10장. 주행 (NAVIGATION)

동의대학교 컴퓨터소프트웨어공학과
이종민 교수

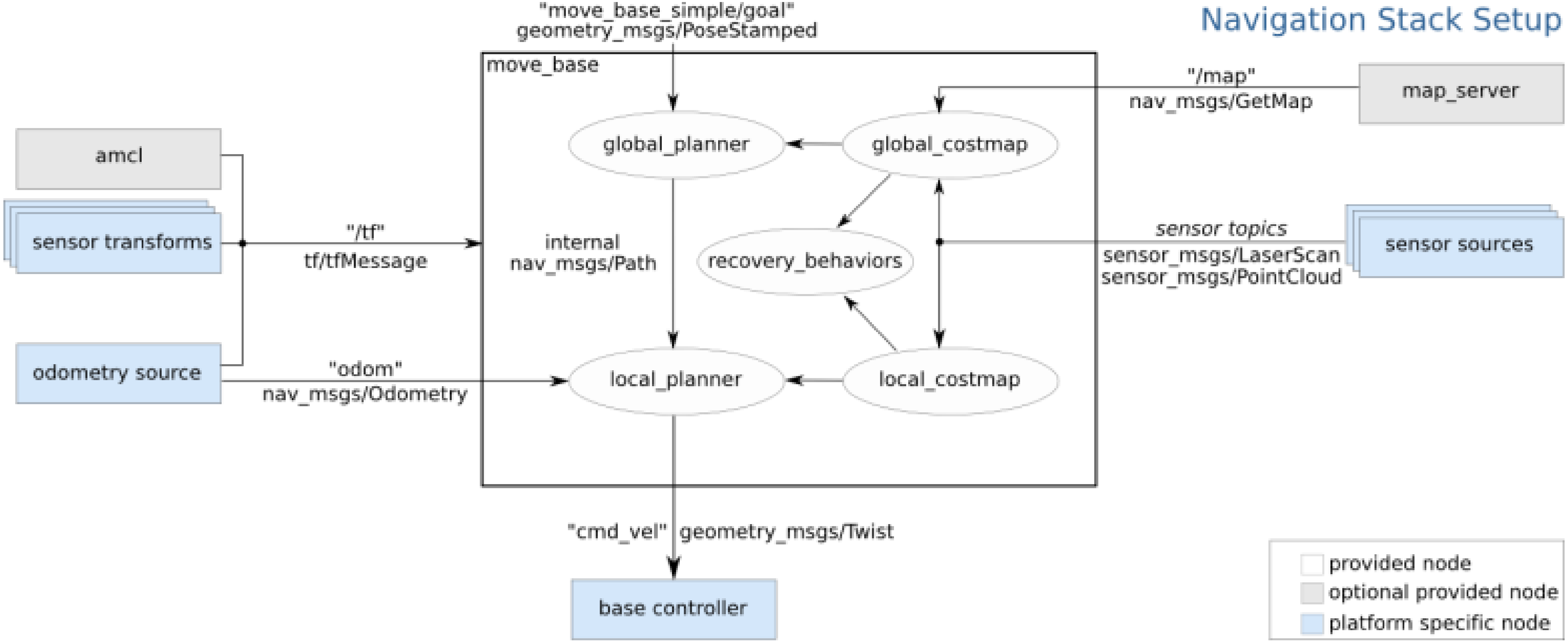
목차

- 주행 스택
- MoveBaseGoal을 사용한 주행

주행 스택

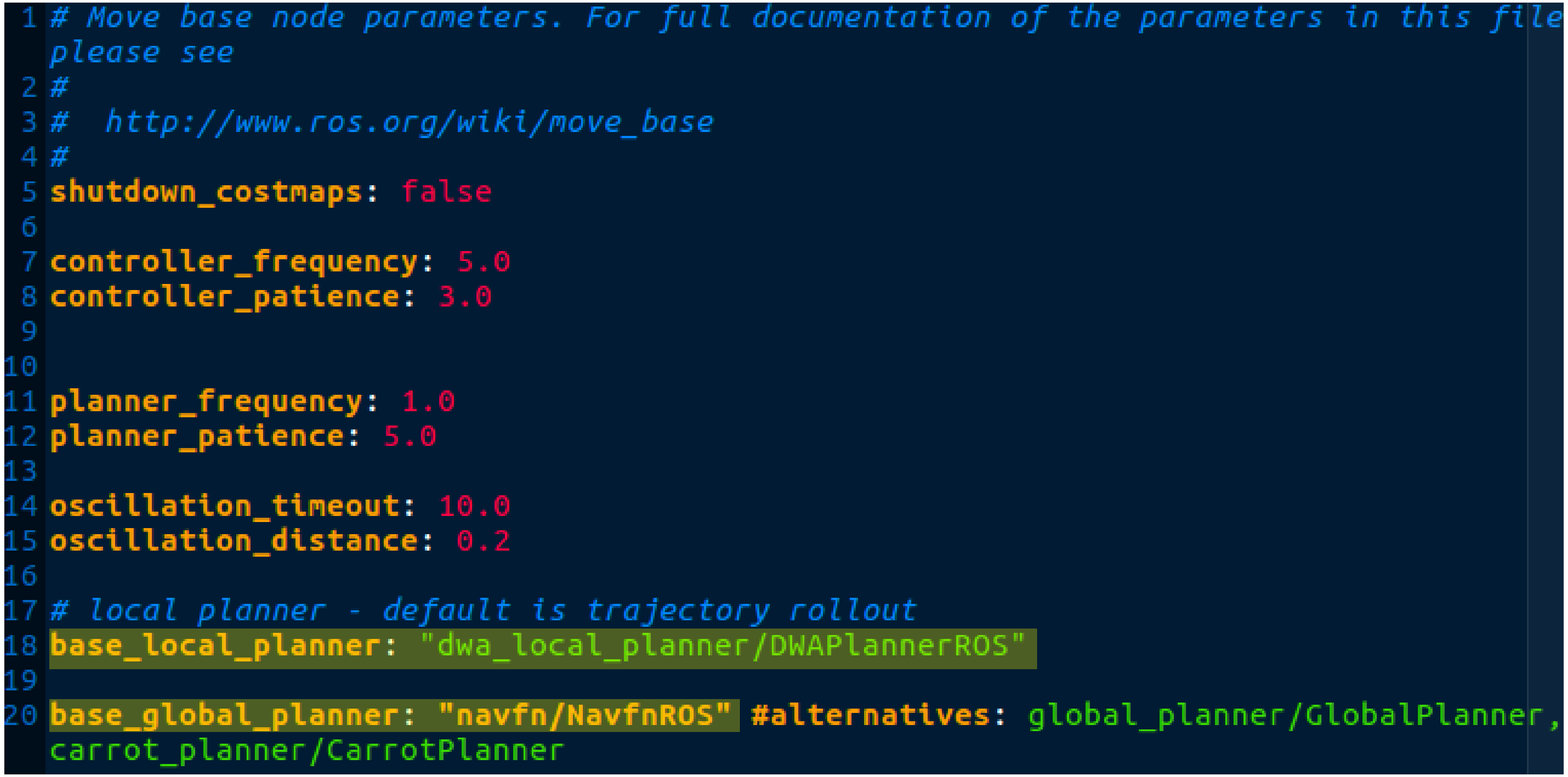
- URL: http://wiki.ros.org/navigation
- 주행 스택(navigation stack)
  - 입력: odometry와 센서 토픽
  - 출력: cmd_vel
- 기능
  - 위치 추정(amcl 패키지 사용. Adaptive Monte Carlo Localization. /scan 토픽 이용)
  - 목적지까지의 최단 경로 계산(Dijkstra & A* 알고리즘)
  - 주변 장애물 회피

주행 스택 구조


Navigation Stack Setup
"move_base_simple/goal"
geometry_msgs/PoseStamped
move_base
global_planner
global_costmap
"/map"
nav_msgs/GetMap
map_server
amcl
sensor transforms
"/tf"
tf/tfMessage
internal
nav_msgs/Path
recovery_behaviors
sensor topics
sensor_msgs/LaserScan
sensor_msgs/PointCloud
sensor sources
odometry source
"odom"
nav_msgs/Odometry
local_planner
local_costmap
"cmd_vel"
geometry_msgs/Twist
base controller
provided node
optional provided node
platform specific node

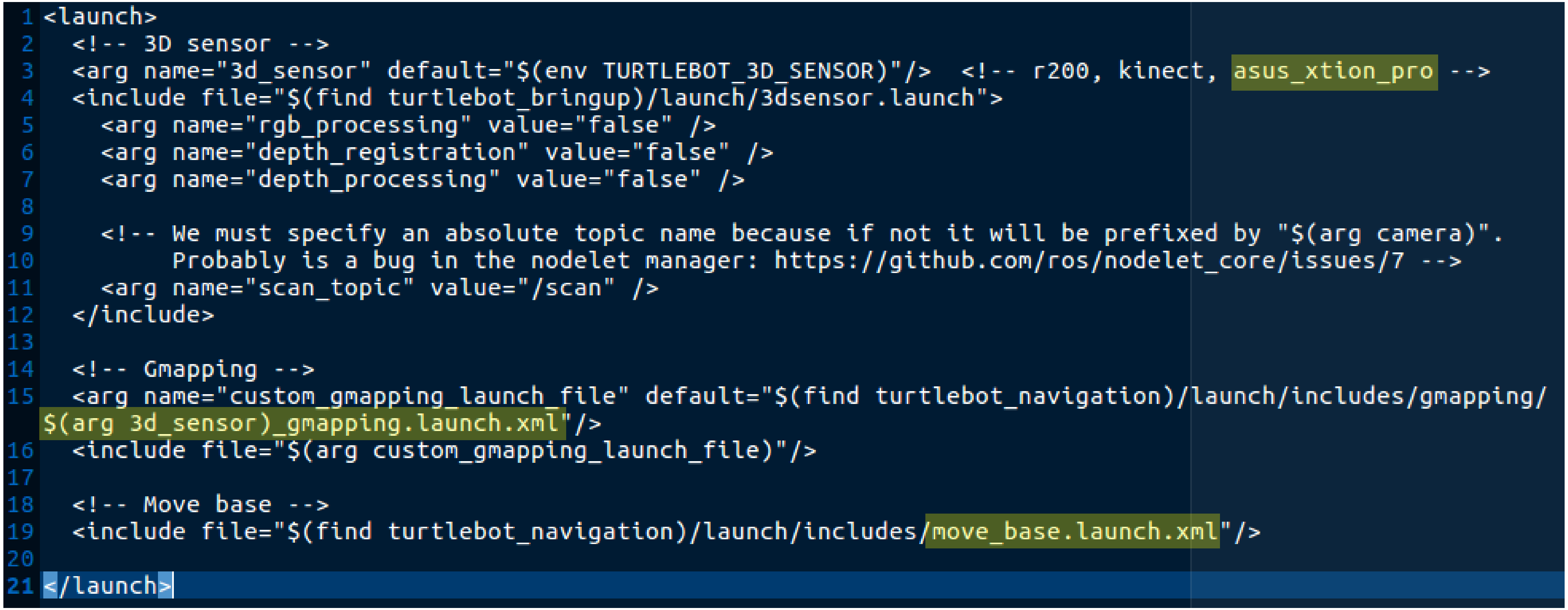
터틀봇 주행 스택

- 관련 launch 파일: /opt/ros/kinetic/share/turtlebot_navigation/launch/gmapping_demo.launch
- 관련 파일: /opt/ros/kinetic/share/turtlebot_navigation/param/move_base_params.yaml



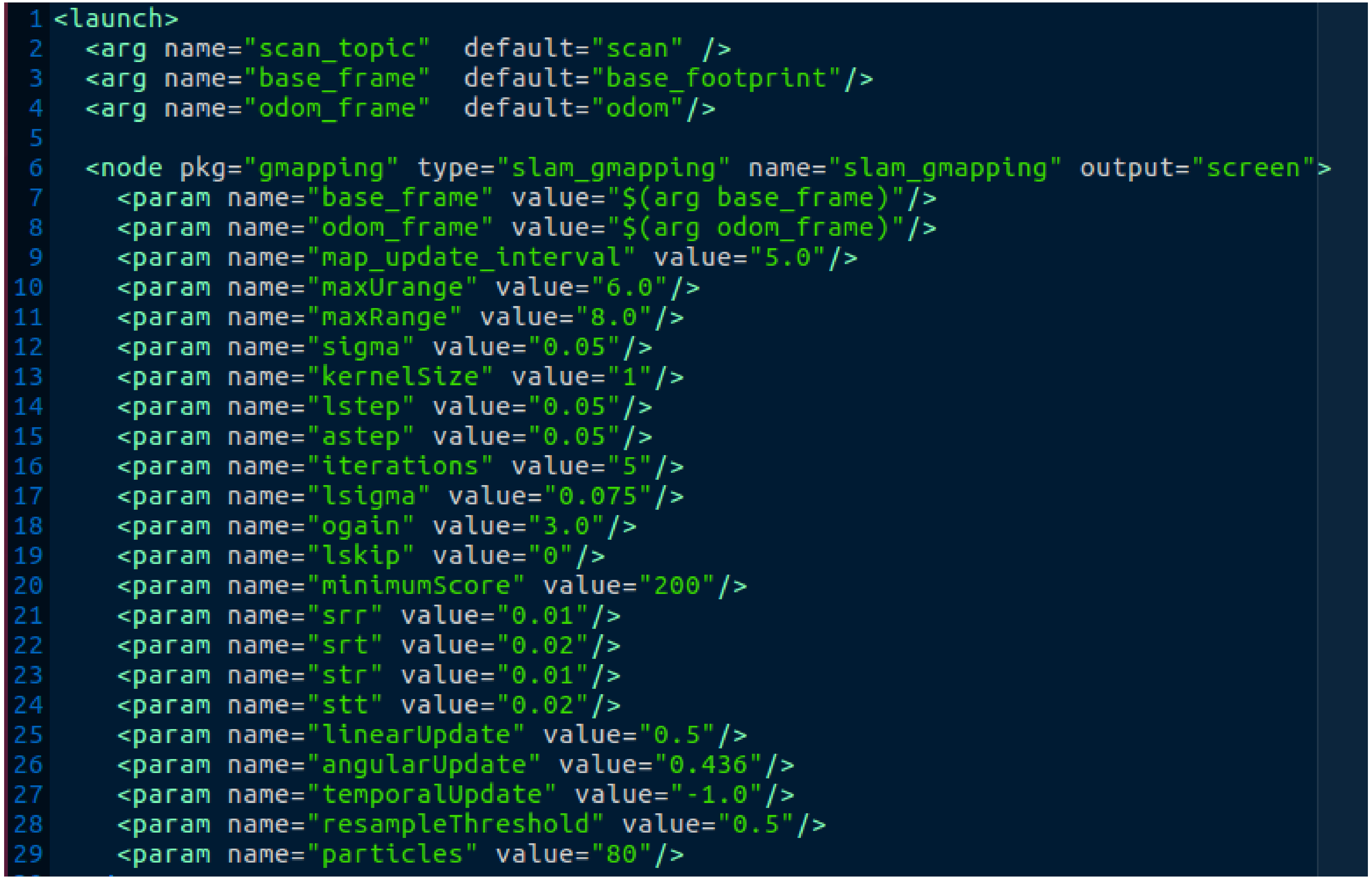
```yaml
# Move base node parameters. For full documentation of the parameters in this file please see
#
#  http://www.ros.org/wiki/move_base
#
shutdown_costmaps: false

controller_frequency: 5.0
controller_patience: 3.0


planner_frequency: 1.0
planner_patience: 5.0

oscillation_timeout: 10.0
oscillation_distance: 0.2

# local planner - default is trajectory rollout
base_local_planner: "dwa_local_planner/DWAPlannerROS"

base_global_planner: "navfn/NavfnROS" #alternatives: global_planner/GlobalPlanner, carrot_planner/CarrotPlanner
```

