32	10.00	True	$\cos(i/20)*20$
data	float32			
/cmd_vel3	std_msgs/Float32	5.00	True	$\sin(i/20)*10$
data	float32			

Message	Severity	Node	Time
#9 Loading Setup Assistant Complete	Info	/moveit_setup_assistant	11:11:25.344 (2012-08-02)
#8 Listening to 'moveit_planning_scene'	Info	/moveit_setup_assistant	11:11:25.294 (2012-08-02)
#7 Starting scene monitor	Info	/moveit_setup_assistant	11:11:25.293 (2012-08-02)
#6 Configuring kinematics solvers	Info	/moveit_setup_assistant	11:11:25.107 (2012-08-02)
#4 Robot semantic model successfully loaded.	Info	/moveit_setup_assistant	11:11:23.119 (2012-08-02)
#5 Setting Param Server with Robot Seman...	Info	/moveit_setup_assistant	11:11:23.119 (2012-08-02)

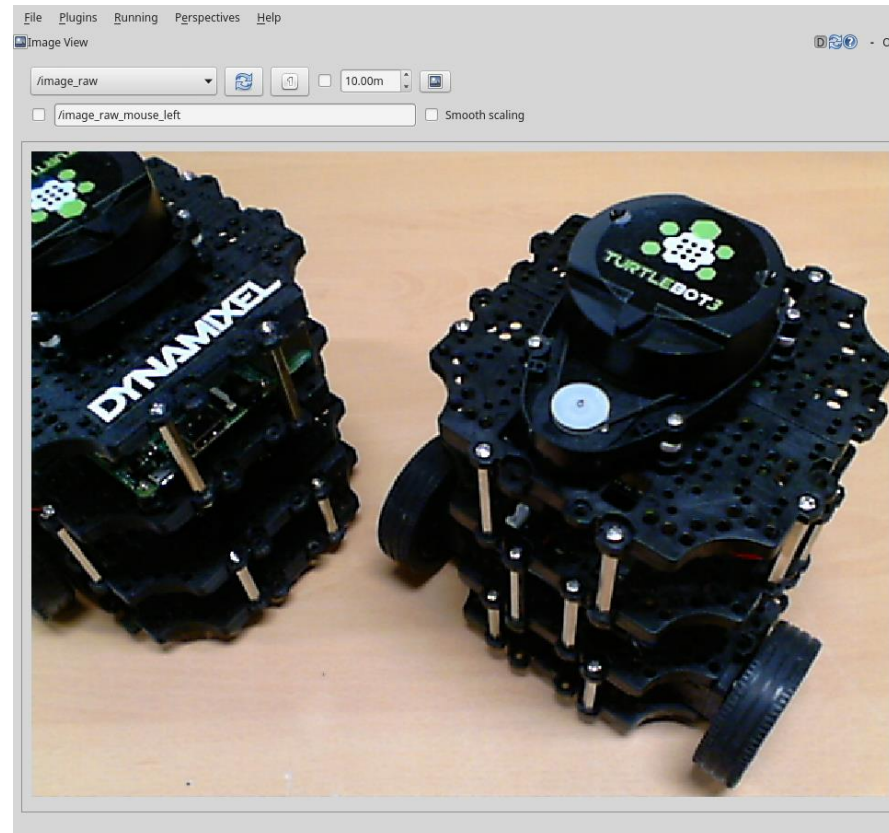
Nodes	Loggers	Levels
/rosout	ros	Debug
/rqt_gui_cpp	ros.moveit_c	Info
/rqt_gui_cpp	ros.roscpp	Warn
/rviz_134392	ros.roscpp.ro	Error
	ros.roscpp.su	Fatal

Plot:  $\cos(i/20)*20$  (red) and  $\sin(i/20)*10$  (blue)

# RQT 실습 #1: rqt\_image\_view

```
$ rosrun uvc_camera uvc_camera_node
```

\$ rqt (메뉴에서 [Plugins] → [Visualization] → [Image View] 를 선택한다.)  
또는  
\$ rqt\_image\_view



