Spring 2014-15
SBASSE, LUMS

EE565: Mobile Robotics

Lab # 11: Setup and perform Robot navigation using ROS navigation stack

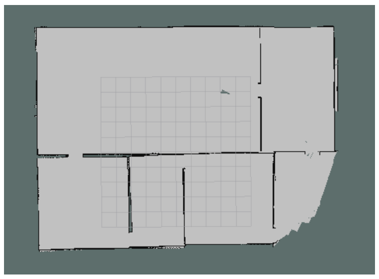
Description

In this lab, we will learn the ROS Navigation stack and use it to localize, come up with a global motion plan and move the robot on a desired path from a start point to an end point given the map of the environment. This is an essential tool to learn in navigation related tasks, understanding of which is easy and useful once you have grip over ROS.

Lab Work

The lab is divided into three components which you will be performing step-by-step. Once, these three are performed independently, a whole package with everything simulatenously running will be performed.

Map Building:



1. Open up the Lab 9 handout, and follow the simulation component instructions throughout to make a map of the “Lab9Maze” using the “my robot” gazebo models. Your map should be visualized in rviz like shown.
2. Using the “map_server” ROS package, save this map:

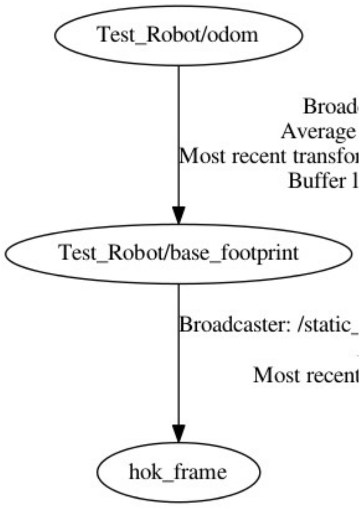
`rosrun map_server map_saver –f lab9maze`

3. Now close the gmapping node, and without closing other processes, run the saved map using:

`rosrun map_server map_server lab9maze.yaml`

(In case you run out of time, you may use somebody else's map)

Localize using Adaptive Monte Carlo Localization *(AMCL):*



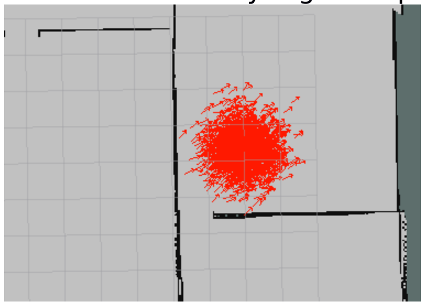
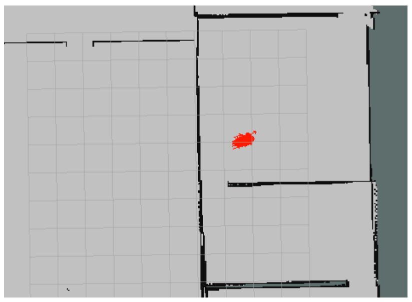
4. Note that at this stage, the following topics are running: /Test_Robot/cmd_vel, /map, /scan, /tf, /initialpose.
5. Make sure that all required transforms are being published on the /tf topic, using `rosrun tf view_frames`
6. Before running amcl, set the following parameters (using `rosparam get` or `rqt`):

```
/amcl/base_frame_id:=/Test_Robot/base_footprint
/amcl/global_frame_id:=/map
/amcl/odom_frame_id:=/Test_Robot/odom
/amcl/use_map_topic:=true
```

7. Now launch the amcl ROS Package for localization of your robot in the saved map.

```
roslaunch amcl amcl_diff.launch
```

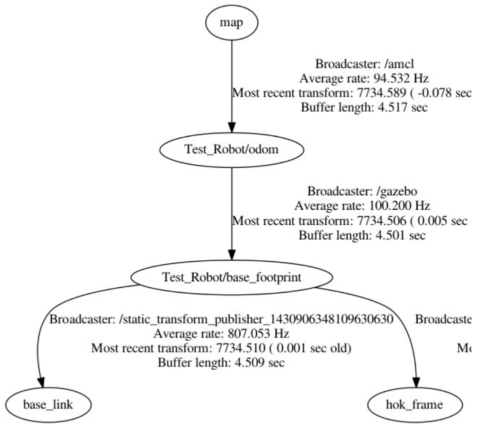
8. Once launched, you can give a pose estimate of your robot using the 2D Pose Estimate button in rviz. /particlecloud topic will be published that tells the pose estimates of the filter. View it in rviz. If you teleop the robot so that it can have better understanding of the environment then this estimate will gradually converge from a large scatter to a more tiny region depicting accuracy in estimate (shown).

Global Path Planner:

9. After the localization part, publish a static transform between the “Test_Robot/base_footprint” and “base_link”. Ensure the /tf frames as shown.

```
rosrun tf static_transform_publisher 0 0 0 0 0 0 Test_Robot/base_footprint base_link 0.001
```



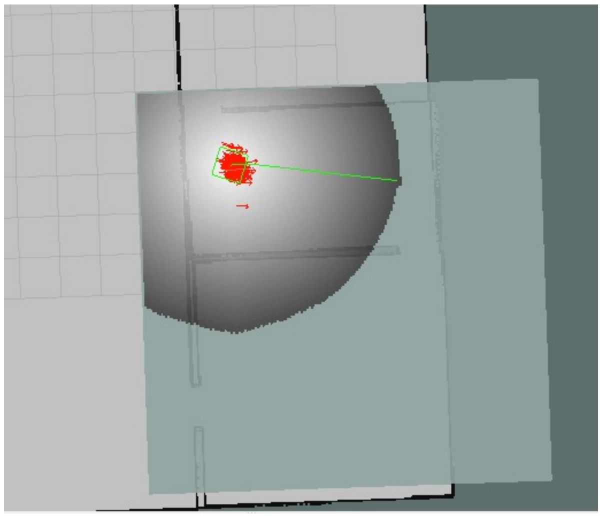
10. Launch the global planner node:

```
rosrun global_planner planner /global_planner/goal:= /move_base_simple/goal
```