gmapping_demo.launch



```
<launch>
  <!-- 3D sensor -->
  <arg name="3d_sensor" default="$(env TURTLEBOT_3D_SENSOR)"/>  <!-- r200, kinect, asus_xtion_pro -->
  <include file="$(find turtlebot_bringup)/launch/3dsensor.launch">
    <arg name="rgb_processing" value="false" />
    <arg name="depth_registration" value="false" />
    <arg name="depth_processing" value="false" />

    <!-- We must specify an absolute topic name because if not it will be prefixed by "$(arg camera)".
         Probably is a bug in the nodelet manager: https://github.com/ros/nodelet_core/issues/7 -->
    <arg name="scan_topic" value="/scan" />
  </include>

  <!-- Gmapping -->
  <arg name="custom_gmapping_launch_file" default="$(find turtlebot_navigation)/launch/includes/gmapping/$(arg 3d_sensor)_gmapping.launch.xml"/>
  <include file="$(arg custom_gmapping_launch_file)"/>

  <!-- Move base -->
  <include file="$(find turtlebot_navigation)/launch/includes/move_base.launch.xml"/>

</launch>
```

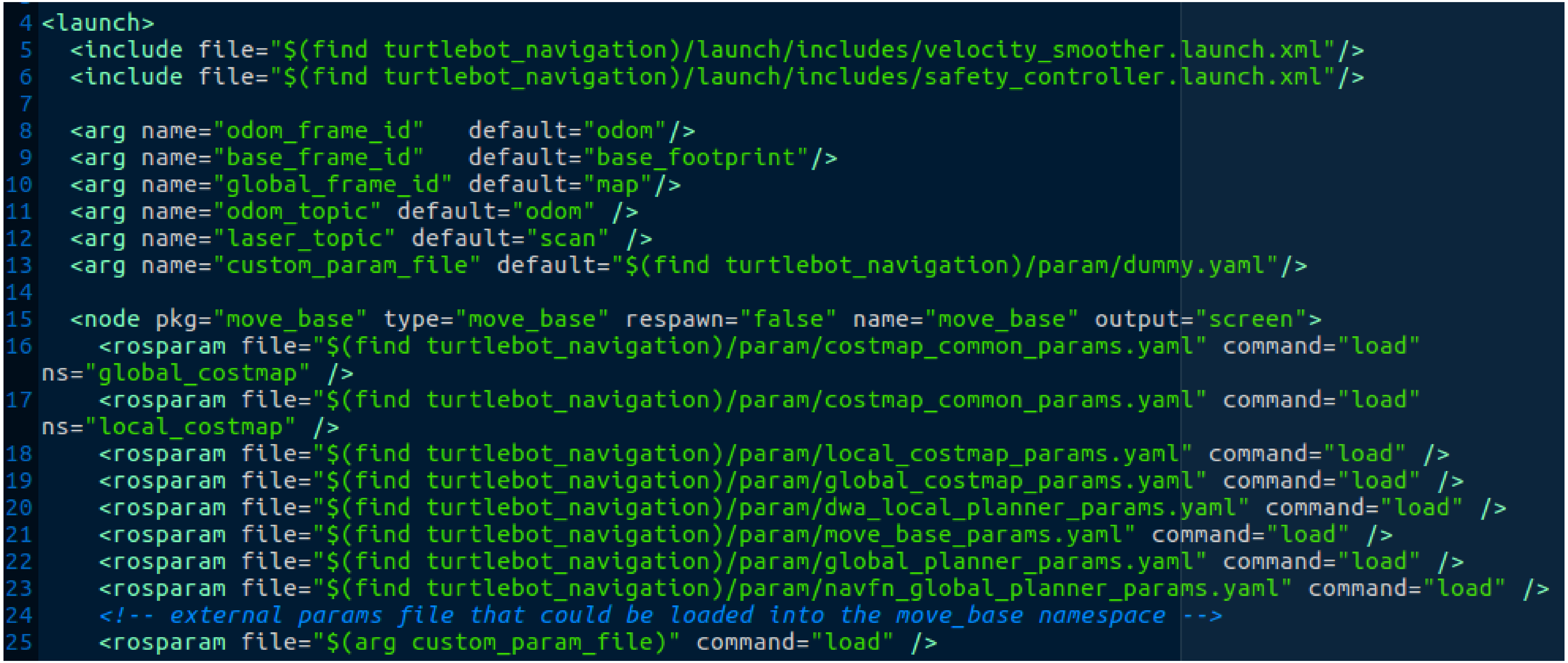
asus_xtion_pro_gmapping.launch.xml



```
<launch>
  <arg name="scan_topic"  default="scan" />
  <arg name="base_frame"  default="base_footprint"/>
  <arg name="odom_frame"  default="odom"/>

  <node pkg="gmapping" type="slam_gmapping" name="slam_gmapping" output="screen">
    <param name="base_frame" value="$(arg base_frame)"/>
    <param name="odom_frame" value="$(arg odom_frame)"/>
    <param name="map_update_interval" value="5.0"/>
    <param name="maxUrange" value="6.0"/>
    <param name="maxRange" value="8.0"/>
    <param name="sigma" value="0.05"/>
    <param name="kernelSize" value="1"/>
    <param name="lstep" value="0.05"/>
    <param name="astep" value="0.05"/>
    <param name="iterations" value="5"/>
    <param name="lsigma" value="0.075"/>
    <param name="ogain" value="3.0"/>
    <param name="lskip" value="0"/>
    <param name="minimumScore" value="200"/>
    <param name="srr" value="0.01"/>
    <param name="srt" value="0.02"/>
    <param name="str" value="0.01"/>
    <param name="stt" value="0.02"/>
    <param name="linearUpdate" value="0.5"/>
    <param name="angularUpdate" value="0.436"/>
    <param name="temporalUpdate" value="-1.0"/>
    <param name="resampleThreshold" value="0.5"/>
    <param name="particles" value="80"/>
```

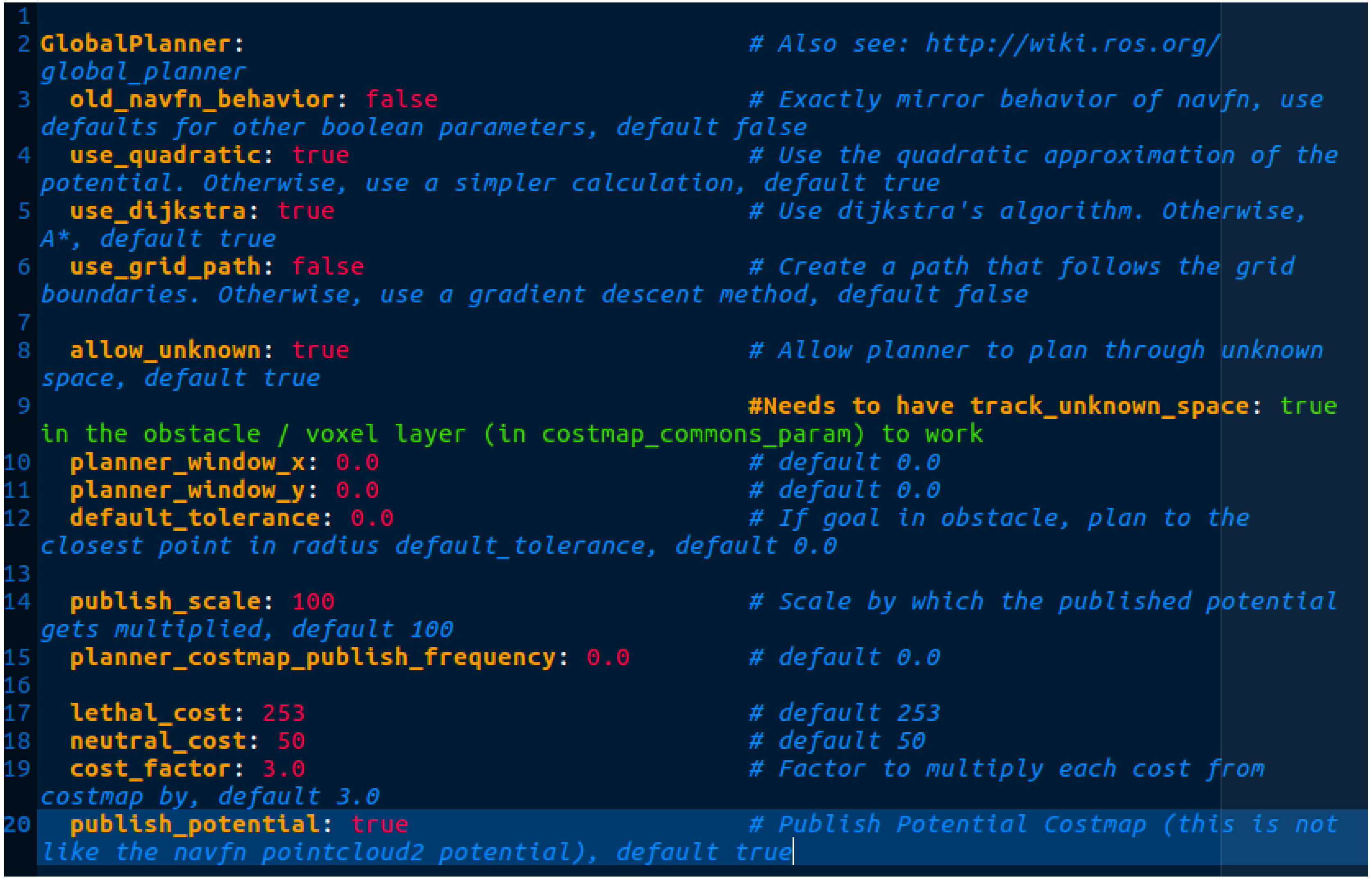
move_base.launch.xml

- /opt/ros/kinetic/share/turtlebot_navigation/launch/includes/move_base.launch.xml



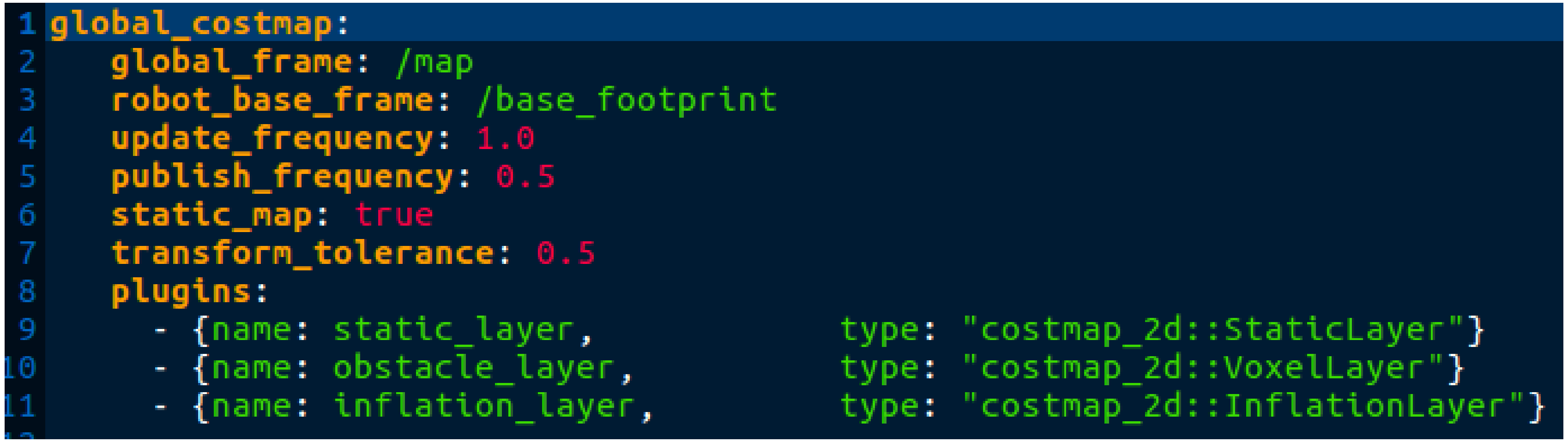
```
<launch>
  <include file="$(find turtlebot_navigation)/launch/includes/velocity_smoother.launch.xml"/>
  <include file="$(find turtlebot_navigation)/launch/includes/safety_controller.launch.xml"/>

  <arg name="odom_frame_id"   default="odom"/>
  <arg name="base_frame_id"   default="base_footprint"/>
  <arg name="global_frame_id" default="map"/>
  <arg name="odom_topic" default="odom" />
  <arg name="laser_topic" default="scan" />
  <arg name="custom_param_file" default="$(find turtlebot_navigation)/param/dummy.yaml"/>

  <node pkg="move_base" type="move_base" respawn="false" name="move_base" output="screen">
    <rosparam file="$(find turtlebot_navigation)/param/costmap_common_params.yaml" command="load"
ns="global_costmap" />
    <rosparam file="$(find turtlebot_navigation)/param/costmap_common_params.yaml" command="load"
ns="local_costmap" />
    <rosparam file="$(find turtlebot_navigation)/param/local_costmap_params.yaml" command="load" />
    <rosparam file="$(find turtlebot_navigation)/param/global_costmap_params.yaml" command="load" />
    <rosparam file="$(find turtlebot_navigation)/param/dwa_local_planner_params.yaml" command="load" />
    <rosparam file="$(find turtlebot_navigation)/param/move_base_params.yaml" command="load" />
    <rosparam file="$(find turtlebot_navigation)/param/global_planner_params.yaml" command="load" />
    <rosparam file="$(find turtlebot_navigation)/param/navfn_global_planner_params.yaml" command="load" />
    <!-- external params file that could be loaded into the move_base namespace -->
    <rosparam file="$(arg custom_param_file)" command="load" />
```