# RQT 실습 #2: rqt\_graph

```
$ rosrun turtlesim turtlesim_node
```

```
$ rosrun turtlesim turtle_teleop_key
```

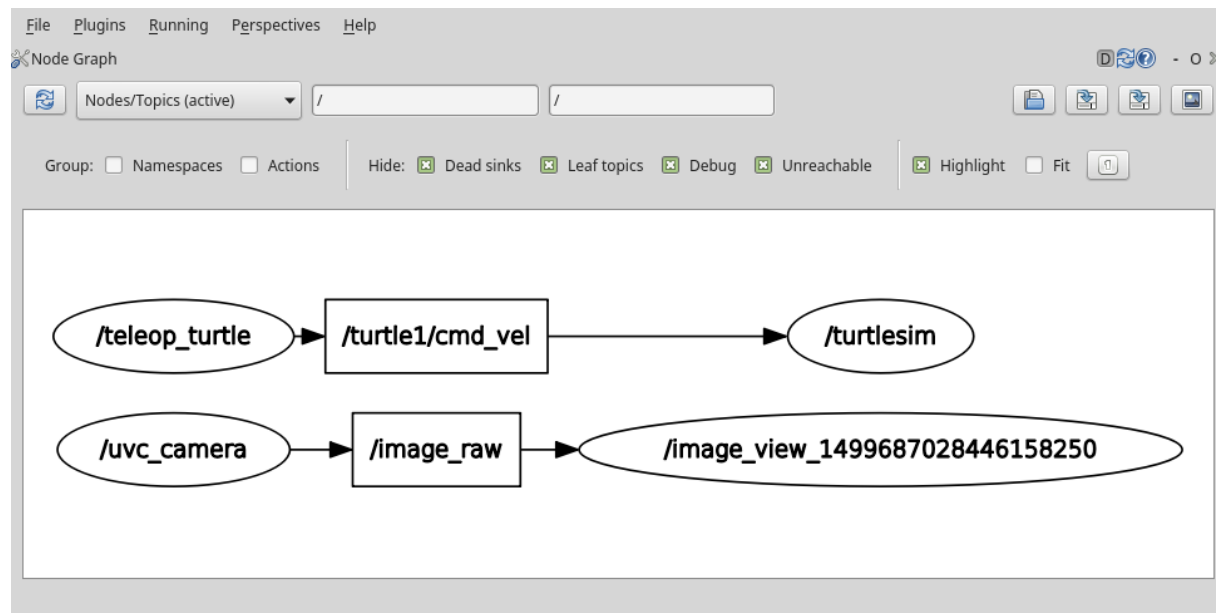
```
$ rosrun uvc_camera uvc_camera_node
```

```
$ rosrun image_view image_view image:=image_raw
```

\$ rqt (메뉴에서 [Plugins] → [Introspection] → [Node\_Graph] 를 선택한다.)