11. If the footprint error pops up then open `rqt`, and set the footprint in Dynamic Configuration of “costmap” having the value:

```
[[0.35,0.35],[-0.35,0.35],[-0.35,-0.35],[0.35,-0.35]]
```

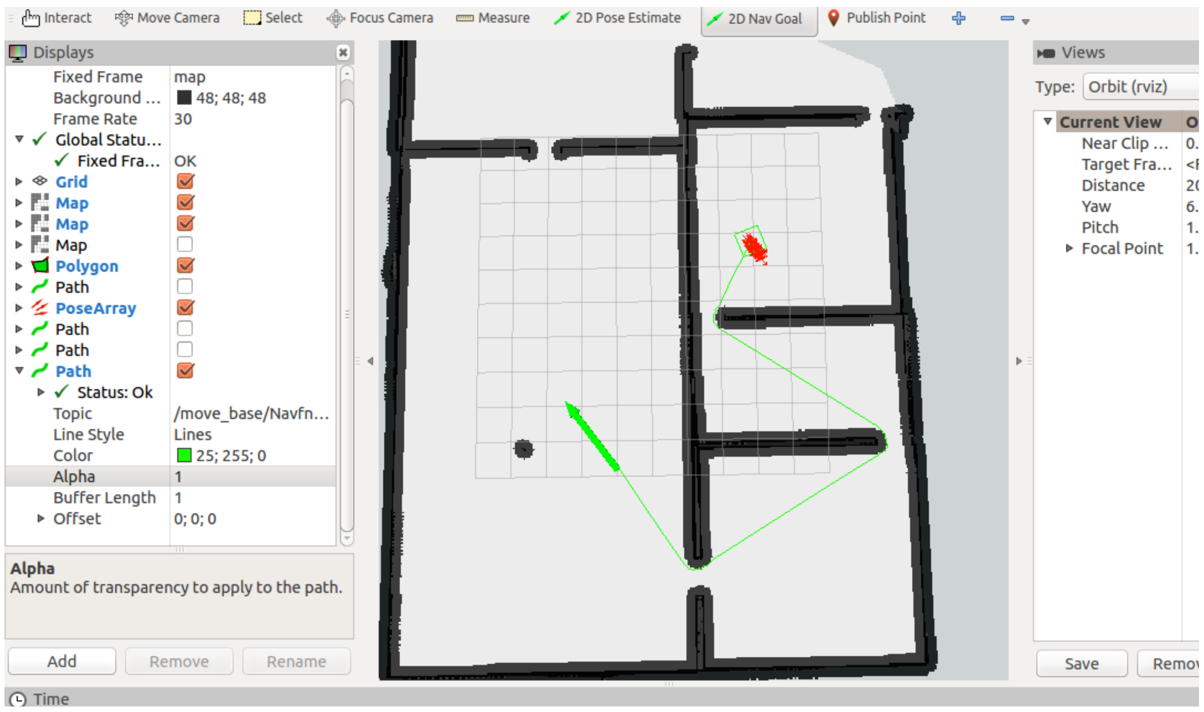
12. Keep running the global planner node, and visualize the published topics in rviz (planner/plan, costmap/costmap).



13. Position your robot within the costmap by teleoping there. Once within the costmap, and with the amcl localization running in the background, you can give a goal pose to the global planner (using rviz's 2D Nav Goal button) which would in turn output a path from current pose to the goal pose (shown).
14. As shown, the green box is footprint of your robot, red poses are the pose estimates, and the gray box overlayed on the map is the costmap. Finally the green line is the planned path.
15. Take screenshot of your path and get it checked.

RUNNING EVERYTHING WITH MOVE BASE*:*

1. You may close down the amcl and global planner nodes.
2. Download the Lab 11 files from LMS.
3. Place the whole folder "my_robot_name_2dnav" in <catkin_ws>/src so that you have a package. Compile it.
4. Run the launch file "move_base.launch". Numerous topics will be available to view in rviz. See the global costmap, footprint and the global path. Now, if you give the robot a 2D Nav Goal using rviz's button, it will show you a path (like global path planner) and start publishing at the command velocity so that the robot traverses that path.
5. Your robot will not move unless the command velocity topic is rightly set. However you'd be able to see a global path.
6. Open up the launch file and remap the cmd_vel topic. If everything is set right, you'll see the robot moving on your planned path. You may also play around with other topics being published by these nodes.
7. Open up other yaml files, and try to understand what is happening.



Fixed Frame	map
Background ...	■ 48; 48; 48
Frame Rate	30
✓ Global Statu...	
✓ Fixed Fra...	OK
Grid	✓
Map	✓
Map	✓
Map	
Polygon	✓
Path	
PoseArray	✓
Path	
Path	
Path	✓
✓ Status: Ok	
Topic	/move_base/Navfn...
Line Style	Lines
Color	🟩 25; 255; 0
Alpha	1
Buffer Length	1
Offset	0; 0; 0

Alpha
Amount of transparency to apply to the path.

Type: Orbit (rviz)

Current View	O
Near Clip ...	0.
Target Fra...	<f
Distance	2(
Yaw	6.
Pitch	1.
Focal Point	1.