global_planner_params.yaml



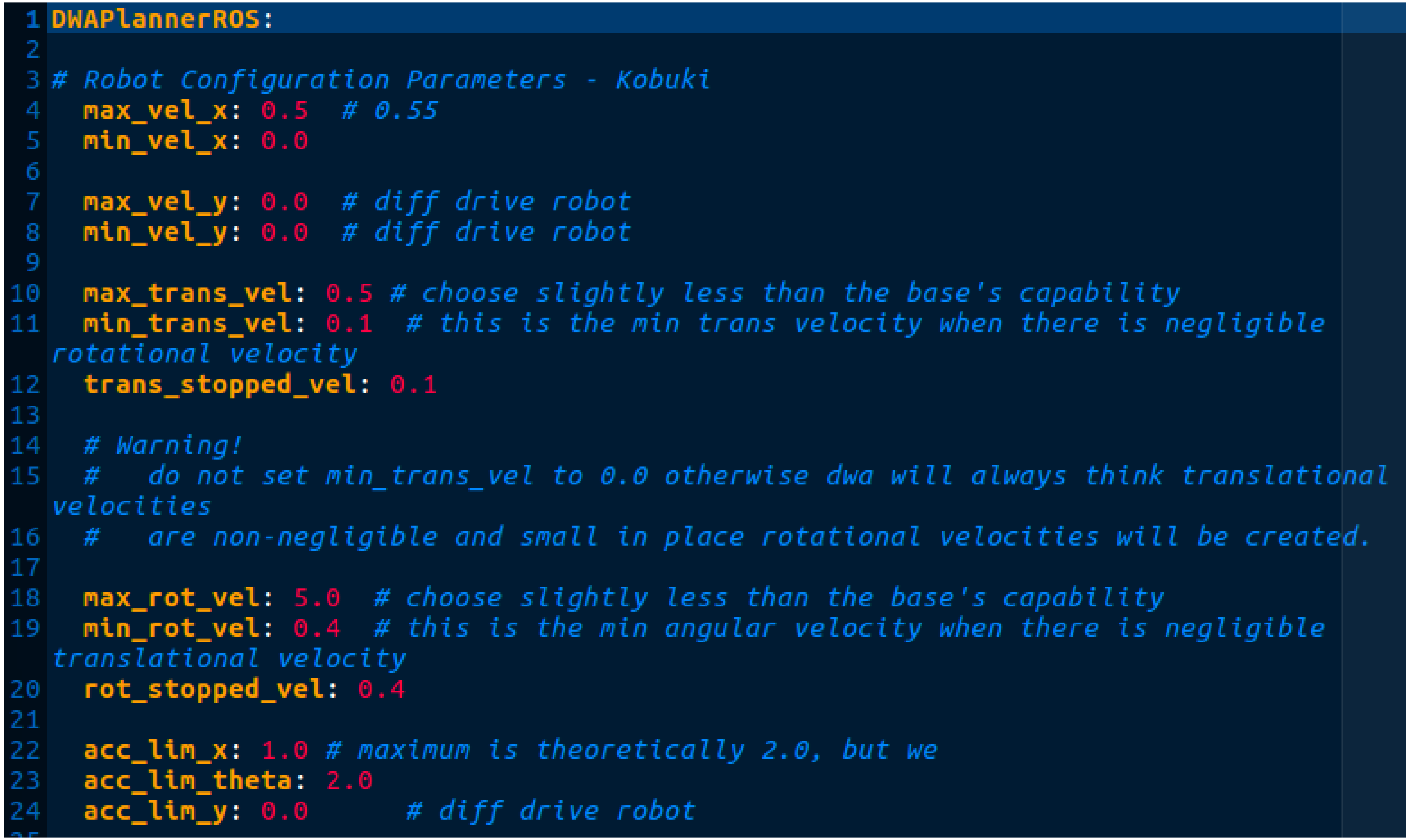
```yaml

GlobalPlanner:                                  # Also see: http://wiki.ros.org/global_planner
  old_navfn_behavior: false                     # Exactly mirror behavior of navfn, use defaults for other boolean parameters, default false
  use_quadratic: true                           # Use the quadratic approximation of the potential. Otherwise, use a simpler calculation, default true
  use_dijkstra: true                            # Use dijkstra's algorithm. Otherwise, A*, default true
  use_grid_path: false                          # Create a path that follows the grid boundaries. Otherwise, use a gradient descent method, default false

  allow_unknown: true                           # Allow planner to plan through unknown space, default true
                                                #Needs to have track_unknown_space: true in the obstacle / voxel layer (in costmap_commons_param) to work
  planner_window_x: 0.0                         # default 0.0
  planner_window_y: 0.0                         # default 0.0
  default_tolerance: 0.0                        # If goal in obstacle, plan to the closest point in radius default_tolerance, default 0.0

  publish_scale: 100                            # Scale by which the published potential gets multiplied, default 100
  planner_costmap_publish_frequency: 0.0        # default 0.0

  lethal_cost: 253                              # default 253
  neutral_cost: 50                              # default 50
  cost_factor: 3.0                              # Factor to multiply each cost from costmap by, default 3.0
  publish_potential: true                       # Publish Potential Costmap (this is not like the navfn pointcloud2 potential), default true
```

global_costmap_params.yaml



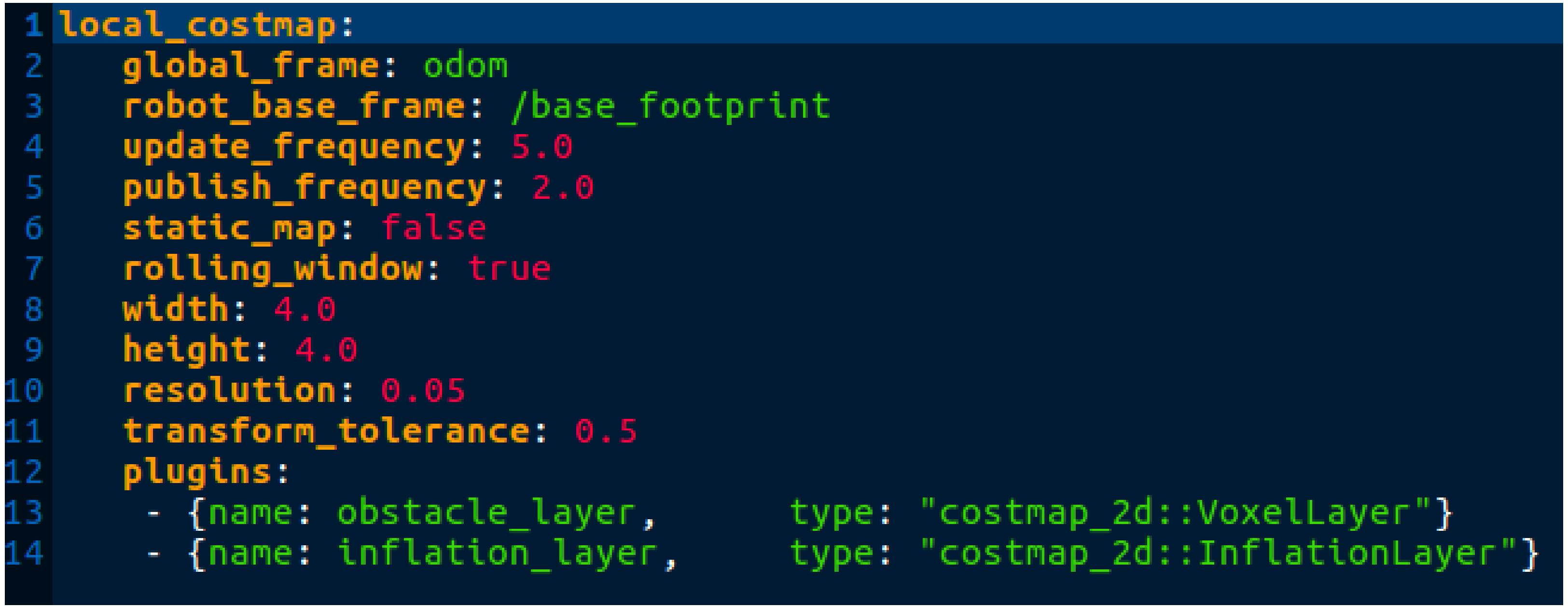
```
global_costmap:
   global_frame: /map
   robot_base_frame: /base_footprint
   update_frequency: 1.0
   publish_frequency: 0.5
   static_map: true
   transform_tolerance: 0.5
   plugins:
     - {name: static_layer,             type: "costmap_2d::StaticLayer"}
     - {name: obstacle_layer,           type: "costmap_2d::VoxelLayer"}
     - {name: inflation_layer,          type: "costmap_2d::InflationLayer"}
```

dwa_local_planner_params.yaml



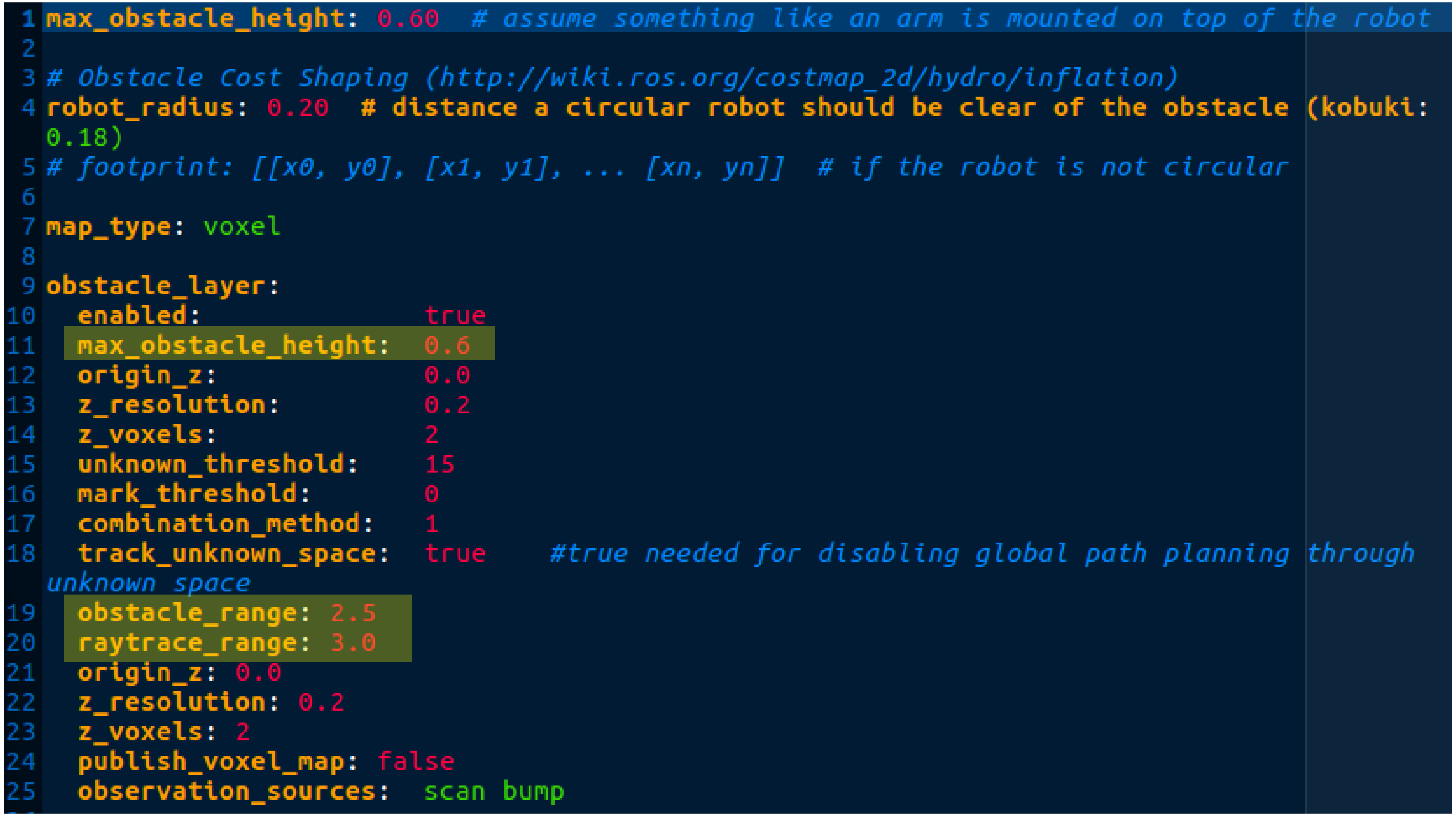
```yaml
DWAPlannerROS:

# Robot Configuration Parameters - Kobuki
  max_vel_x: 0.5  # 0.55
  min_vel_x: 0.0

  max_vel_y: 0.0  # diff drive robot
  min_vel_y: 0.0  # diff drive robot

  max_trans_vel: 0.5 # choose slightly less than the base's capability
  min_trans_vel: 0.1  # this is the min trans velocity when there is negligible rotational velocity
  trans_stopped_vel: 0.1

  # Warning!
  #   do not set min_trans_vel to 0.0 otherwise dwa will always think translational velocities
  #   are non-negligible and small in place rotational velocities will be created.

  max_rot_vel: 5.0  # choose slightly less than the base's capability
  min_rot_vel: 0.4  # this is the min angular velocity when there is negligible translational velocity
  rot_stopped_vel: 0.4

  acc_lim_x: 1.0 # maximum is theoretically 2.0, but we
  acc_lim_theta: 2.0
  acc_lim_y: 0.0      # diff drive robot
```