또는

```
$ rqt_graph
```



# RQT 실습 #3: rqt\_plot

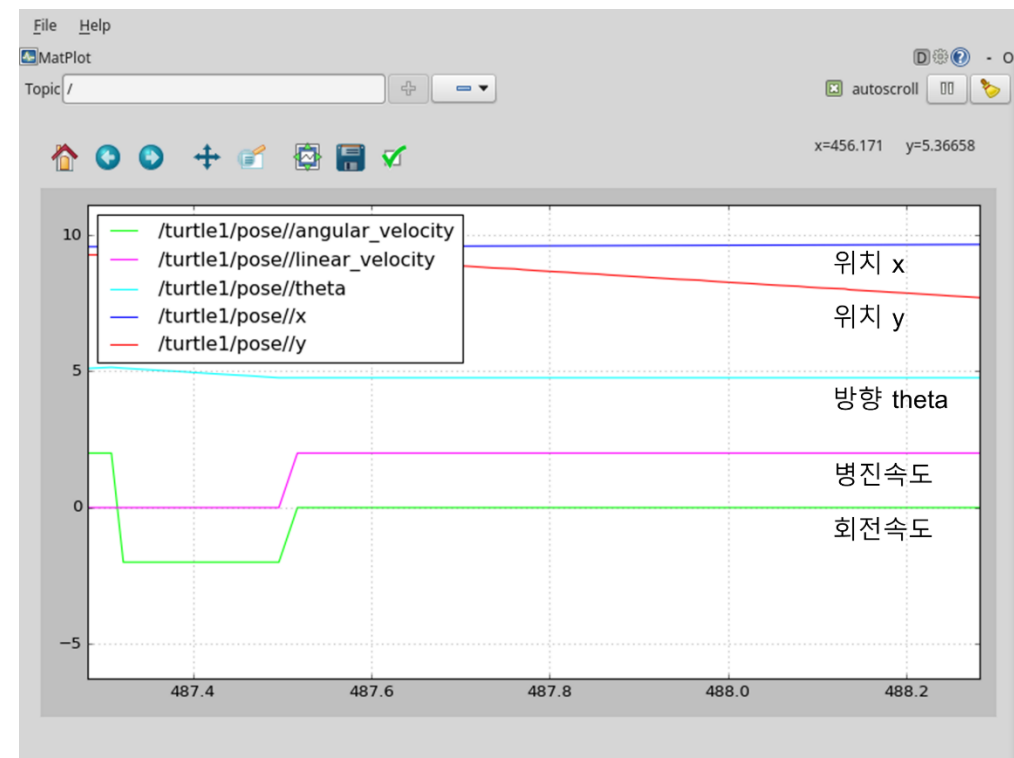
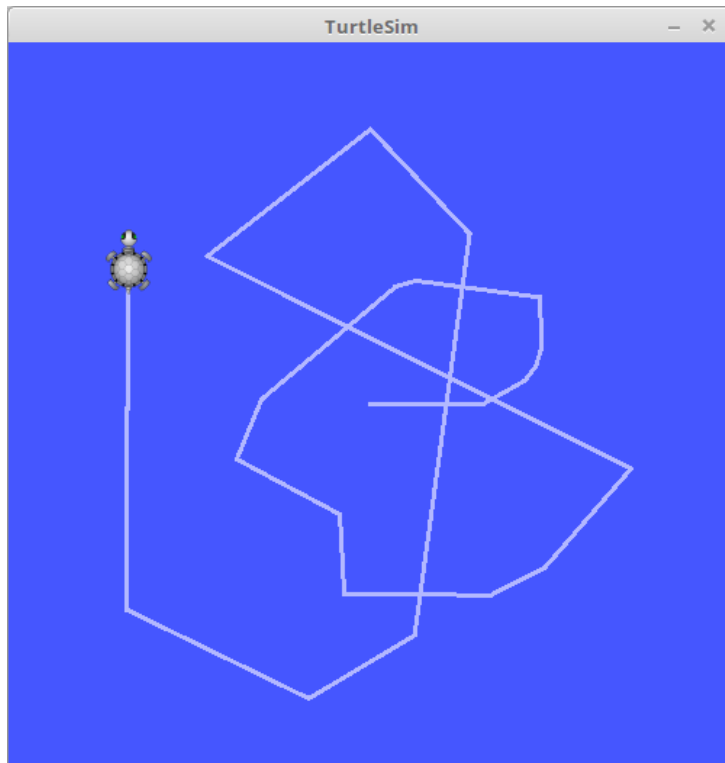
```
$ roslaunch turtlesim turtlesim_node
```

```
$ roslaunch turtlesim turtle_teleop_key
```

\$ rqt (메뉴에서 [Plugins] → [Visualization] → [Plot] 를 선택한다.)

또는

```
$ rqt_plot /turtle1/pose/
```



# RQT 실습 #4: rqt\_bag

```
$ roslaunch uvc_camera uvc_camera_node
```

```
$ rosbag record /image_raw
```

```
$ rqt (메뉴에서 [Plugins] → [Logging] → [Bag] 를 선택한다.)
```

