8장. TELEOP-BOT

동의대학교 컴퓨터소프트웨어공학과
이종민 교수

목차

- 키보드 구동기
- 운동 생성기
- 매개변수 서버
- 속도 경사
- rviz

개발 패턴

- 작은 기능을 가지는 새로운 ROS 노드를 점증적으로 개발
- 레고 블록 맞추기: 요철 대신 토픽, 서비스, 액션으로 통신하는 노드를 이용
- 텔레옵-봇
  - 키보드 입력을 통하여 터틀봇의 이동을 제어
  - 구성 요소
    - key_publisher.py: 입력된 키 값을 발행
    - keys_to_twist.py: 키 입력 시 지정된 cmd_vel 한 번 발행
    - keys_to_twist_using_rate.py: 최근 입력된 키 값에 따라서 지속적으로 cmd_vel 발행
    - keys_to_twist_parameterized.py: linear_scale, angular_scale 매개변수를 이용하여 cmd_vel 크기 변경
    - key_to_twist_with_ramps.py: 점차적으로 속도 증감

key_publisher.py

```
1  #!/usr/bin/env python
2
3  import sys, select, tty, termios
4  import rospy
5  from std_msgs.msg import String
6
7  if __name__ == '__main__':
8      key_pub = rospy.Publisher('keys', String, queue_size=1)
9      rospy.init_node("keyboard_driver")
10     rate = rospy.Rate(100)
11     old_attr = termios.tcgetattr(sys.stdin)
12     tty.setcbreak(sys.stdin.fileno())
13     print "Publishing keystrokes. Press Ctrl-C to exit..."
14     while not rospy.is_shutdown():
15         if select.select([sys.stdin], [], [], 0)[0] == [sys.stdin]:
16             key_pub.publish(sys.stdin.read(1))
17         rate.sleep()
18     termios.tcsetattr(sys.stdin, termios.TCSADRAIN, old_attr)
```

std_msgs/String 자료형을 사용하는 keys 토픽 발행자 생성

표준 입력의 상태 저장

표준 입력의 모드를 cbreak로 설정

표준 입력을 통한 키보드 입력이 있으면

표준 입력을 1바이트만 읽어옴.

저장한 표준 입력의 모드로 복구

Python 함수 설명

- termios.tcgetattr(fd)
  - 파일 기술자 fd의 tty 속성 목록 반환
- termios.tcsetattr(fd, when, attributes)
  - attributes 인자 값을 사용하여 파일 기술자 fd의 tty 속성 설정
  - when: 언제 속성 변환할 지 지정. termios.TCSANOW는 즉시 반영, termios.TCSADRAIN은 큐에 있는 모든 출력이 전송된 후에 반영, termios.TCSAFLUSH는 큐에 있는 모든 출력을 전송하고 큐의 모든 입력을 폐기한 후에 반영.
- tty.setcbreak(fd, when=termios.TCSAFLUSH)
  - 파일 기술자 fd의 모드를 cbreak로 변경. 참고. tty.setraw(fd, when=termios.TCSAFLUSH)
- select.select(rlist, wlist, xlist[, timeout])
  - 유닉스 select() 시스템 콜 호출
  - rlist: 입력 목록
  - wlist: 출력 목록
  - xlist: 예외 조건 목록
  - timeout: 설정된 시간 동안 rlist, wlist, xlist에 있는 입출력 있는지 대기

select() 함수

- 유닉스/리눅스 명령창에서 man select 실행하면 함수 사용법 볼 수 있음.
- Synopsis

```
int select(int nfds, fd_set *readfds, fd_set *writefds,
                    fd_set *exceptfds, struct timeval *timeout);
```

- 여러 개의 파일 기술자를 모니터링할 수 있게 해주는 함수

실행 방법

Terminal 1:

$ roscore

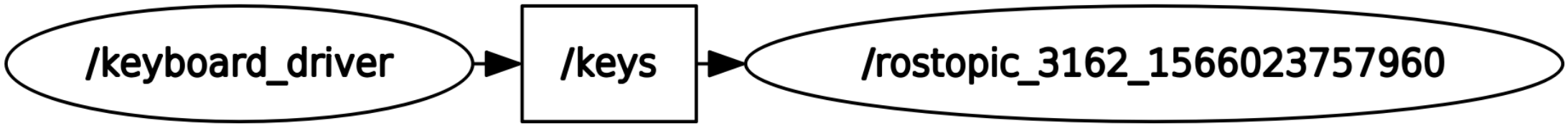
Terminal 2:

$ rosrun deu_ros key_publisher.py

Terminal 3:

$ rostopic echo keys

ROS 그래프: key_publisher.py



실행 결과

- Terminal 2:

```
jongmin@ubuntu:~$ rosrun deu_ros key_publisher.py
Publishing keystrokes. Press Ctrl-C to exit...
```

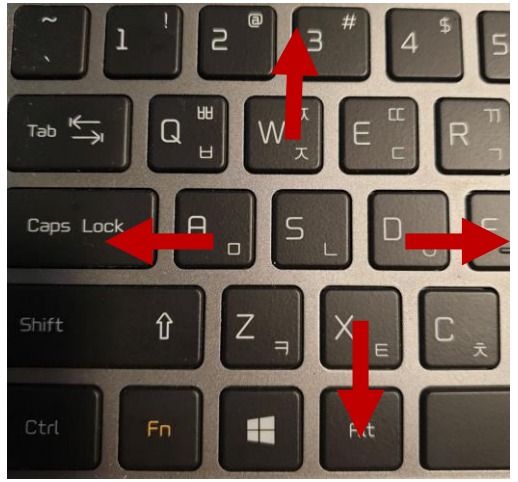
- Terminal 3:

```
^Cjongmin@ubuntu:~$ rostopic echo keys
data: "a"
---
data: "b"
---
data: "c"
---
data: "d"
---
data: "e"
---
```

keys_to_twist.py

```
1  #!/usr/bin/env python
2
3  import rospy
4  from geometry_msgs.msg import Twist
5  from std_msgs.msg import String
6
7  key_mapping = {'w': [0, 1], 'x': [0, -1],
8                 'a': [-1, 0], 'd': [1, 0],
9                 's': [0, 0]}
10
11
12 def keys_cb(msg, twist_pub):
13     if len(msg.data) == 0 or not key_mapping.has_key(msg.data[0]):
14         return  # unknown key.
15     vels = key_mapping[msg.data[0]]
16     t = Twist()
17     t.angular.z = vels[0]
18     t.linear.x = vels[1]
19     twist_pub.publish(t)
```