local_cost_map_params.yaml

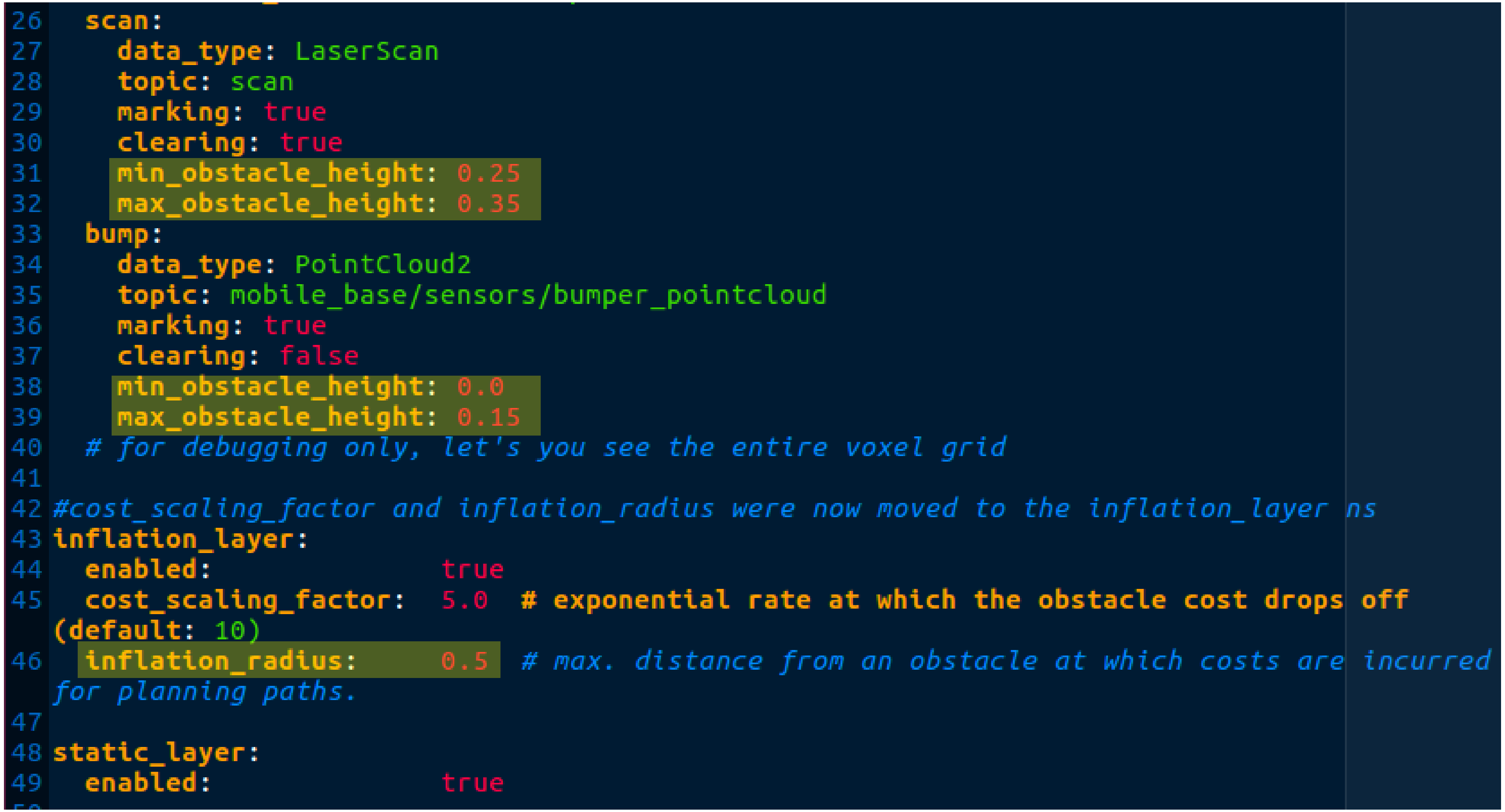


```
local_costmap:
   global_frame: odom
   robot_base_frame: /base_footprint
   update_frequency: 5.0
   publish_frequency: 2.0
   static_map: false
   rolling_window: true
   width: 4.0
   height: 4.0
   resolution: 0.05
   transform_tolerance: 0.5
   plugins:
    - {name: obstacle_layer,      type: "costmap_2d::VoxelLayer"}
    - {name: inflation_layer,     type: "costmap_2d::InflationLayer"}
```

costmap_common_params.yaml

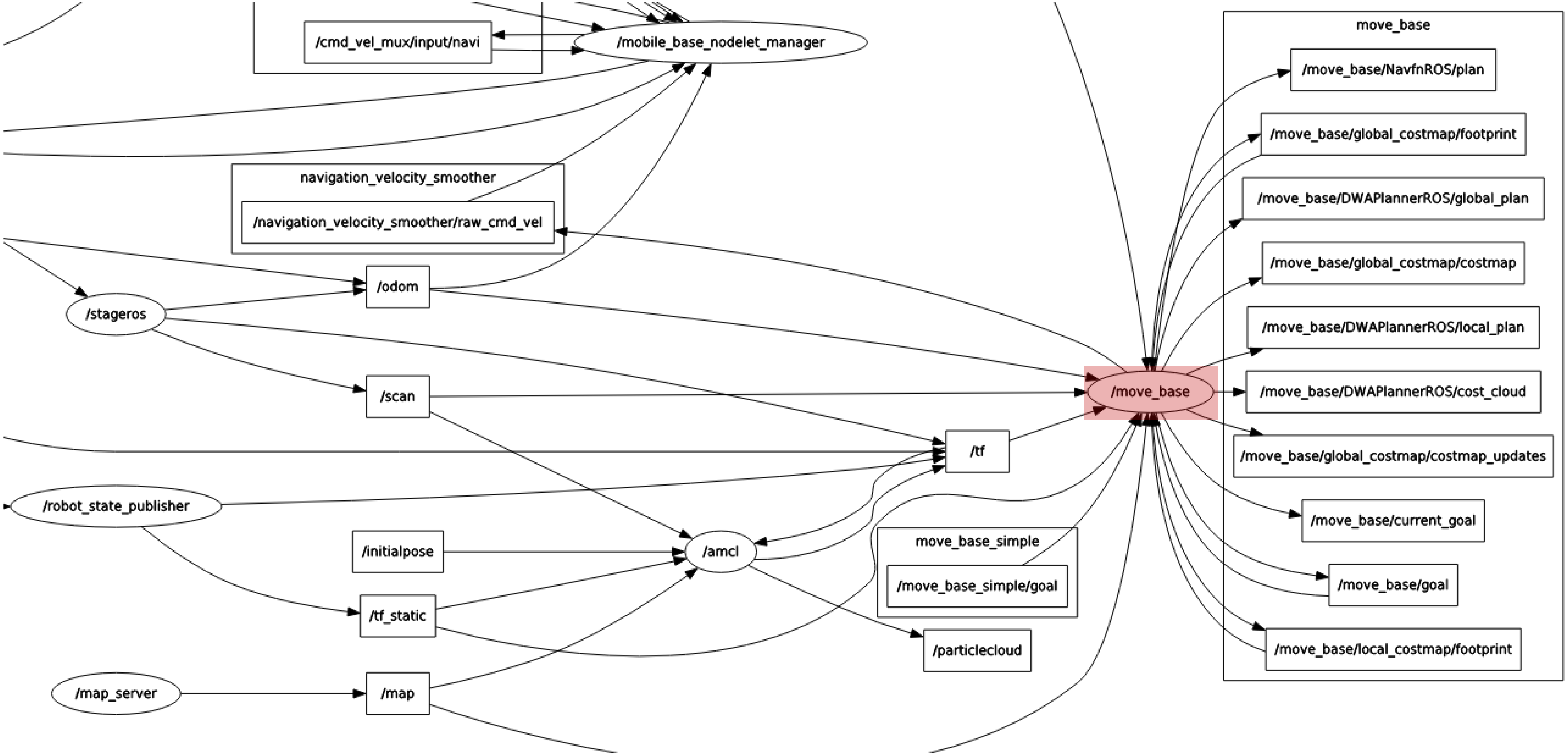


```yaml
max_obstacle_height: 0.60  # assume something like an arm is mounted on top of the robot

# Obstacle Cost Shaping (http://wiki.ros.org/costmap_2d/hydro/inflation)
robot_radius: 0.20  # distance a circular robot should be clear of the obstacle (kobuki: 0.18)
# footprint: [[x0, y0], [x1, y1], ... [xn, yn]]  # if the robot is not circular

map_type: voxel

obstacle_layer:
  enabled:              true
  max_obstacle_height:  0.6
  origin_z:             0.0
  z_resolution:         0.2
  z_voxels:             2
  unknown_threshold:    15
  mark_threshold:       0
  combination_method:   1
  track_unknown_space:  true    #true needed for disabling global path planning through unknown space
  obstacle_range: 2.5
  raytrace_range: 3.0
  origin_z: 0.0
  z_resolution: 0.2
  z_voxels: 2
  publish_voxel_map: false
  observation_sources:  scan bump
```



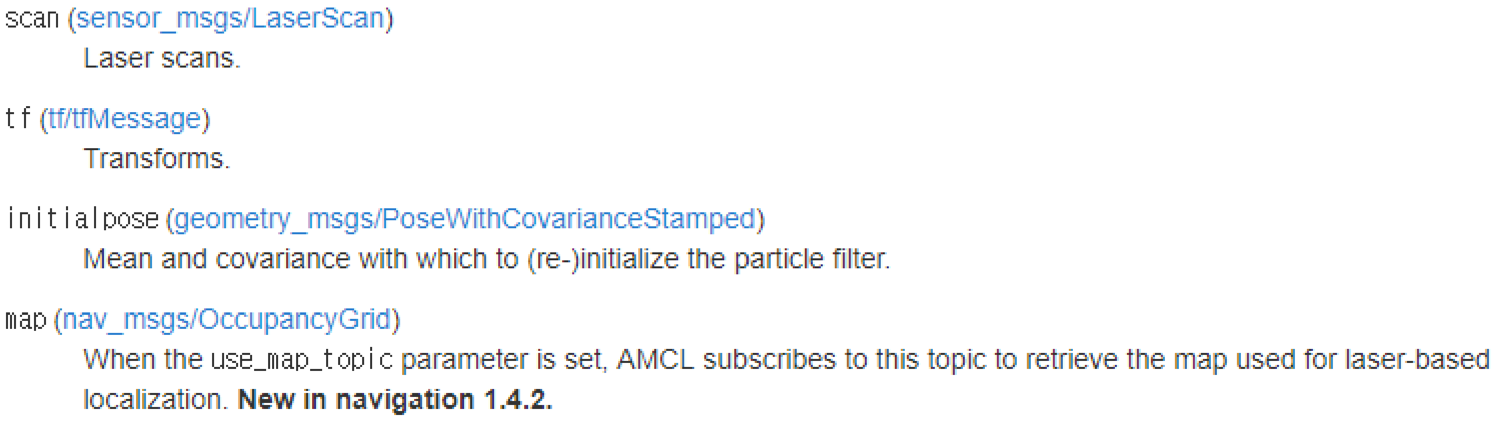
```
  scan:
    data_type: LaserScan
    topic: scan
    marking: true
    clearing: true
    min_obstacle_height: 0.25
    max_obstacle_height: 0.35
  bump:
    data_type: PointCloud2
    topic: mobile_base/sensors/bumper_pointcloud
    marking: true
    clearing: false
    min_obstacle_height: 0.0
    max_obstacle_height: 0.15
  # for debugging only, let's you see the entire voxel grid

#cost_scaling_factor and inflation_radius were now moved to the inflation_layer ns
inflation_layer:
  enabled:              true
  cost_scaling_factor:  5.0  # exponential rate at which the obstacle cost drops off (default: 10)
  inflation_radius:     0.5  # max. distance from an obstacle at which costs are incurred for planning paths.

static_layer:
  enabled:              true
```

터틀봇 move_base 노드


/cmd_vel_mux/input/navi
/mobile_base_nodelet_manager
navigation_velocity_smoother
/navigation_velocity_smoother/raw_cmd_vel
/odom
/stageros
/scan
/tf
/robot_state_publisher
/initialpose
/amcl
/tf_static
/map_server
/map
move_base_simple
/move_base_simple/goal
/particlecloud
/move_base
move_base
/move_base/NavfnROS/plan
/move_base/global_costmap/footprint
/move_base/DWAPlannerROS/global_plan
/move_base/global_costmap/costmap
/move_base/DWAPlannerROS/local_plan
/move_base/DWAPlannerROS/cost_cloud
/move_base/global_costmap/costmap_updates
/move_base/current_goal
/move_base/goal
/move_base/local_costmap/footprint

amcl 패키지

- URL: http://wiki.ros.org/amcl
- 이차원 이동 로봇을 위한 확률 위치 추정 패키지
- 노드: amcl
  - 구독 토픽