또는

```
$ rqt_bag
```



# RQT를 이용하면

1. GUI 형태로 ROS 이용 가능
2. GUI Tool 제작이 간단!

# Index

---

**I. Command-Line Tools**

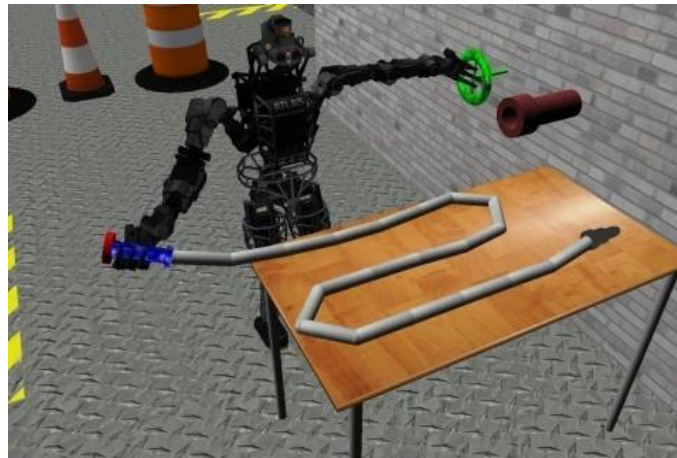
**II. Visualization Tool: Rviz**









**III. GUI Tool Box: RQT**

**IV. 3D Simulator: Gazebo**

# Gazebo

- Gazebo는 로봇 개발에 필요한 3차원 시뮬레이션을 위한 로봇, 센서, 환경 모델 등을 지원하고 **물리 엔진을 탑재**하여 실제와 근사한 결과를 얻을 수 있는 **3차원 시뮬레이터**이다.
- Gazebo는 최근에 나온 오픈 진영 시뮬레이터 중 가장 좋은 평가를 받고 있고, 미국 **DARPA Robotics Challenge**의 공식 시뮬레이터로 선정되어 개발에 더욱 박차를 가하고 있는 상황이다.
- ROS에서는 그 태생이 Player/Stage, Gazebo를 기본 시뮬레이터로 사용하고 있어서 **ROS와의 호완성도** 매우 좋다.