키 입력을 [각속도, 선속도]로 변환하기 위한 dict 객체



키보드 입력이 없거나 a, d, s, w, x가 아닌 경우 그냥 반환

유효 키 입력에 대한 [각속도, 선속도] 찾기

Twist 객체 생성

각속도 설정

선속도 설정

Twist 객체 발행

```
22 if __name__ == '__main__':
23     rospy.init_node('keys_to_twist')
24     twist_pub = rospy.Publisher('cmd_vel', Twist, queue_size=1)
25     rospy.Subscriber('keys', String, keys_cb, twist_pub)
26     rospy.spin()
```

cmd_vel 토픽 발행자 객체 생성
keys 토픽 구독자 객체 생성

→ keys_cb 함수의 인자

실행 방법

Terminal 1:

$ roscore

Terminal 2:

$ rosrun deu_ros key_publisher.py

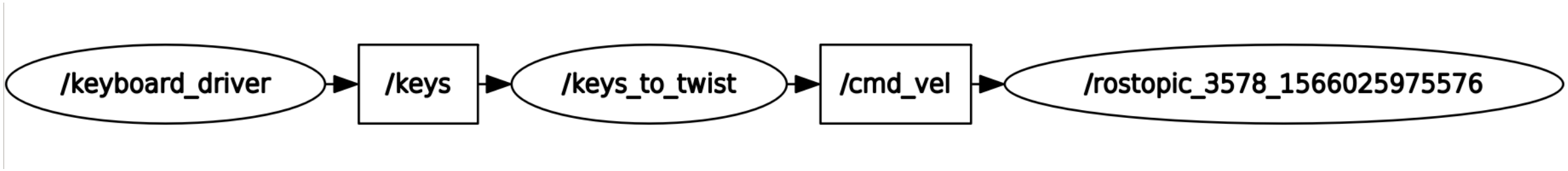
Terminal 3:

$ rosrun deu_ros keys_to_twist.py

Terminal 4:

$ rostopic echo cmd_vel

ROS 그래프: keys_to_twist.py



실행 결과

- Terminal 4:

```
jongmin@ubuntu:~$ rostopic echo cmd_vel
linear:
  x: 1.0
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0
---
linear:
  x: -1.0
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0
---
```

keys_to_twist_using_rate.py

```
1  #!/usr/bin/env python
2
3  import rospy
4  from geometry_msgs.msg import Twist
5  from std_msgs.msg import String
6
7  key_mapping = {'w': [0, 1], 'x': [0, -1],
8                 'a': [-1, 0], 'd': [1, 0],
9                 's': [0, 0]}
10 g_last_twist = None
11
12
13 def keys_cb(msg, pub):
14     global g_last_twist
15     if len(msg.data) == 0 or not key_mapping.has_key(msg.data[0]):
16         return  # unknown key.
17     vels = key_mapping[msg.data[0]]
18     g_last_twist.angular.z = vels[0]
19     g_last_twist.linear.x = vels[1]
20     pub.publish(g_last_twist)
```

최근 입력 키 값에 따른 Twist 객체

```
23 if __name__ == '__main__':
24     rospy.init_node('keys_to_twist_using_rate')
25     twist_pub = rospy.Publisher('cmd_vel', Twist, queue_size=1)
26     rospy.Subscriber('keys', String, keys_cb, twist_pub)
27     rate = rospy.Rate(10)
28     g_last_twist = Twist()  # initializes to zero
29     while not rospy.is_shutdown():
30         twist_pub.publish(g_last_twist)
31         rate.sleep()
```

실행 주기를 10Hz로 설정

Twist 객체 생성

g_last_twist 값을 주기적으로 발행

실행 주기에 맞게 휴면(대기)

실행 방법

Terminal 1:
$ roscore

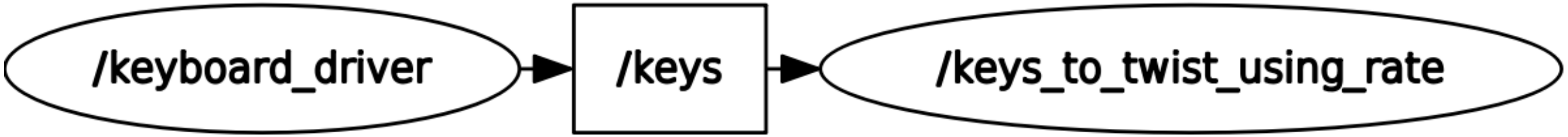
Terminal 2:
$ rosrun deu_ros key_publisher.py

Terminal 3:
$ rosrun deu_ros keys_to_twist_using_rate.py

Terminal 4:
$ rostopic echo cmd_vel

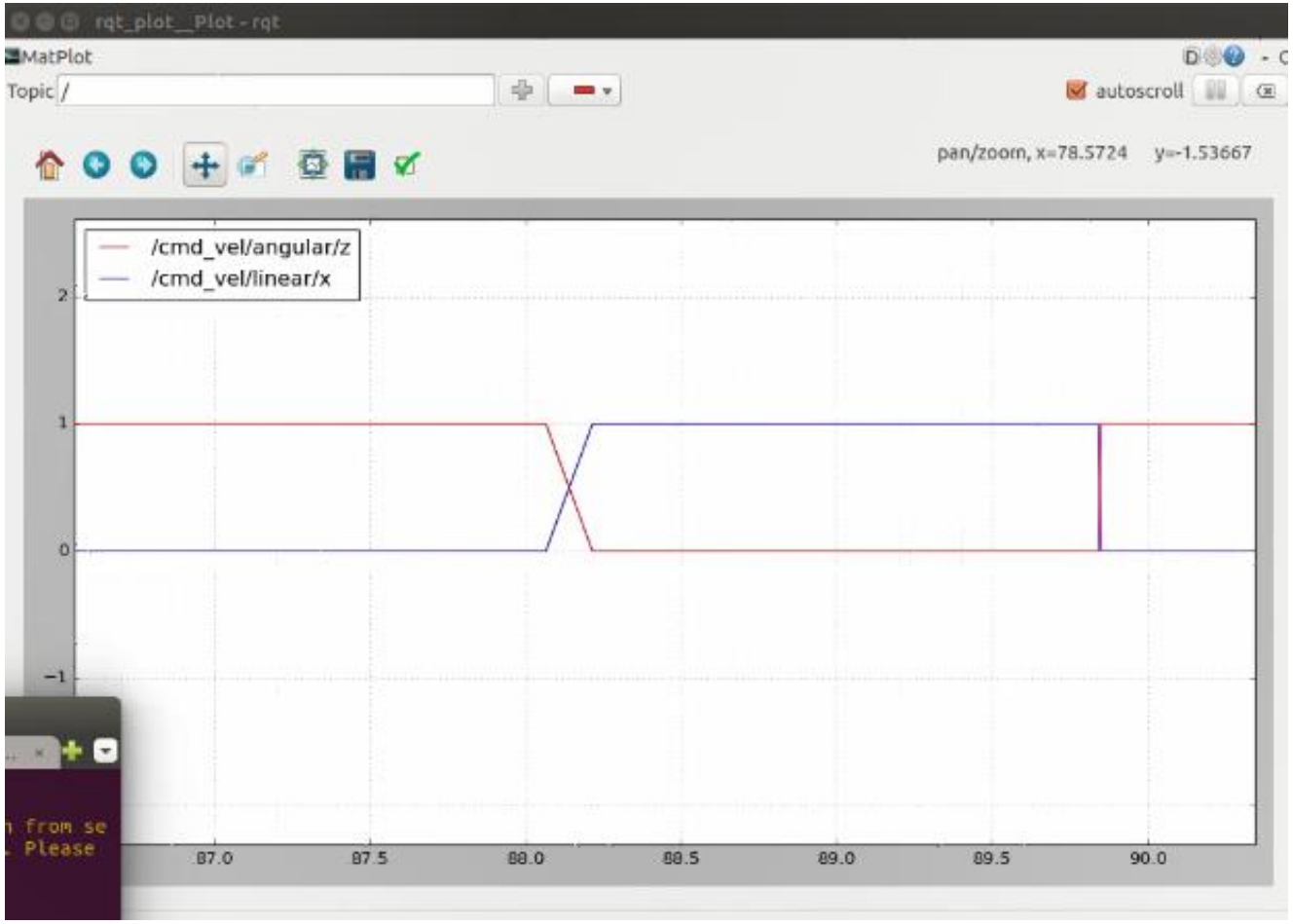
Terminal 5:
$ **rqt_plot** /cmd_vel/linear/x /cmd_vel/angular/z