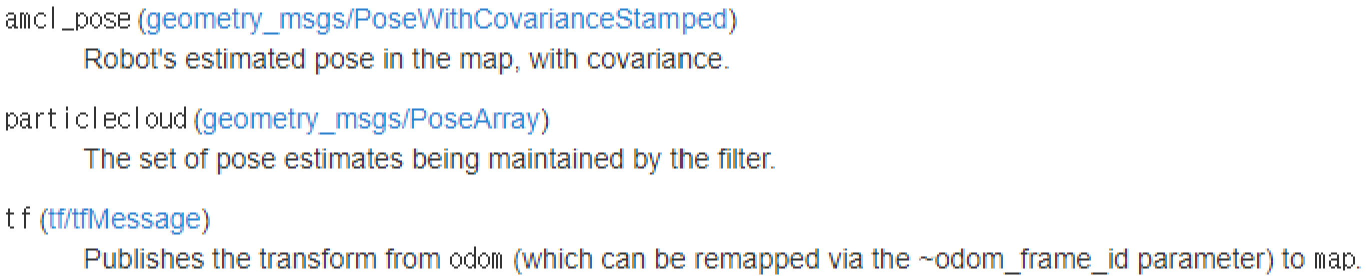
scan (sensor_msgs/LaserScan)
    Laser scans.

tf (tf/tfMessage)
    Transforms.

initialpose (geometry_msgs/PoseWithCovarianceStamped)
    Mean and covariance with which to (re-)initialize the particle filter.

map (nav_msgs/OccupancyGrid)
    When the use_map_topic parameter is set, AMCL subscribes to this topic to retrieve the map used for laser-based localization. **New in navigation 1.4.2.**

  - 발행 토픽

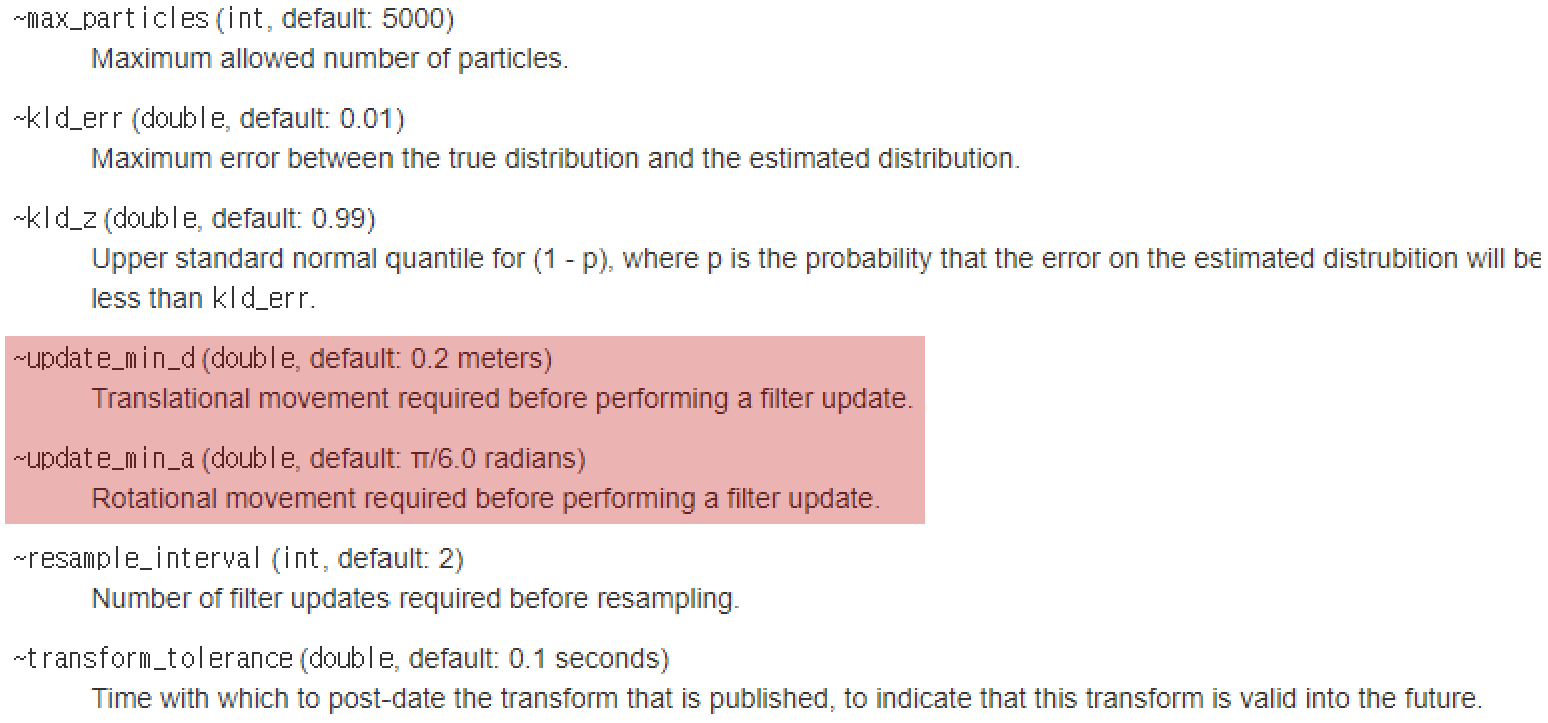


amcl_pose (geometry_msgs/PoseWithCovarianceStamped)
    Robot's estimated pose in the map, with covariance.

particlecloud (geometry_msgs/PoseArray)
    The set of pose estimates being maintained by the filter.

tf (tf/tfMessage)
    Publishes the transform from odom (which can be remapped via the ~odom_frame_id parameter) to map.

- 노드: amcl (계속)
  - 매개변수



~max_particles (int, default: 5000)

Maximum allowed number of particles.

~kld_err (double, default: 0.01)

Maximum error between the true distribution and the estimated distribution.

~kld_z (double, default: 0.99)

Upper standard normal quantile for (1 - p), where p is the probability that the error on the estimated distrubition will be less than kld_err.

~update_min_d (double, default: 0.2 meters)

Translational movement required before performing a filter update.

~update_min_a (double, default: π/6.0 radians)

Rotational movement required before performing a filter update.

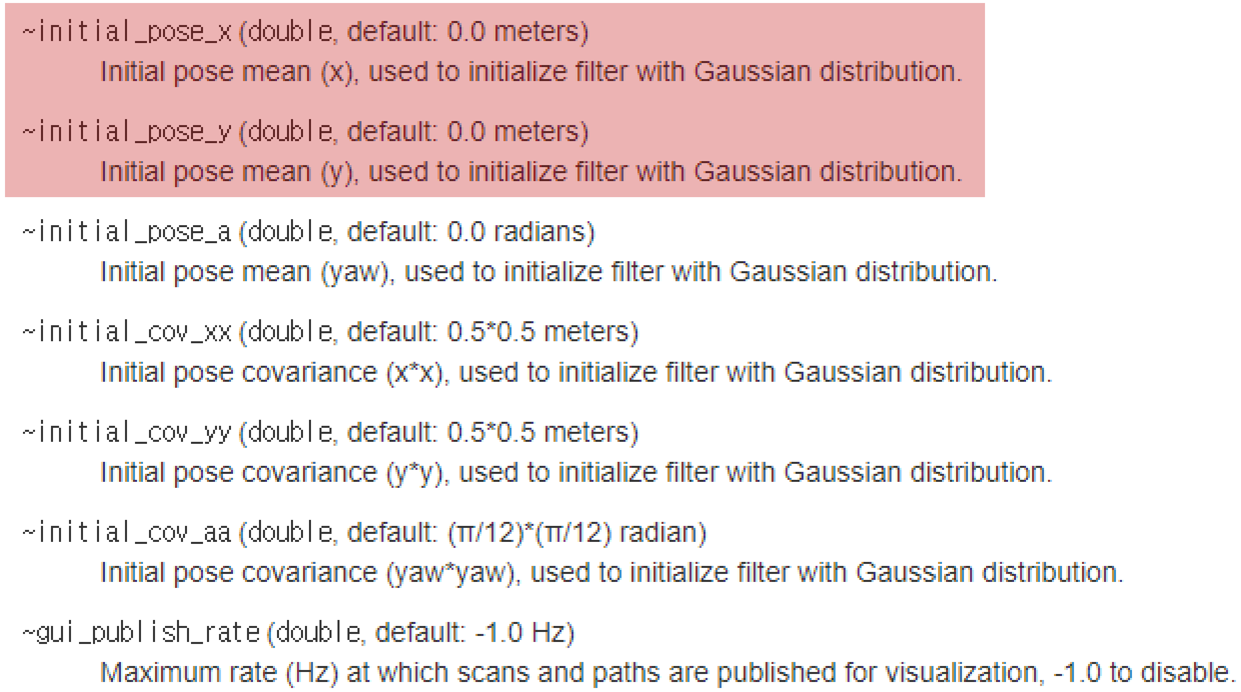
~resample_interval (int, default: 2)

Number of filter updates required before resampling.

~transform_tolerance (double, default: 0.1 seconds)

Time with which to post-date the transform that is published, to indicate that this transform is valid into the future.

- 노드: amcl (계속)
  - 매개변수



~initial_pose_x (double, default: 0.0 meters)

Initial pose mean (x), used to initialize filter with Gaussian distribution.

~initial_pose_y (double, default: 0.0 meters)

Initial pose mean (y), used to initialize filter with Gaussian distribution.

~initial_pose_a (double, default: 0.0 radians)

Initial pose mean (yaw), used to initialize filter with Gaussian distribution.

~initial_cov_xx (double, default: 0.5*0.5 meters)

Initial pose covariance (x*x), used to initialize filter with Gaussian distribution.

~initial_cov_yy (double, default: 0.5*0.5 meters)

Initial pose covariance (y*y), used to initialize filter with Gaussian distribution.

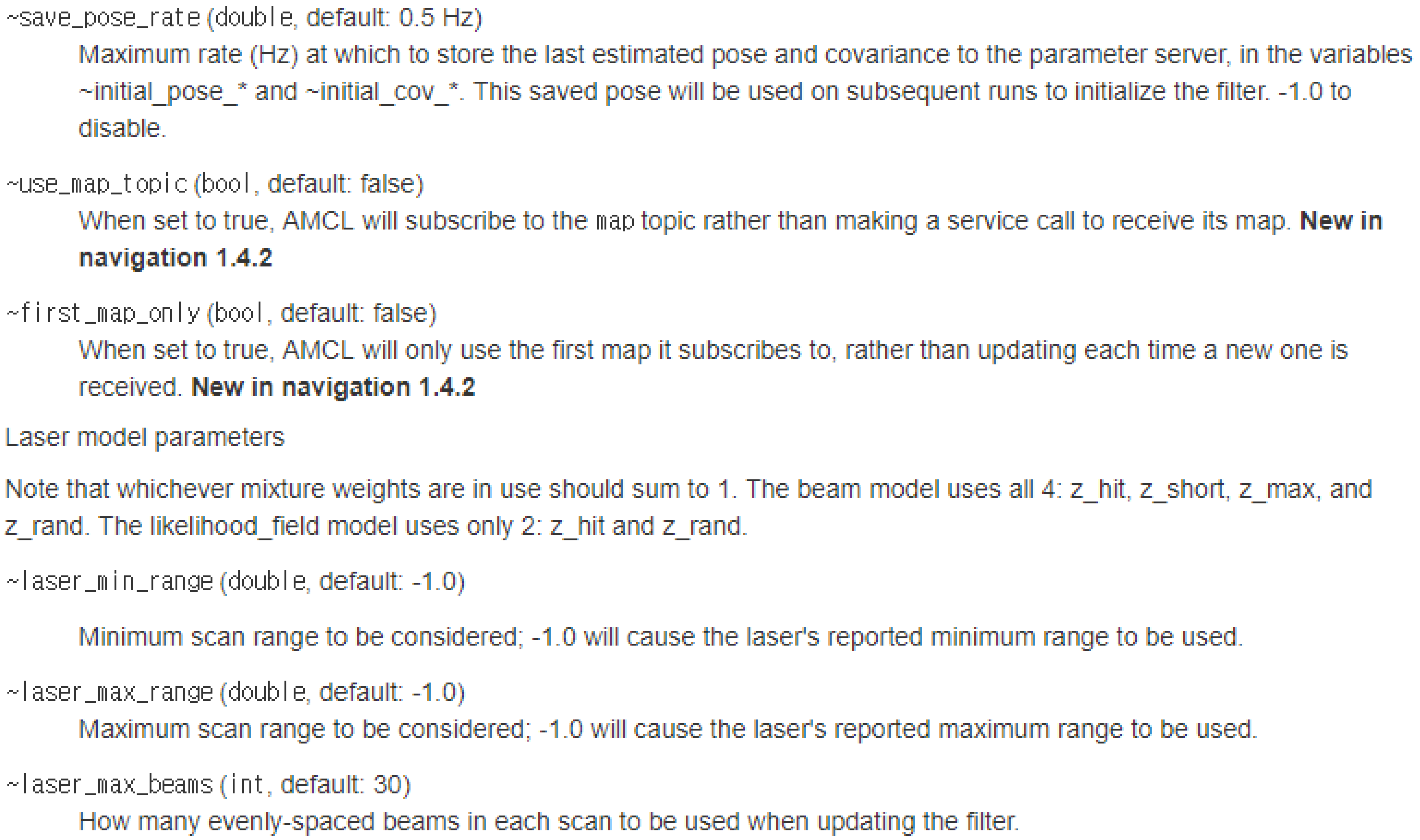
~initial_cov_aa (double, default: (π/12)*(π/12) radian)

Initial pose covariance (yaw*yaw), used to initialize filter with Gaussian distribution.

~gui_publish_rate (double, default: -1.0 Hz)

Maximum rate (Hz) at which scans and paths are published for visualization, -1.0 to disable.

- 노드: amcl (계속)
  - 매개변수



~save_pose_rate (double, default: 0.5 Hz)

Maximum rate (Hz) at which to store the last estimated pose and covariance to the parameter server, in the variables ~initial_pose_* and ~initial_cov_*. This saved pose will be used on subsequent runs to initialize the filter. -1.0 to disable.

~use_map_topic (bool, default: false)

When set to true, AMCL will subscribe to the map topic rather than making a service call to receive its map. **New in navigation 1.4.2**

~first_map_only (bool, default: false)

When set to true, AMCL will only use the first map it subscribes to, rather than updating each time a new one is received. **New in navigation 1.4.2**

Laser model parameters

Note that whichever mixture weights are in use should sum to 1. The beam model uses all 4: z_hit, z_short, z_max, and z_rand. The likelihood_field model uses only 2: z_hit and z_rand.