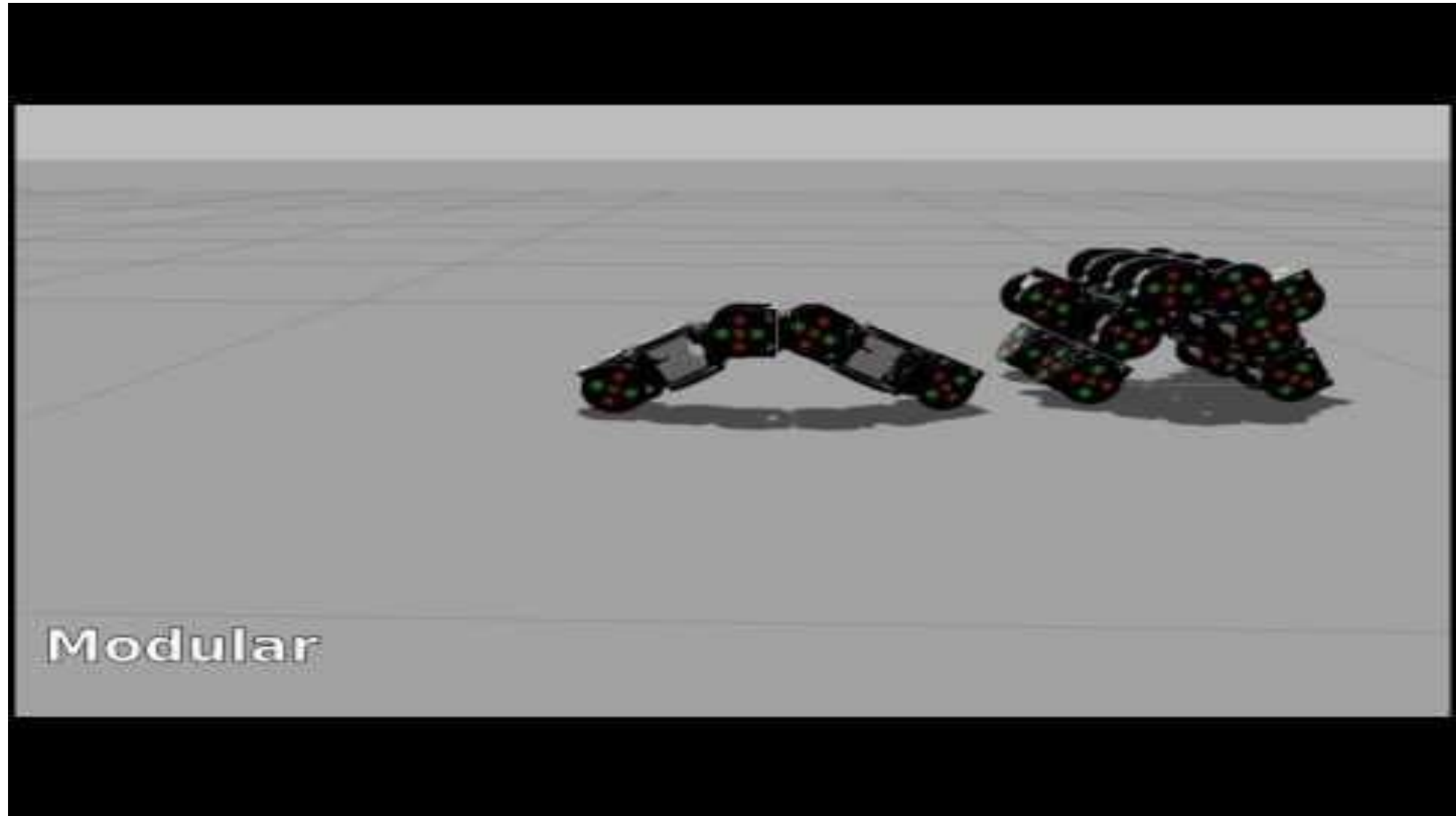


 <b>Dynamics Simulation</b> Access multiple high-performance physics engines including ODE, Bullet, Simbody, and DART.	 <b>Advanced 3D Graphics</b> Utilizing OGRE, Gazebo provides realistic rendering of environments including high-quality lighting, shadows, and textures.	 <b>Sensors and Noise</b> Generate sensor data, optionally with noise, from laser range finders, 2D/3D cameras, Kinect style sensors, contact sensors, force-torque, and more.	 <b>Plugins</b> Develop custom plugins for robot, sensor, and environmental control. Plugins provide direct access to Gazebo's API.
 <b>Robot Models</b> Many robots are provided including PR2, Pioneer2 DX, iRobot Create, and TurtleBot. Or build your own using SDF.	 <b>TCP/IP Transport</b> Run simulation on remote servers, and interface to Gazebo through socket-based message passing using Google Protobufs.	 <b>Cloud Simulation</b> Use CloudSim to run Gazebo on Amazon, Softlayer, or your own OpenStack instance.	 <b>Command Line Tools</b> Extensive command line tools facilitate simulation introspection and control.



# Gazebo

---



Key Point?