ROS 그래프: keys_to_twist_using_rate.py

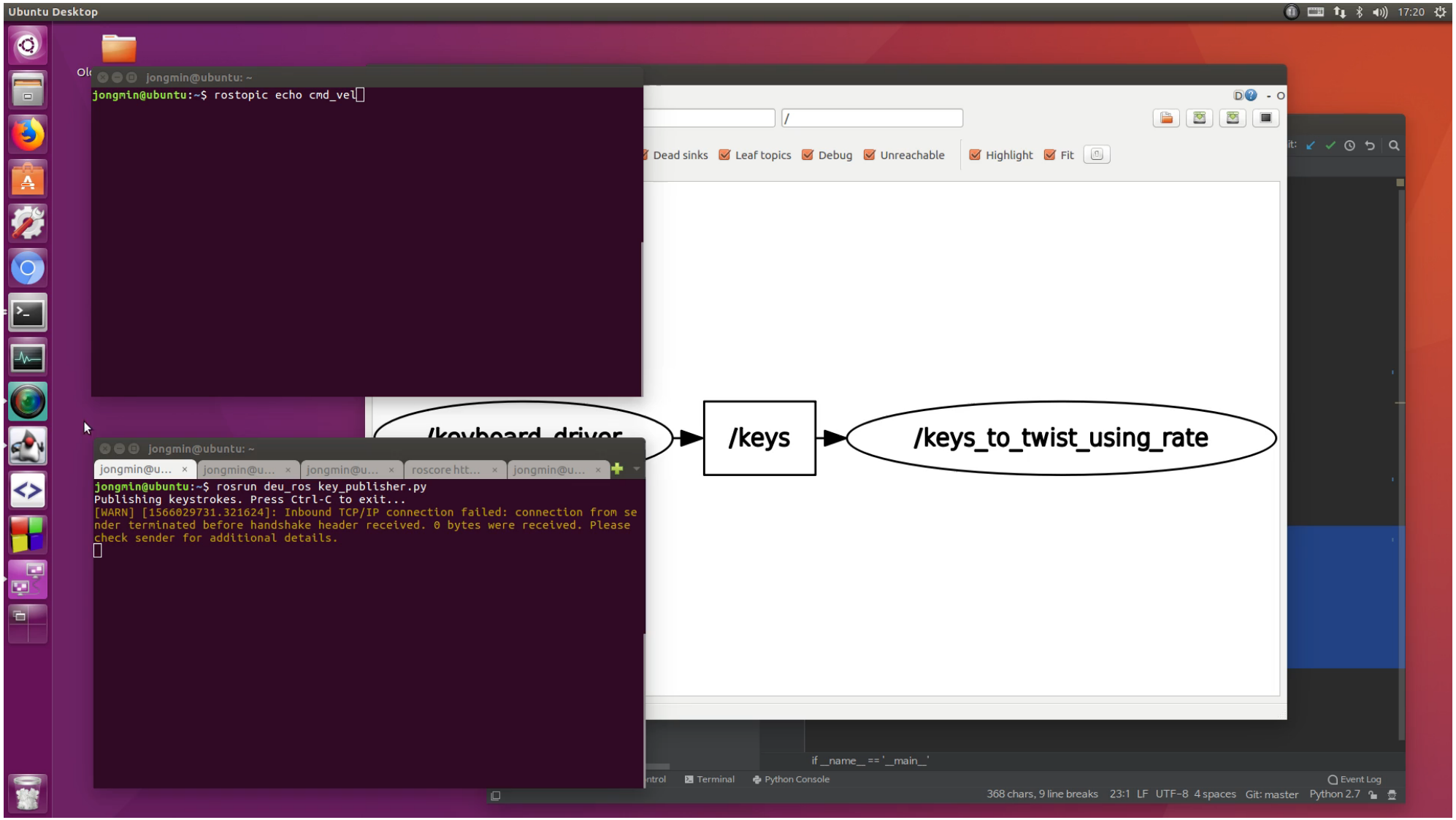


실행 결과

- rqt_plot 캡쳐 화면



rqt_plot


Ubuntu Desktop
jongmin@ubuntu: ~
jongmin@ubuntu:~$ rostopic echo cmd_vel
Dead sinks
Leaf topics
Debug
Unreachable
Highlight
Fit
/keyboard_driver
/keys
/keys_to_twist_using_rate
jongmin@ubuntu: ~
jongmin@ubuntu:~$ rosrun deu_ros key_publisher.py
Publishing keystrokes. Press Ctrl-C to exit...
[WARN] [1566029731.321624]: Inbound TCP/IP connection failed: connection from sender terminated before handshake header received. 0 bytes were received. Please check sender for additional details.
if __name__ == '__main__'
Terminal
Python Console
Event Log
368 chars, 9 line breaks 23:1 LF UTF-8 4 spaces Git: master Python 2.7

keys_to_twist_parameterized.py

```
1  #!/usr/bin/env python
2
3  import rospy
4  from geometry_msgs.msg import Twist
5  from std_msgs.msg import String
6
7  key_mapping = {'w': [0, 1], 'x': [0, -1],
8                 'a': [-1, 0], 'd': [1, 0],
9                 's': [0, 0]}
10 g_last_twist = None
11 g_vel_scales = [0.1, 0.1]  # default to very slow
12
13
14 def keys_cb(msg, pub):
15     global g_last_twist, g_vel_scales
16     if len(msg.data) == 0 or not key_mapping.has_key(msg.data[0]):
17         return  # unknown key.
18     vels = key_mapping[msg.data[0]]
19     g_last_twist.angular.z = vels[0] * g_vel_scales[0]
20     g_last_twist.linear.x = vels[1] * g_vel_scales[1]
21     pub.publish(g_last_twist)
```