~laser_min_range (double, default: -1.0)

Minimum scan range to be considered; -1.0 will cause the laser's reported minimum range to be used.

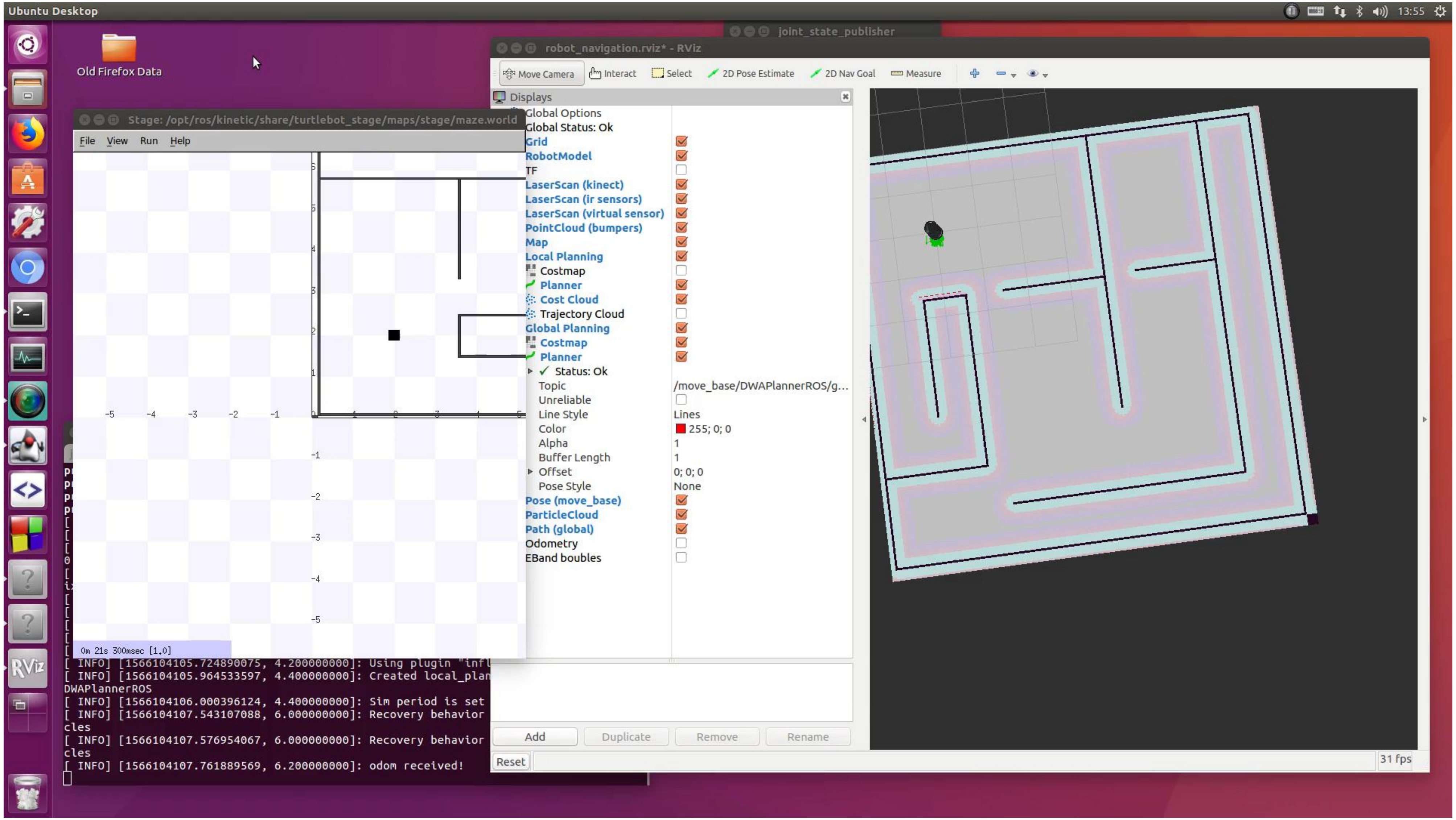
~laser_max_range (double, default: -1.0)

Maximum scan range to be considered; -1.0 will cause the laser's reported maximum range to be used.

~laser_max_beams (int, default: 30)

How many evenly-spaced beams in each scan to be used when updating the filter.

위치 추정 동작 사례



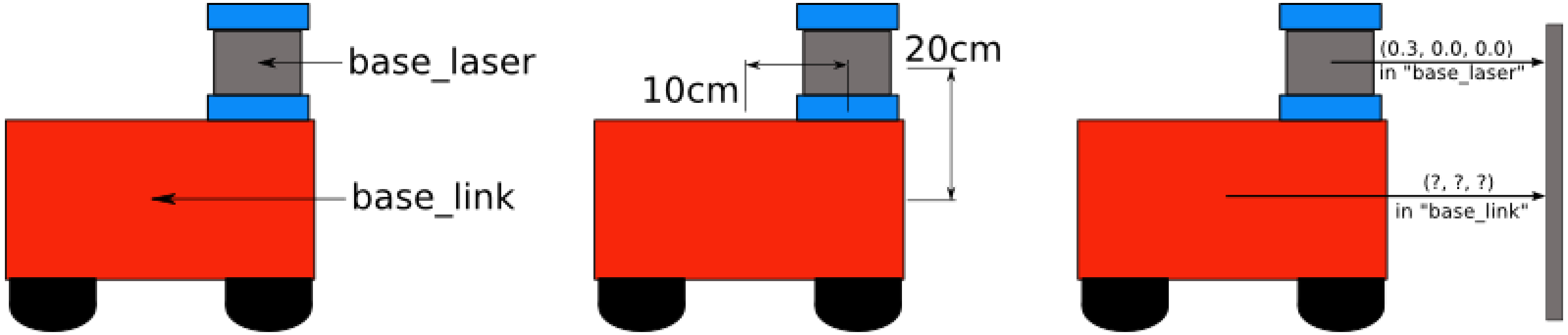
Displays

Global Options	
Global Status: Ok	
Grid	☑
RobotModel	☑
TF	□
LaserScan (kinect)	☑
LaserScan (ir sensors)	☑
LaserScan (virtual sensor)	☑
PointCloud (bumpers)	☑
Map	☑
Local Planning	☑
Costmap	□
Planner	☑
Cost Cloud	☑
Trajectory Cloud	□
Global Planning	☑
Costmap	☑
Planner	☑
✓ Status: Ok	
Topic	/move_base/DWAPlannerROS/g...
Unreliable	□
Line Style	Lines
Color	🟥 255; 0; 0
Alpha	1
Buffer Length	1
Offset	0; 0; 0
Pose Style	None
Pose (move_base)	☑
ParticleCloud	☑
Path (global)	☑
Odometry	□
EBand boubles	□

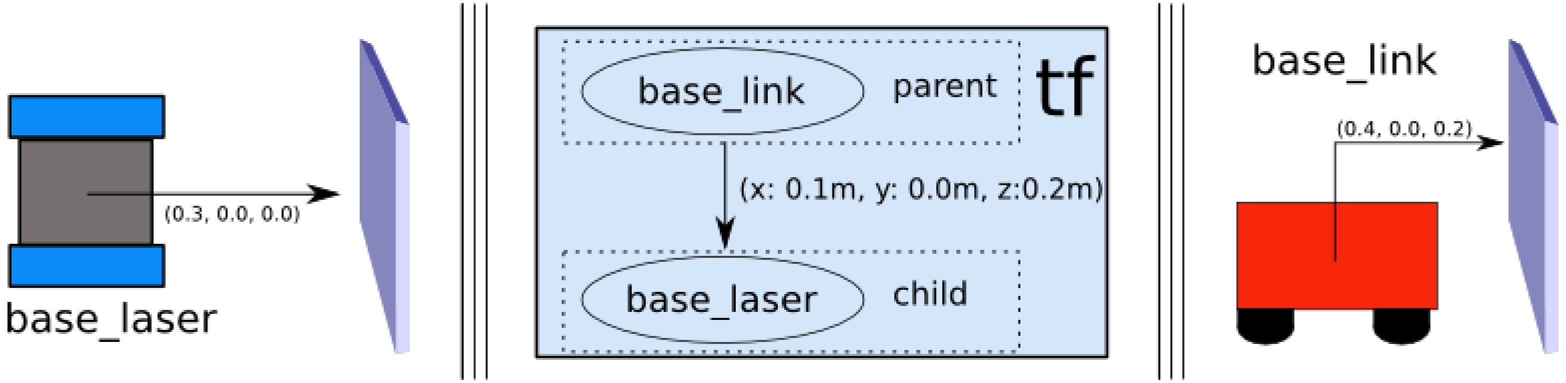
```
[ INFO] [1566104105.724890075, 4.200000000]: Using plugin "infl
[ INFO] [1566104105.964533597, 4.400000000]: Created local_plan
DWAPlannerROS
[ INFO] [1566104106.000396124, 4.400000000]: Sim period is set
[ INFO] [1566104107.543107088, 6.000000000]: Recovery behavior
cles
[ INFO] [1566104107.576954067, 6.000000000]: Recovery behavior
cles
[ INFO] [1566104107.761889569, 6.200000000]: odom received!
```

로봇의 tf 환경 설정

- URL: http://wiki.ros.org/navigation/Tutorials/RobotSetup/TF
- 가상 로봇 구조



- tf 설정



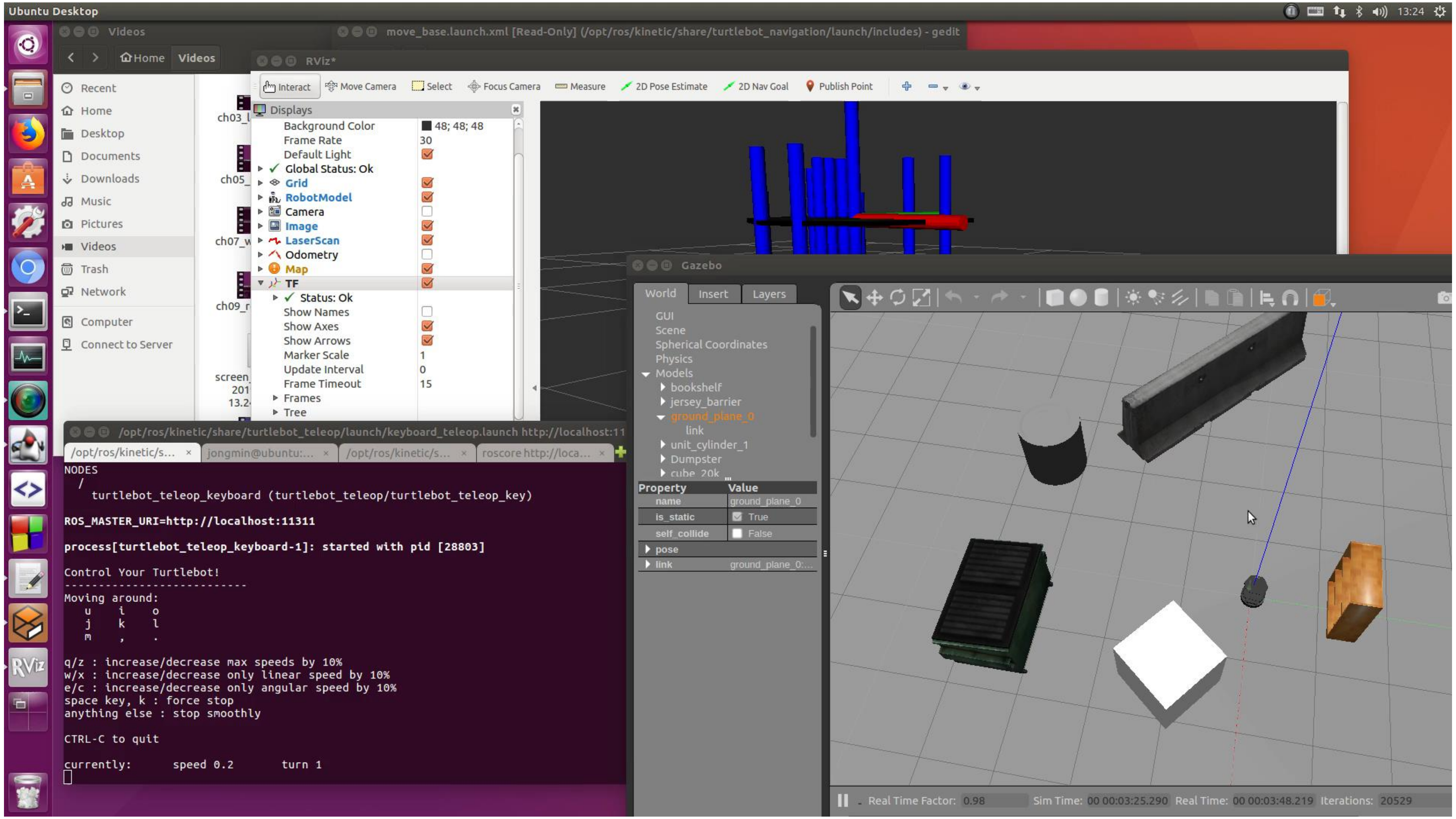
tf (transformation) 패키지

- URL: http://wiki.ros.org/tf
- 여러 개의 좌표 프레임을 추적할 수 있게 해주는 패키지
- 색상 의미
  - 빨간 색: x축
  - 연두 색: y축
  - 파란 색: z축

- tf 보기

```
$ rosrun tf view_frames && evince frames.pdf
```

터틀봇 tf



Displays

Background Color	48; 48; 48
Frame Rate	30
Default Light	☑
✓ Global Status: Ok	
Grid	☑
RobotModel	☑
Camera	□
Image	☑
LaserScan	☑
Odometry	□
Map	☑
TF	☑
✓ Status: Ok	
Show Names	□
Show Axes	☑
Show Arrows	☑
Marker Scale	1
Update Interval	0
Frame Timeout	15
Frames	
Tree	