시뮬레이션

이 필요하다하면 ROS와 연동하기 쉬운

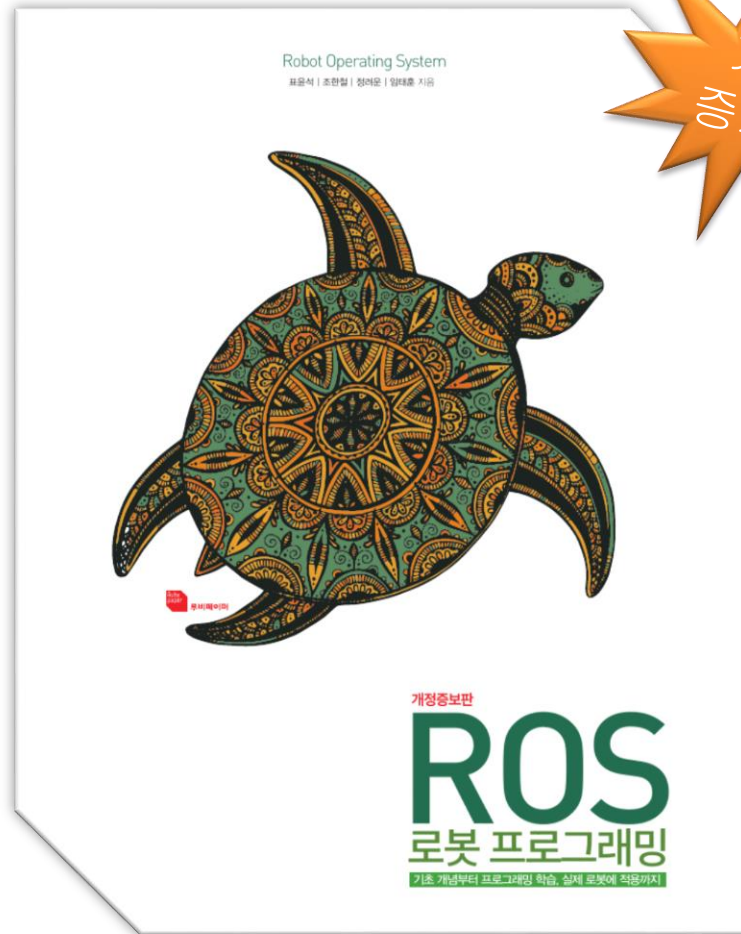
Gazebo 이용!~

---

**질문  
대환영!**

---

여기서! 광고 하나 나가요~



✓ Direct Link

국내 유일! 최초! ROS 참고서!  
ROS 공식 플랫폼 TurtleBot3 개발팀이  
직접 저술한 바이블급 ROS 책

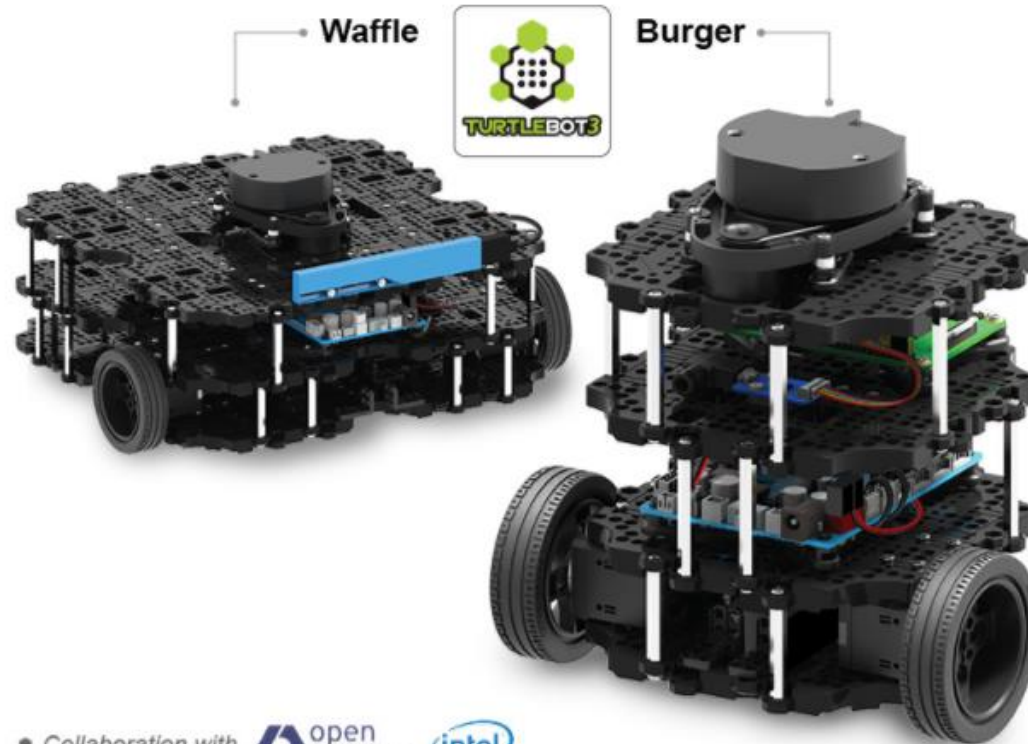
여기서! 광고 둘 나가요~

**TURTLEBOT3**

## 인공지능(AI) 연구의 시작, ROS 교육용 공식 로봇 플랫폼

터틀봇3는 ROS기반의 저가형 모바일 로봇으로  
교육, 연구, 제품개발, 취미 등 다양한 분야에서 활용 할 수 있습니다.

✓ Direct Link



• Collaboration with  

여기서! 광고 셋 나가요~



- 오로카
- [www.oroqa.org](http://www.oroqa.org)
- 오픈 로보틱스 지향
- 풀뿌리 로봇공학의 저변 활성화
- 공개 강좌, 세미나, 프로젝트 진행

- 로봇공학을 위한 열린 모임 (KOS-ROBOT)
- [www.facebook.com/groups/KoreanRobotics](https://www.facebook.com/groups/KoreanRobotics)
- 로봇공학 통합 커뮤니티 지향
- 일반인과 전문가가 어울러지는 한마당
- 로봇공학 소식 공유
- 연구자 간의 협력

혼자 하기에 답답하시다고요?  
커뮤니티에서 함께 해요~

# 끝.

표윤석

Yoonseok Pyo  
pyo@robotis.com  
www.robotpilot.net



[www.facebook.com/yonseok.pyo](https://www.facebook.com/yonseok.pyo)