각속도, 선속도 비율

각속도에 각속도 비율 곱함.
선속도에 선속도 비율 곱함.

```
24 if __name__ == '__main__':
25     rospy.init_node('keys_to_twist')
26     twist_pub = rospy.Publisher('cmd_vel', Twist, queue_size=1)
27     rospy.Subscriber('keys', String, keys_cb, twist_pub)
28     g_last_twist = Twist()  # initializes to zero
29     if rospy.has_param('~linear_scale'):
30         g_vel_scales[1] = rospy.get_param('~linear_scale')
31     else:
32         rospy.logwarn("linear scale not provided; using %.1f" % ₩
33                       g_vel_scales[1])
34
35     if rospy.has_param('~angular_scale'):
36         g_vel_scales[0] = rospy.get_param('~angular_scale')
37     else:
38         rospy.logwarn("angular scale not provided; using %.1f" % ₩
39                       g_vel_scales[0])
40
41     rate = rospy.Rate(10)
42     while not rospy.is_shutdown():
43         twist_pub.publish(g_last_twist)
44         rate.sleep()
```

linear_scale 매개변수 있는지 확인하여 있으면
linear_scale 매개변수 값을 가져옴.
참고) '~'는 비공개(private)을 의미. 현재 노드 이름 공간에 속함.

angular_scale 매개변수 있는지 확인하여 있으면
angular_scale 매개변수 값을 가져옴.

실행 방법

Terminal 1:
$ roscore

Terminal 2:
$ rosrun deu_ros key_publisher.py

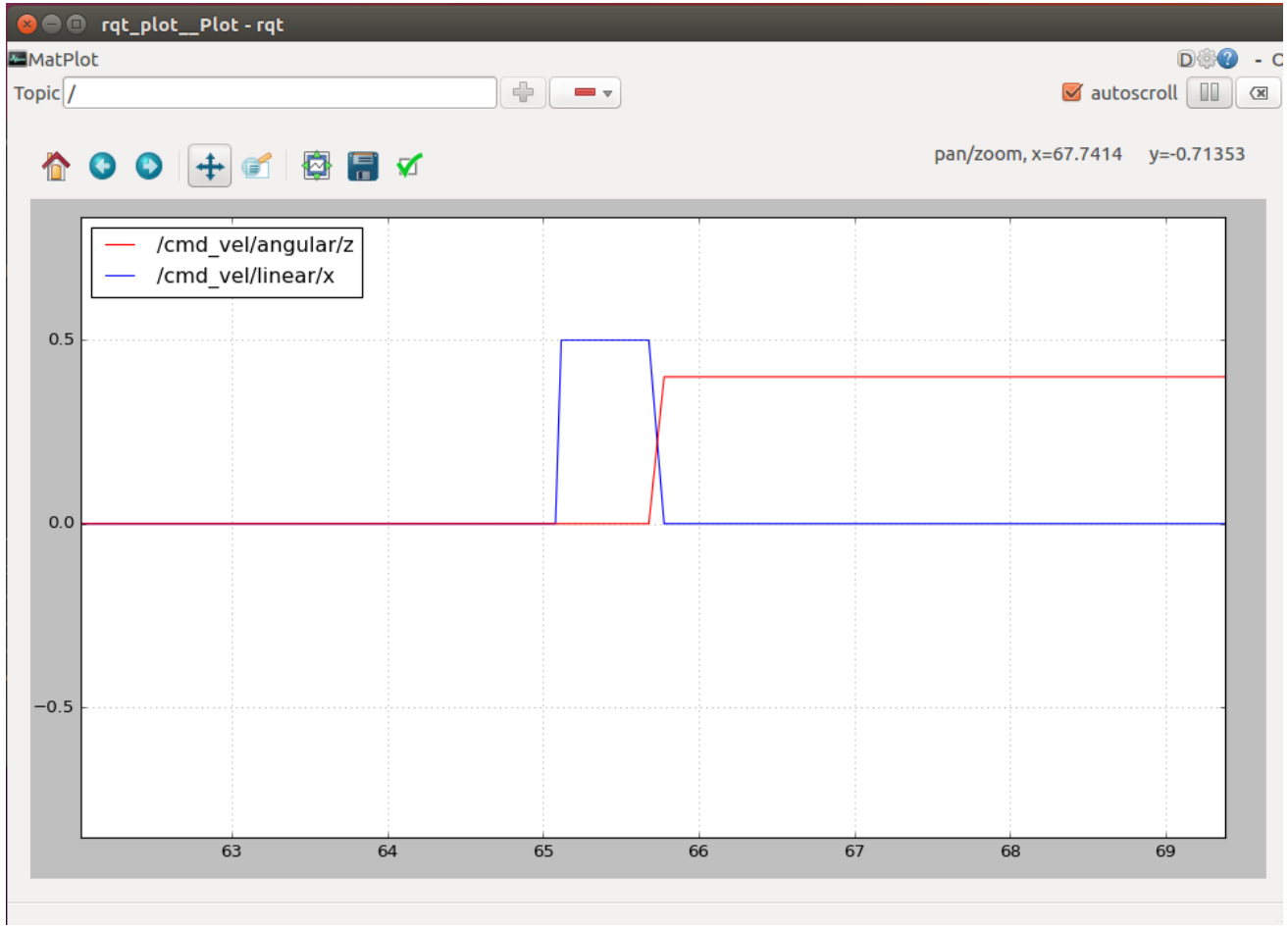
Terminal 3:
$ rosrun deu_ros keys_to_twist_parameterized.py **_linear_scale:=0.5 _angular_scale:=0.4**

Terminal 4:
$ rostopic echo cmd_vel

Terminal 5:
$ rqt_plot /cmd_vel/linear/x /cmd_vel/angular/z

실행 결과

- rqt_plot 캡쳐 화면



keys_to_twist_with_ramps.py

```
1  #!/usr/bin/env python
2
3  import rospy
4  from geometry_msgs.msg import Twist
5  from std_msgs.msg import String
6  import math
7
8  key_mapping = {'w': [0, 1], 'x': [0, -1],
9                 'a': [1, 0], 'd': [-1, 0],
10                's': [0, 0]}
11 g_twist_pub = None
12 g_target_twist = None
13 g_last_twist = None
14 g_last_send_time = None
15 g_vel_scales = [0.1, 0.1]  # default to very slow
16 g_vel_ramps = [1, 1]  # units: meters per second^2
```