Gazebo — World

- GUI
- Scene
- Spherical Coordinates
- Physics
- Models
  - bookshelf
  - jersey_barrier
  - ground_plane_0
    - link
  - unit_cylinder_1
  - Dumpster
  - cube_20k

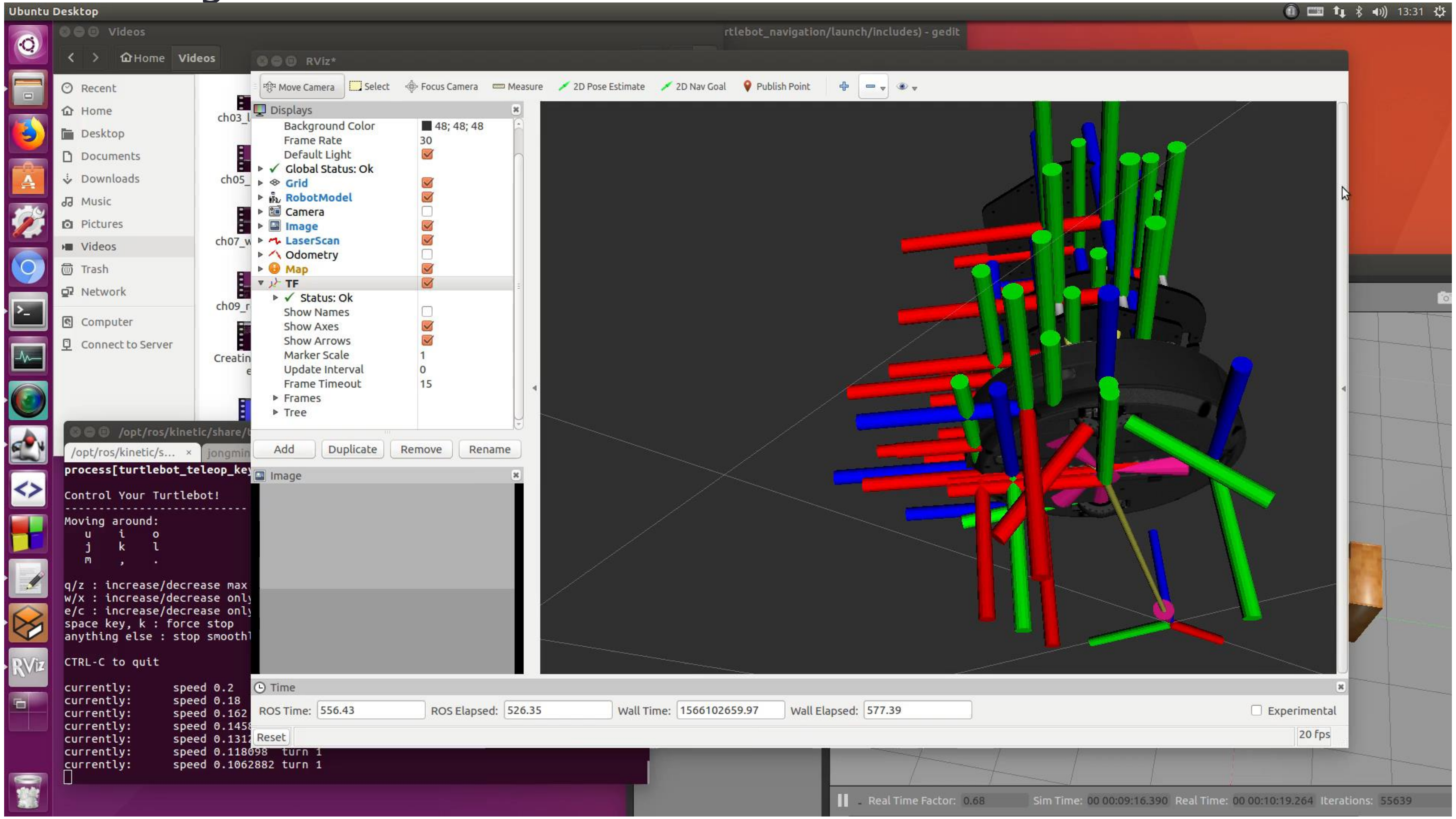
Property	Value
name	ground_plane_0
is_static	☑ True
self_collide	□ False
pose	
link	ground_plane_0:...

Real Time Factor: 0.98 Sim Time: 00 00:03:25.290 Real Time: 00 00:03:48.219 Iterations: 20529

```
NODES
  /
    turtlebot_teleop_keyboard (turtlebot_teleop/turtlebot_teleop_key)

ROS_MASTER_URI=http://localhost:11311

process[turtlebot_teleop_keyboard-1]: started with pid [28803]

Control Your Turtlebot!
---------------------------
Moving around:
   u    i    o
   j    k    l
   m    ,    .

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
space key, k : force stop
anything else : stop smoothly

CTRL-C to quit

currently:      speed 0.2       turn 1
```

터틀봇 바퀴 tf

- 바퀴의 angular.z 값을 조정하여 직진, 후진, 회전 움직임 생성



Background Color 48; 48; 48
Frame Rate 30
Default Light ☑
✓ Global Status: Ok
Grid ☑
RobotModel ☑
Camera □
Image ☑
LaserScan ☑
Odometry □
Map ☑
TF ☑
✓ Status: Ok
Show Names □
Show Axes ☑
Show Arrows ☑
Marker Scale 1
Update Interval 0
Frame Timeout 15
Frames
Tree

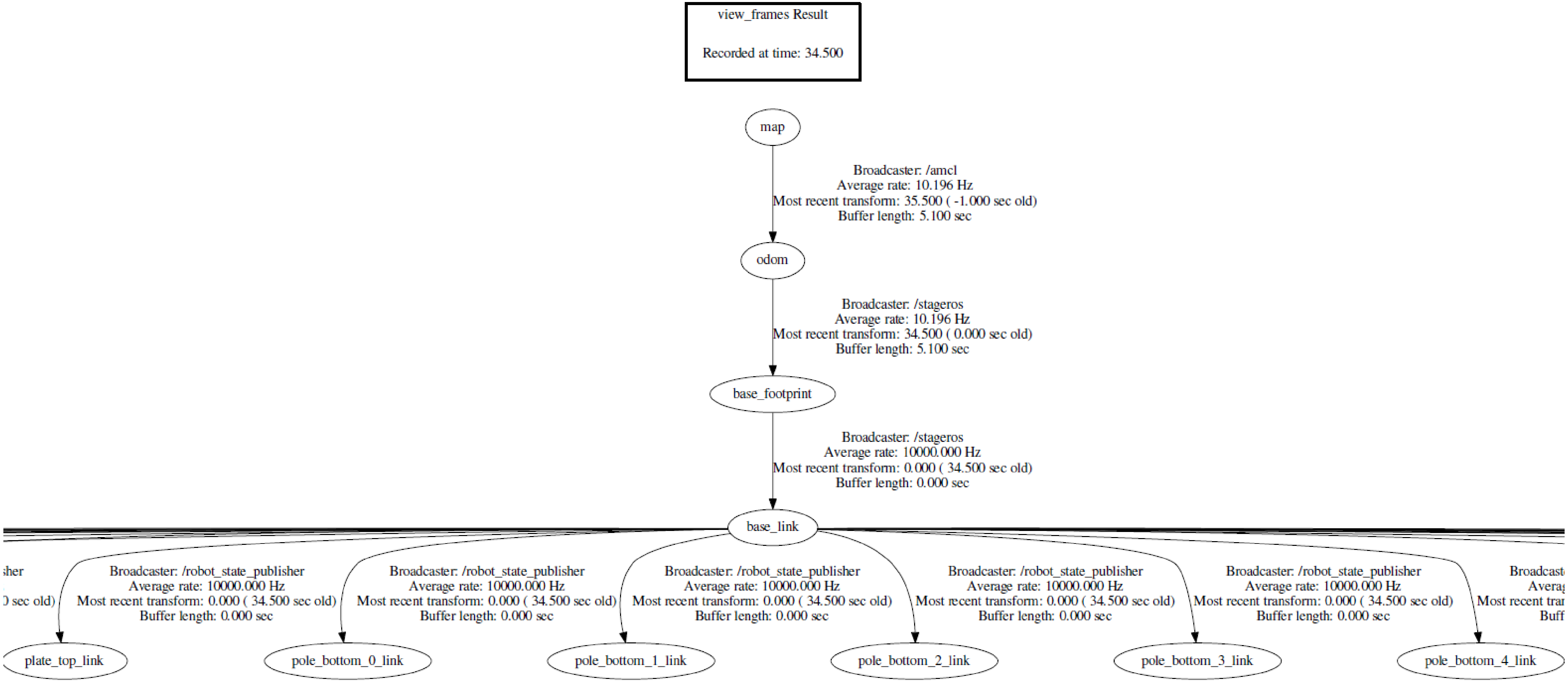
ROS Time: 556.43 ROS Elapsed: 526.35 Wall Time: 1566102659.97 Wall Elapsed: 577.39 □ Experimental

20 fps

Real Time Factor: 0.68 Sim Time: 00 00:09:16.390 Real Time: 00 00:10:19.264 Iterations: 55639

```
process[turtlebot_teleop_key

Control Your Turtlebot!
---------------------------
Moving around:
   u    i    o
   j    k    l
   m    ,    .

q/z : increase/decrease max
w/x : increase/decrease only
e/c : increase/decrease only
space key, k : force stop
anything else : stop smoothl

CTRL-C to quit

currently:      speed 0.2
currently:      speed 0.18
currently:      speed 0.162
currently:      speed 0.1458
currently:      speed 0.1312
currently:      speed 0.118098  turn 1
currently:      speed 0.1062882 turn 1
```

turtlebot_in_stage.launch 실행 후 view_frames



view_frames Result

Recorded at time: 34.500

Broadcaster: /amcl
Average rate: 10.196 Hz
Most recent transform: 35.500 (-1.000 sec old)
Buffer length: 5.100 sec

Broadcaster: /stageros
Average rate: 10.196 Hz
Most recent transform: 34.500 (0.000 sec old)
Buffer length: 5.100 sec

Broadcaster: /stageros
Average rate: 10000.000 Hz
Most recent transform: 0.000 (34.500 sec old)
Buffer length: 0.000 sec

sher
0 sec old)

Broadcaster: /robot_state_publisher
Average rate: 10000.000 Hz
Most recent transform: 0.000 (34.500 sec old)
Buffer length: 0.000 sec

Broadcaster: /robot_state_publisher
Average rate: 10000.000 Hz
Most recent transform: 0.000 (34.500 sec old)
Buffer length: 0.000 sec

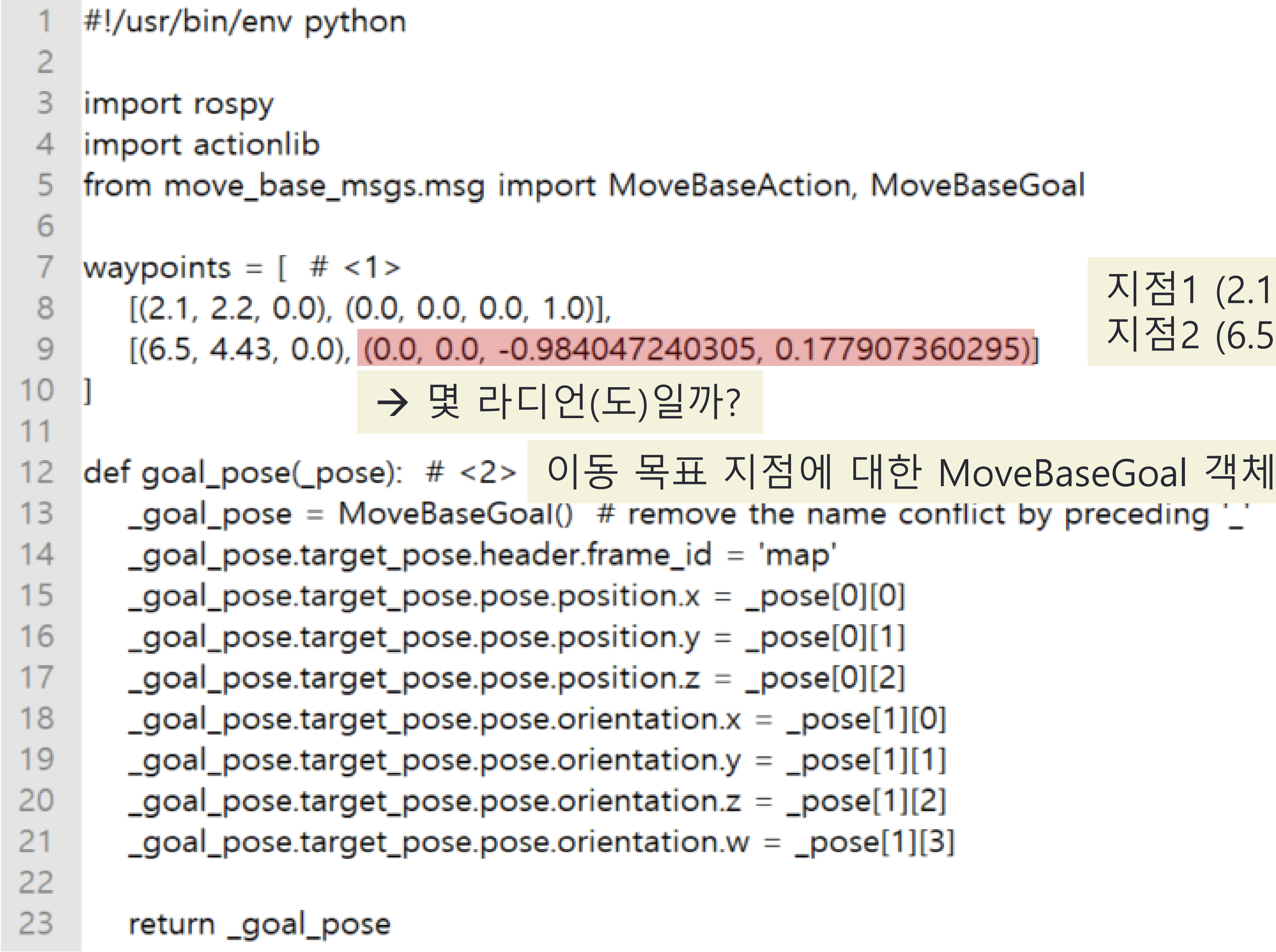
Broadcaster: /robot_state_publisher
Average rate: 10000.000 Hz
Most recent transform: 0.000 (34.500 sec old)
Buffer length: 0.000 sec