g_target_twist, g_last_twist: 보정 안 된 Twist 객체

g_vel_scales: 각속도, 선속도 비율
g_vel_ramps: 각속도와 선속도에 대한 가속도

```
19 def ramped_vel(v_prev, v_target, t_prev, t_now, ramp_rate):
20     # print('v_prev =', v_prev)
21     # print('v_target =', v_target)
22     # compute maximum velocity step
23     step = ramp_rate * (t_now - t_prev).to_sec()
24     sign = 1.0 if (v_target > v_prev) else -1.0
25     error = math.fabs(v_target - v_prev)
26     if error < step:  # we
27         return v_target
28     else:
29         return v_prev + sign * step
30
31
32 def ramped_twist(prev, target, t_prev, t_now, ramps):
33     tw = Twist()
34     tw.angular.z = ramped_vel(prev.angular.z, target.angular.z, t_prev,
35                               t_now, ramps[0])
36     tw.linear.x = ramped_vel(prev.linear.x, target.linear.x, t_prev,
37                              t_now, ramps[1])
38     return tw
```

v_prev: 직전 속도, v_target: 최종 속도,
t_prev: 직전 시각, t_now: 현재 시각, ramp_rate: 가속도

step: 이동 속도 (m/sec^2 * sec → m/sec)

v_prev보다 v_target이 크면 +1, 아니면 -1

오차(error) 범위 안이면 v_target을 반환

이전 속도 v_prev에 sign * step을 더하여 반환

Twist 객체 생성

가속도를 고려한 각속도 계산

가속도를 고려한 선속도 계산

Twist 객체 반환

```
41  def send_twist():
42      global g_last_twist_send_time, g_target_twist, g_last_twist, ₩
43          g_vel_scales, g_vel_ramps, g_twist_pub
44      t_now = rospy.Time.now()
45      g_last_twist = ramped_twist(g_last_twist, g_target_twist,
46                                  g_last_twist_send_time, t_now, g_vel_ramps)
47      g_last_twist_send_time = t_now
48      g_twist_pub.publish(g_last_twist)
49
50
51  def keys_cb(msg):
52      global g_target_twist, g_last_twist, g_vel_scales
53      if len(msg.data) == 0 or not key_mapping.has_key(msg.data[0]):
54          return  # unknown key.
55      vels = key_mapping[msg.data[0]]
56      g_target_twist.angular.z = vels[0] * g_vel_scales[0]
57      g_target_twist.linear.x = vels[1] * g_vel_scales[1]
```

가속도를 고려한 Twist 계산

→ 문제점: 키보드 입력 값에 의해 계속 Twist 메시지 발생
→ 해결 방안: 키보드 입력 없으면 ramped_vel()를 적용하여 1초 후 속도가 0이 되게 해준다.

각속도/선속도 비율을 고려한 목표 각속도/선속도 계산

```
60  def fetch_param(name, default):
61      if rospy.has_param(name):
62          return rospy.get_param(name)
63      else:
64          print("parameter [%s] not defined. Defaulting to %.3f" % (name, default))
65          return default
66
67
68  if __name__ == '__main__':
69      rospy.init_node('keys_to_twist_with_ramps')
70      g_last_twist_send_time = rospy.Time.now()
71      g_twist_pub = rospy.Publisher('cmd_vel', Twist, queue_size=1)
72      rospy.Subscriber('keys', String, keys_cb)
73      g_target_twist = Twist()  # initializes to zero
74      g_last_twist = Twist()
```

매개변수 존재 여부 확인하여 매개변수 값을 반환.
없을 경우 default를 반환함.

```
75  g_vel_scales[0] = fetch_param('~angular_scale', 0.1)
76  g_vel_scales[1] = fetch_param('~linear_scale', 0.1)
77  g_vel_ramps[0] = fetch_param('~angular_accel', 1.0)
78  g_vel_ramps[1] = fetch_param('~linear_accel', 1.0)
79  print('g_vel_scales =', g_vel_scales)
80  print('g_vel_ramps =', g_vel_ramps)
81
82  rate = rospy.Rate(20)
83  while not rospy.is_shutdown():
84      send_twist()
85      rate.sleep()
```

각속도/선속도 비율을 매개변수에서 가져옴.

각속도/선속도의 가속도를 매개변수에서 가져옴.

실행 방법

Terminal 1:
$ roscore

Terminal 2:
$ rosrun deu_ros key_publisher.py

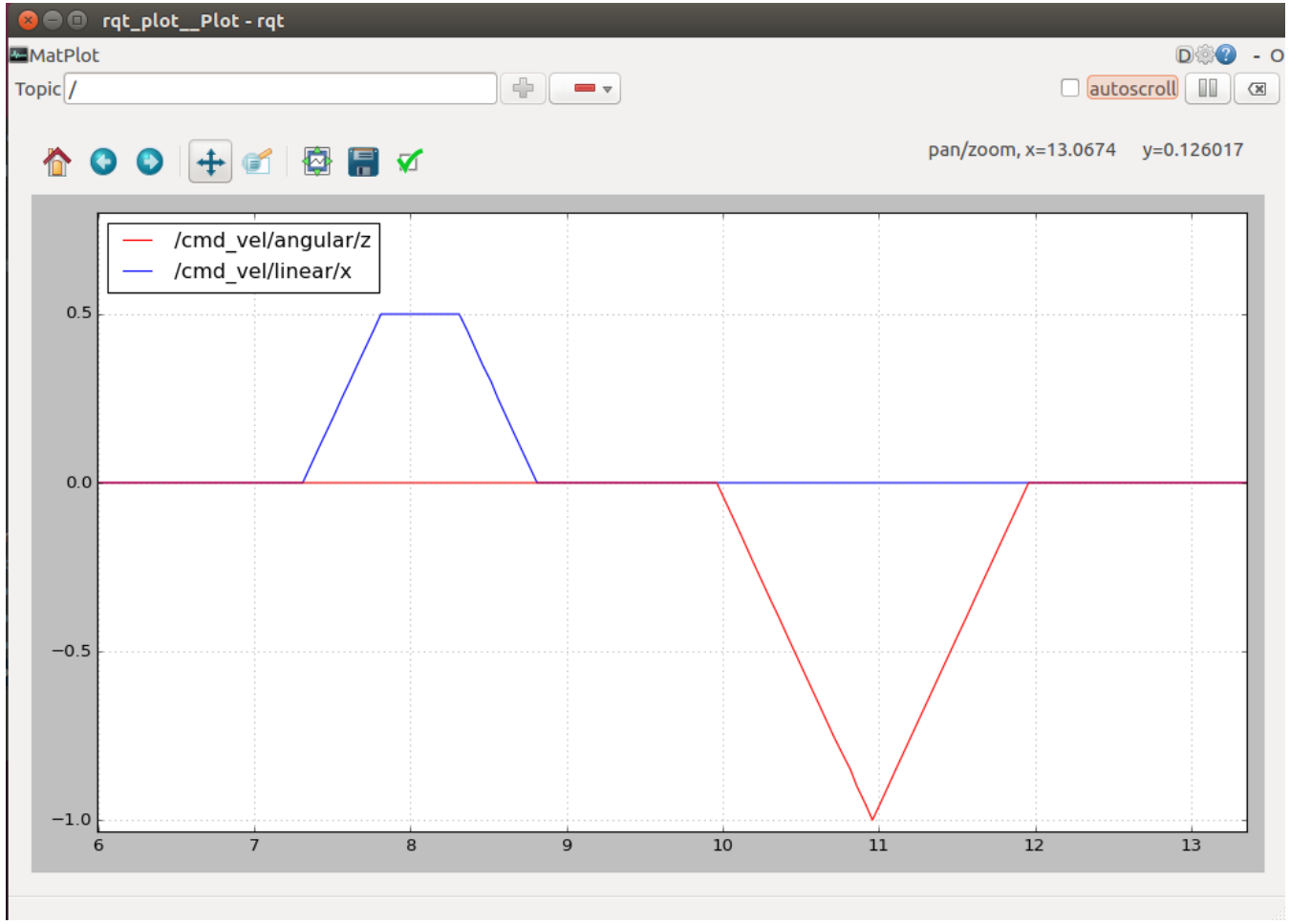
Terminal 3:
$ rosrun deu_ros keys_to_twist_with_ramps.py _linear_scale:=0.5 _angular_scale:=1.0 ₩
_linear_accel:=1.0 _angular_accel:=1.0

Terminal 4:
$ rostopic echo cmd_vel

Terminal 5:
$ rqt_plot /cmd_vel/linear/x /cmd_vel/angular/z

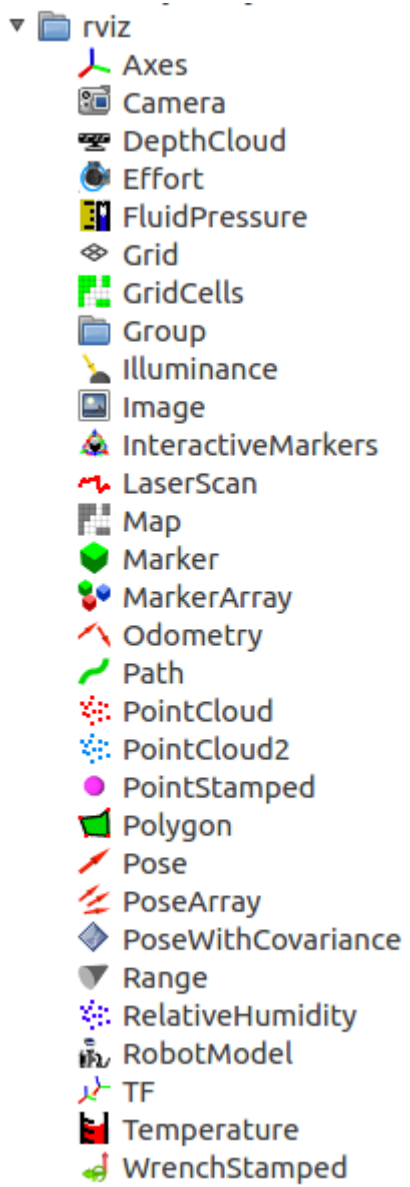
실행 결과

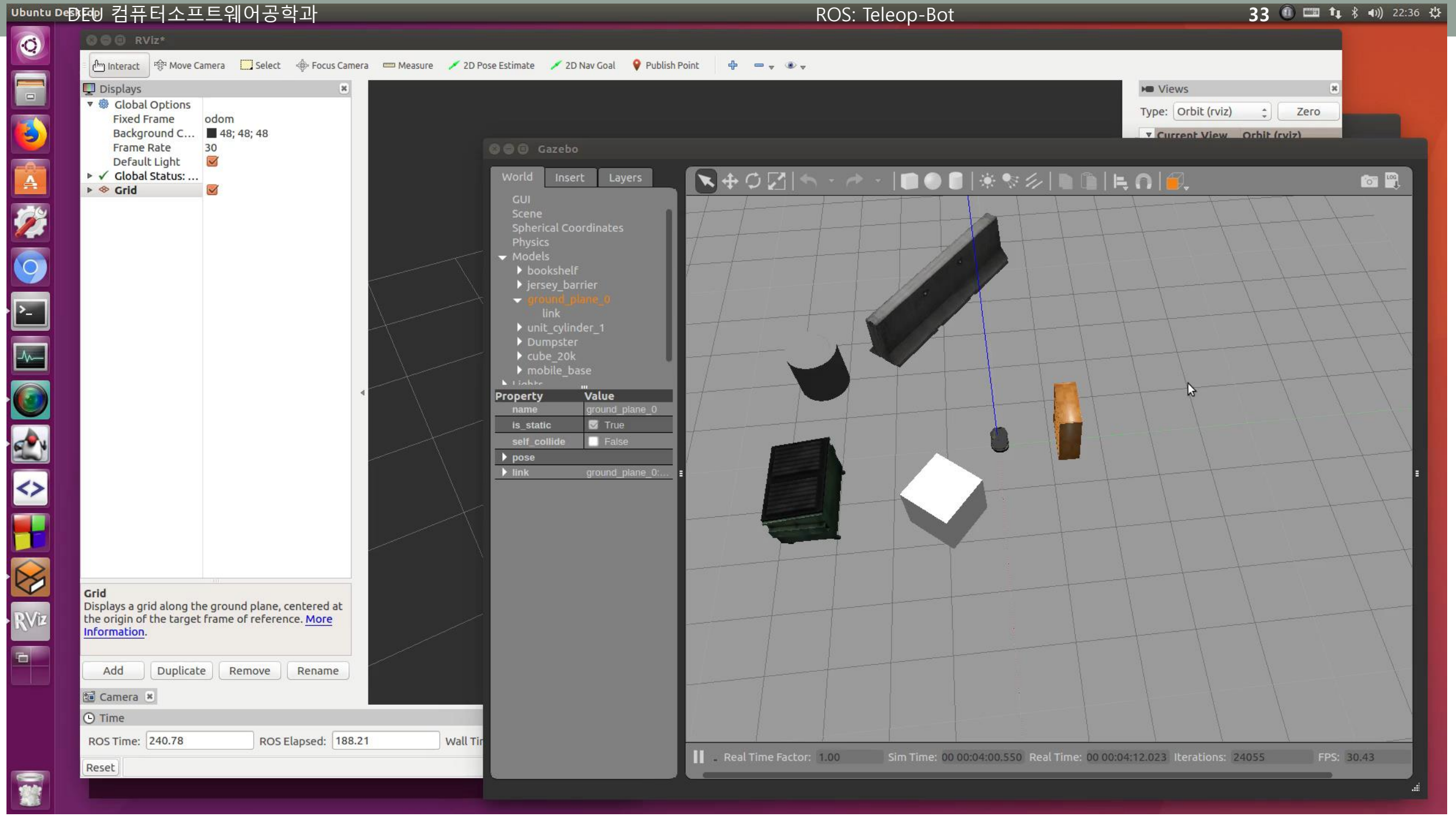
- rqt_plot 캡처 화면



rviz

- ROS Visualization
- Global Options
  - Fixed Frame: ROS의 모든 데이터는 참조 프레임에 부착됨.
    → 기본 값: 로봇은 base_link, 단순 주행 시 odom, 지도 사용 시 map,
    카메라는 camera_rgb_frame 등 (다를 수도 있음)
- 사용 가능한 시각화 요소
  - RobotModel: URDF 형식의 로봇 모델 보기
  - Camera: Image topic 보기. RobotModel, Image, LaserScan 등의 다른 토픽도 같이 볼 수 있음.
  - Image: Image topic 보기
  - LaserScan: LiDAR 토픽(/scan) 보기
  - Map: 주행 시 지도 보기
  - Odometry: nav_msgs/Odometry 메시지(/odom) 토픽 보기
  - Path: nav_msgs/Path 메시지 보기
  - …




RViz*
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Fixed Frame
odom
Background C...
48; 48; 48
Frame Rate
30
Default Light
Global Status: ...
Grid
Views
Type:
Orbit (rviz)
Zero
Grid
Displays a grid along the ground plane, centered at the origin of the target frame of reference. More Information.
Add
Duplicate
Remove
Rename
Camera
Time
ROS Time:
240.78
ROS Elapsed:
188.21
Reset
Gazebo
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Models
bookshelf
jersey_barrier
ground_plane_0
link
unit_cylinder_1
Dumpster
cube_20k
mobile_base
Property
Value
name
ground_plane_0
is_static
True
self_collide
False
pose
link
Real Time Factor:
1.00
Sim Time:
00 00:04:00.550
Real Time:
00 00:04:12.023
Iterations:
24055
FPS:
30.43

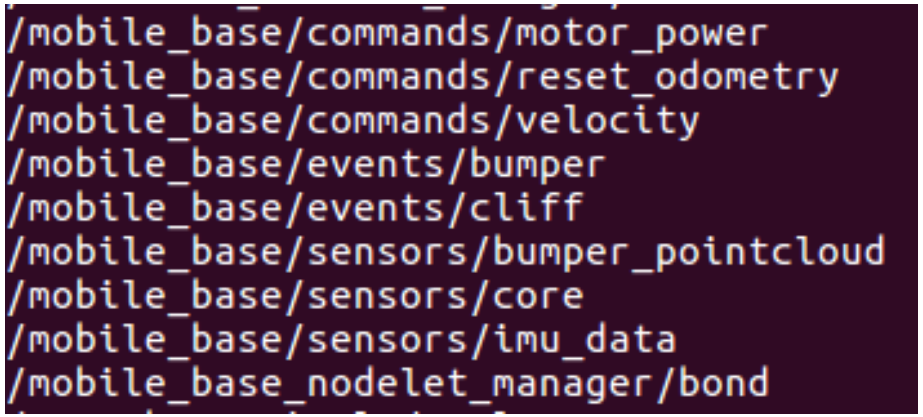
터틀봇 범퍼 이벤트

- 관련 메시지: kobuki_msgs/BumperEvent
- BumperEvent 메시지 형식
  - bumper: LEFT, CENTER, RIGHT 범퍼
  - state: RELEASED, PRESSED 상태 정보

```
jongmin@ubuntu:~$ rosmsg info kobuki_msgs/BumperEvent
uint8 LEFT=0
uint8 CENTER=1
uint8 RIGHT=2
uint8 RELEASED=0
uint8 PRESSED=1
uint8 bumper
uint8 state
```