Broadcaster: /robot_state_publisher
Average rate: 10000.000 Hz
Most recent transform: 0.000 (34.500 sec old)
Buffer length: 0.000 sec

Broadcaster: /robot_state_publisher
Average rate: 10000.000 Hz
Most recent transform: 0.000 (34.500 sec old)
Buffer length: 0.000 sec

Broadcast
Averag
Most recent trai
Buff

patrol.py

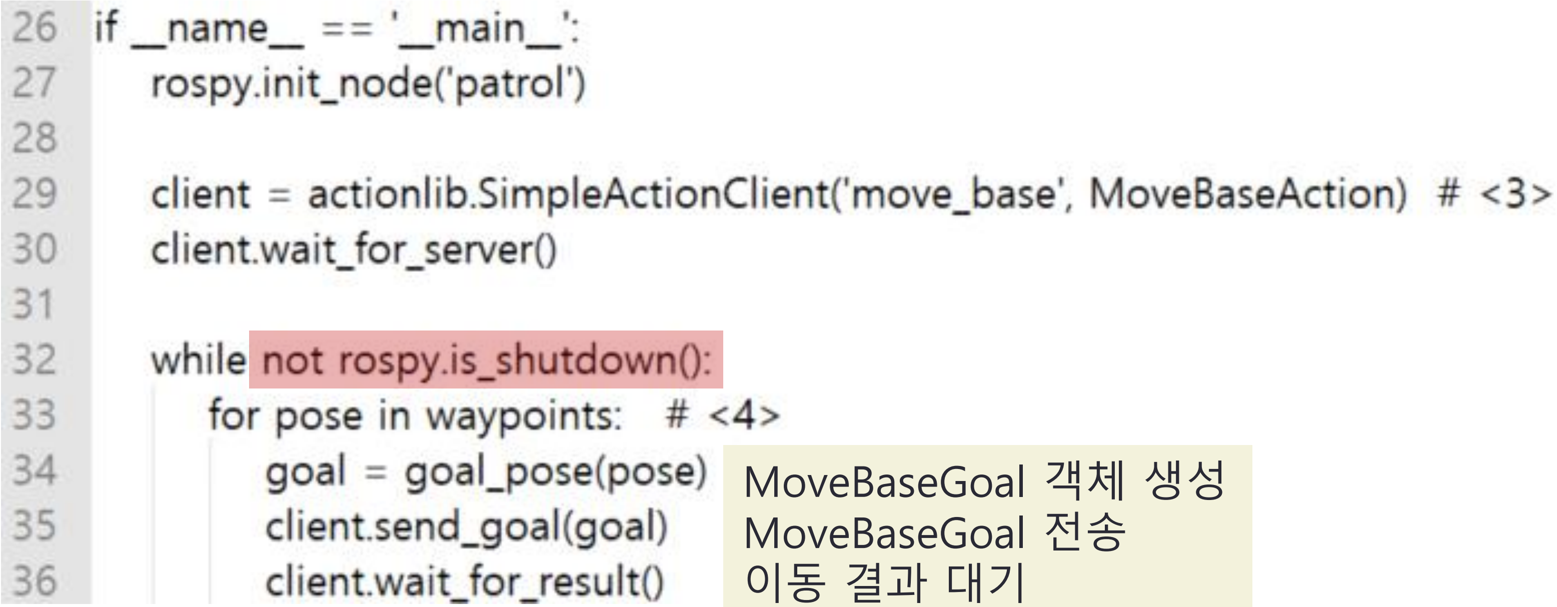


```
#!/usr/bin/env python

import rospy
import actionlib
from move_base_msgs.msg import MoveBaseAction, MoveBaseGoal

waypoints = [  # <1>
    [(2.1, 2.2, 0.0), (0.0, 0.0, 0.0, 1.0)],
    [(6.5, 4.43, 0.0), (0.0, 0.0, -0.984047240305, 0.177907360295)]
]

def goal_pose(_pose):  # <2>
    _goal_pose = MoveBaseGoal()  # remove the name conflict by preceding '_'
    _goal_pose.target_pose.header.frame_id = 'map'
    _goal_pose.target_pose.pose.position.x = _pose[0][0]
    _goal_pose.target_pose.pose.position.y = _pose[0][1]
    _goal_pose.target_pose.pose.position.z = _pose[0][2]
    _goal_pose.target_pose.pose.orientation.x = _pose[1][0]
    _goal_pose.target_pose.pose.orientation.y = _pose[1][1]
    _goal_pose.target_pose.pose.orientation.z = _pose[1][2]
    _goal_pose.target_pose.pose.orientation.w = _pose[1][3]

    return _goal_pose
```

지점1 (2.1, 2.2)
지점2 (6.5, 4.43)

→ 몇 라디언(도)일까?

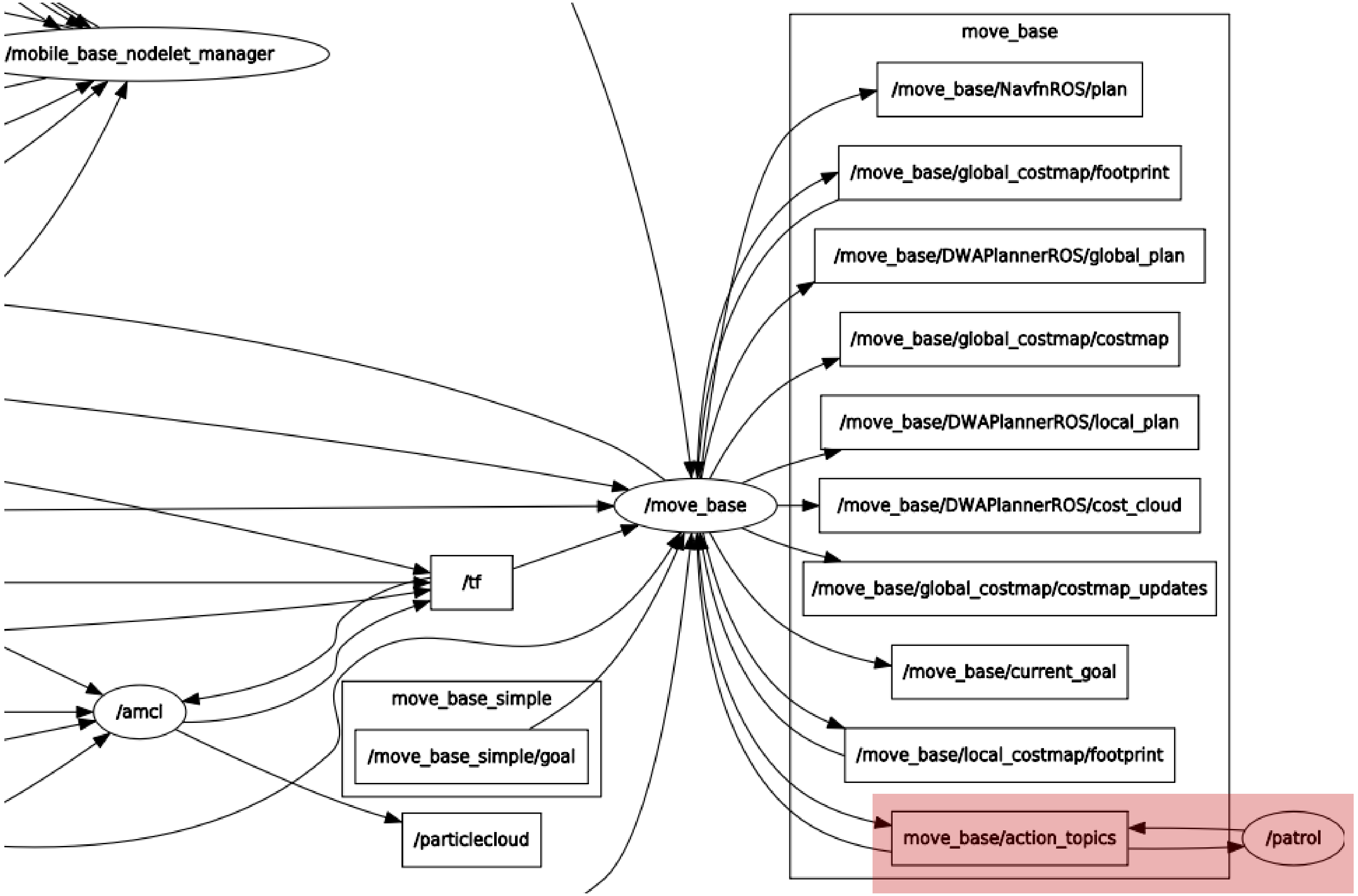
이동 목표 지점에 대한 MoveBaseGoal 객체 반환 함수 (5장 강의 자료 pp.26~ 참조)



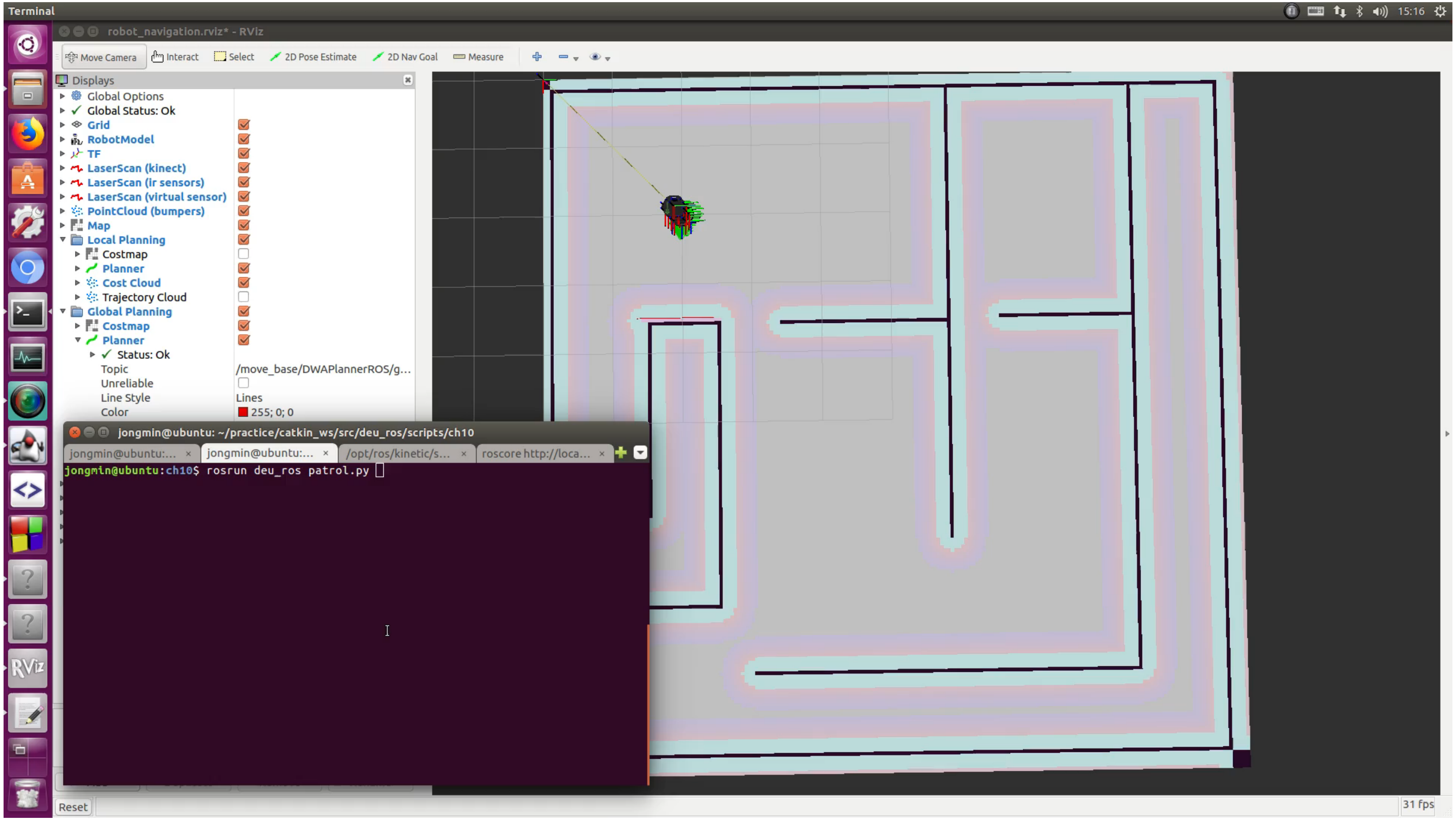
```
if __name__ == '__main__':
    rospy.init_node('patrol')

    client = actionlib.SimpleActionClient('move_base', MoveBaseAction)  # <3>
    client.wait_for_server()

    while not rospy.is_shutdown():
        for pose in waypoints:   # <4>
            goal = goal_pose(pose)
            client.send_goal(goal)
            client.wait_for_result()
```

MoveBaseGoal 객체 생성
MoveBaseGoal 전송
이동 결과 대기

ROS 그래프: patrol.py


/mobile_base_nodelet_manager
move_base
/move_base/NavfnROS/plan
/move_base/global_costmap/footprint
/move_base/DWAPlannerROS/global_plan
/move_base/global_costmap/costmap
/move_base/DWAPlannerROS/local_plan
/move_base
/move_base/DWAPlannerROS/cost_cloud
/tf
/move_base/global_costmap/costmap_updates
/move_base/current_goal
/amcl
move_base_simple
/move_base_simple/goal
/move_base/local_costmap/footprint
/particlecloud
move_base/action_topics
/patrol

실행 결과



Topic /move_base/DWAPlannerROS/g...
Unreliable ☐
Line Style Lines
Color 🟥 255; 0; 0

```
jongmin@ubuntu:ch10$ rosrun deu_ros patrol.py
```