터틀봇 mobile_base 관련 메시지

- rostopic list 명령어로 확인

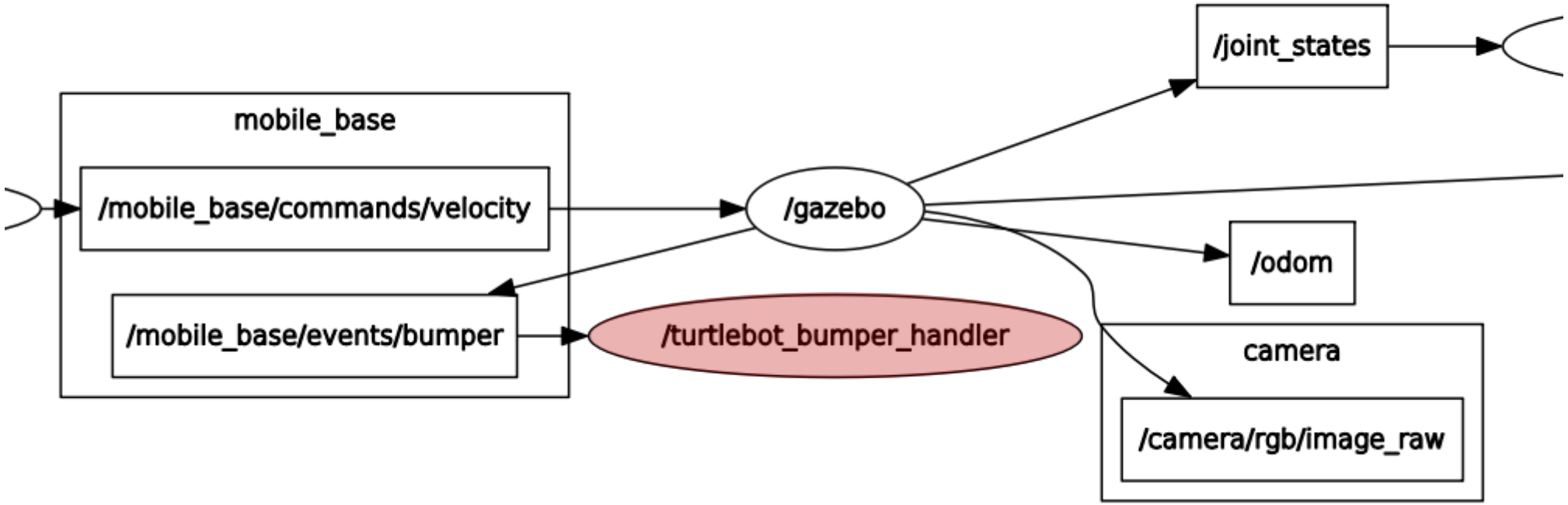


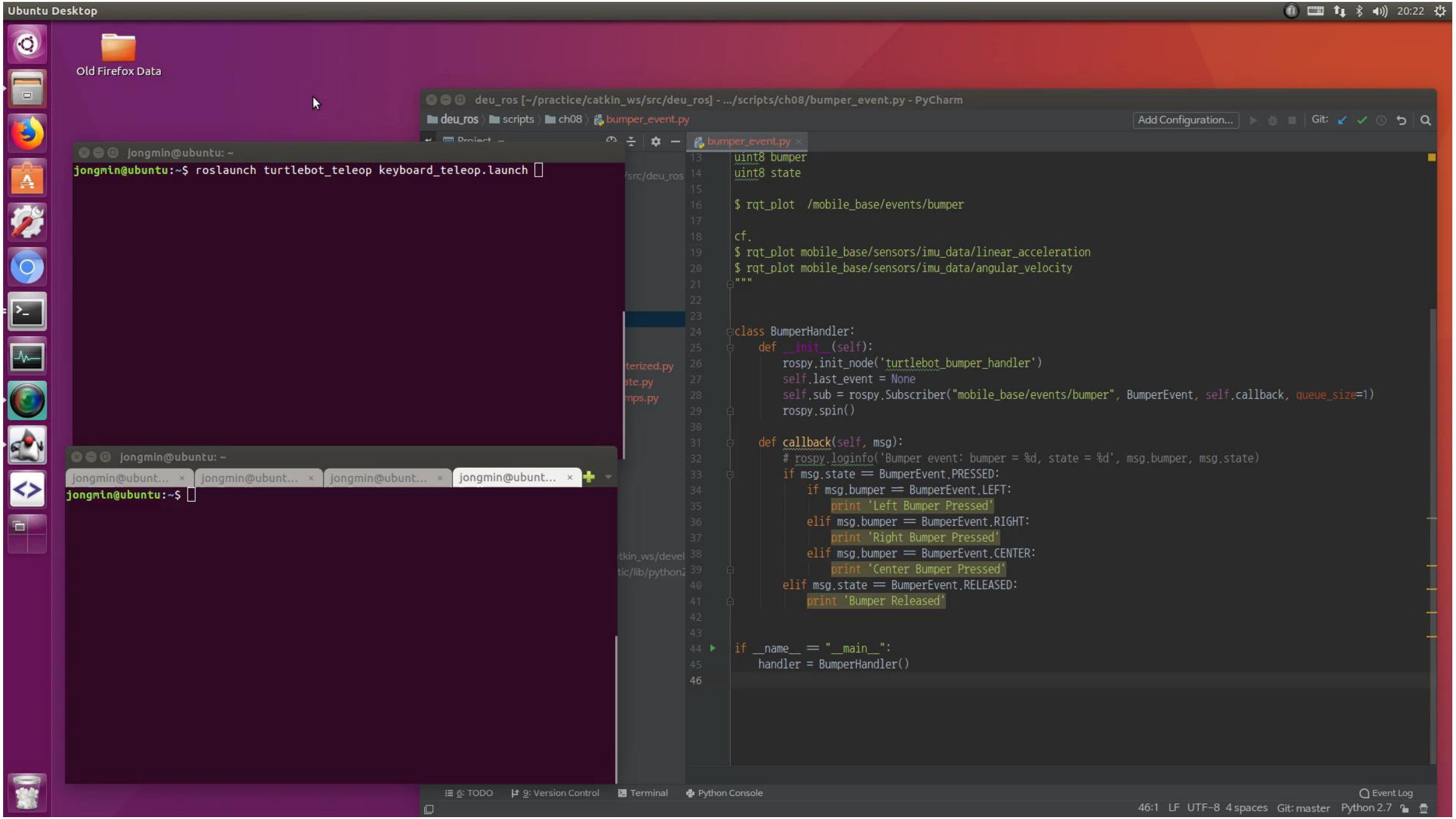
bumper_event.py

```
1  #!/usr/bin/env python
2
3  import rospy
4  from kobuki_msgs.msg import BumperEvent
5
6  """
7  jongmin@ubuntu:~$ rosmsg info kobuki_msgs/BumperEvent
8  uint8 LEFT=0
9  uint8 CENTER=1
10 uint8 RIGHT=2
11 uint8 RELEASED=0
12 uint8 PRESSED=1
13 uint8 bumper
14 uint8 state
15
16 $ rqt_plot  /mobile_base/events/bumper
17
18 cf.
19 $ rqt_plot mobile_base/sensors/imu_data/linear_acceleration
20 $ rqt_plot mobile_base/sensors/imu_data/angular_velocity
21 """
```

```
24 class BumperHandler:
25     def __init__(self):
26         rospy.init_node('turtlebot_bumper_handler')
27         self.last_event = None
28         self.sub = rospy.Subscriber("mobile_base/events/bumper", BumperEvent,
           self.callback, queue_size=1)
29         rospy.spin()
30
31     def callback(self, msg):
32         # rospy.loginfo('Bumper event: bumper = %d, state = %d', msg.bumper, msg.state)
33         if msg.state == BumperEvent.PRESSED:
34             if msg.bumper == BumperEvent.LEFT:
35                 print 'Left Bumper Pressed'
36             elif msg.bumper == BumperEvent.RIGHT:
37                 print 'Right Bumper Pressed'
38             elif msg.bumper == BumperEvent.CENTER:
39                 print 'Center Bumper Pressed'
40         elif msg.state == BumperEvent.RELEASED:
41             print 'Bumper Released'
42
43
44 if __name__ == "__main__":
45     handler = BumperHandler()
```

ROS 그래프: bumper_event.py




Ubuntu Desktop
20:22
Old Firefox Data
jongmin@ubuntu: ~
jongmin@ubuntu:~$ roslaunch turtlebot_teleop keyboard_teleop.launch
jongmin@ubuntu: ~
jongmin@ubuntu:~$
deu_ros [~/practice/catkin_ws/src/deu_ros] - .../scripts/ch08/bumper_event.py - PyCharm
deu_ros scripts ch08 bumper_event.py
Add Configuration...
Git:
bumper_event.py
uint8 bumper
uint8 state
$ rqt_plot /mobile_base/events/bumper
cf.
$ rqt_plot mobile_base/sensors/imu_data/linear_acceleration
$ rqt_plot mobile_base/sensors/imu_data/angular_velocity
"""
class BumperHandler:
def __init__(self):
rospy.init_node('turtlebot_bumper_handler')
self.last_event = None
self.sub = rospy.Subscriber("mobile_base/events/bumper", BumperEvent, self.callback, queue_size=1)
rospy.spin()
def callback(self, msg):
# rospy.loginfo('Bumper event: bumper = %d, state = %d', msg.bumper, msg.state)
if msg.state == BumperEvent.PRESSED:
if msg.bumper == BumperEvent.LEFT:
print 'Left Bumper Pressed'
elif msg.bumper == BumperEvent.RIGHT:
print 'Right Bumper Pressed'
elif msg.bumper == BumperEvent.CENTER:
print 'Center Bumper Pressed'
elif msg.state == BumperEvent.RELEASED:
print 'Bumper Released'
if __name__ == "__main__":
handler = BumperHandler()
6: TODO
9: Version Control
Terminal
Python Console
Event Log
46:1 LF UTF-8 4 spaces Git: master Python 2.7