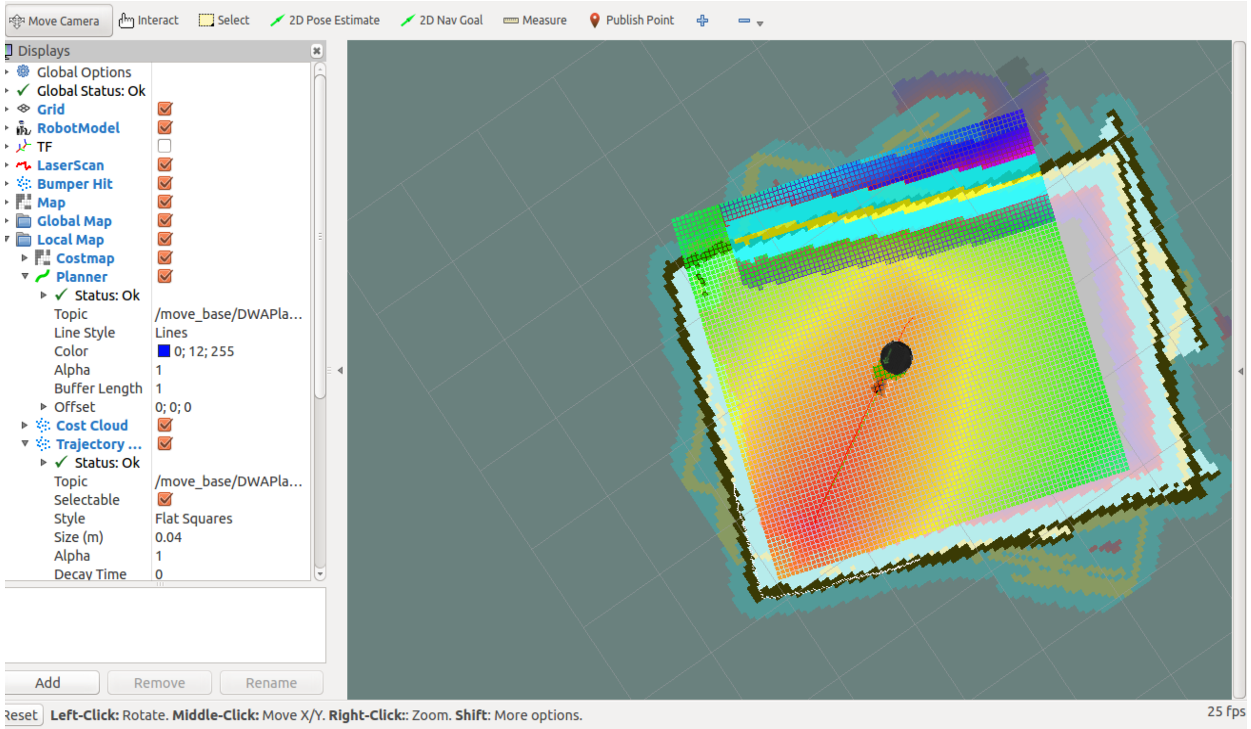
Tutorial 3- AMCL

MIE 443 – Aaron Hao Tan

Developed By Christopher Thompson, Daniel Dworakowski and Richard Hu

Introduction to AMCL

- AMCL (Adaptive Monte Carlo Localization)
- Algorithm for robots to localize in an environment given a map using a particle filter.
- AMCL allows the robot to estimate position and orientation, facilitating path planning and navigation functionality.

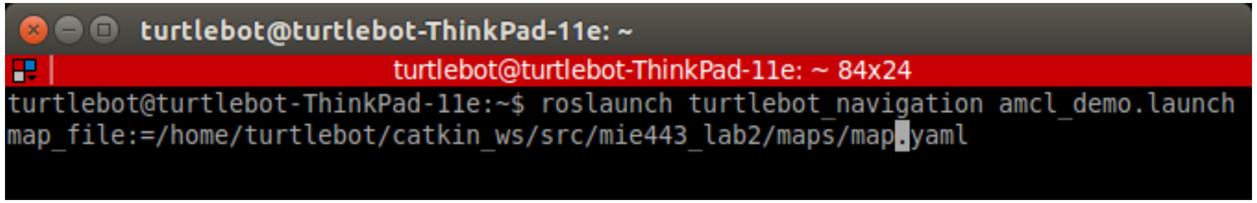


Global Options	
✓ Global Status: Ok	
Grid	☑
RobotModel	☑
TF	□
LaserScan	☑
Bumper Hit	☑
Map	☑
Global Map	☑
Local Map	☑
Costmap	☑
Planner	☑
✓ Status: Ok	
Topic	/move_base/DWAPla...
Line Style	Lines
Color	🟦 0; 12; 255
Alpha	1
Buffer Length	1
Offset	0; 0; 0
Cost Cloud	☑
Trajectory ...	☑
✓ Status: Ok	
Topic	/move_base/DWAPla...
Selectable	☑
Style	Flat Squares
Size (m)	0.04
Alpha	1
Decay Time	0

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Zoom. **Shift**: More options.

Running AMCL

- Run the following commands in their own terminal to launch the Simulated Turtlebot AMCL code to begin localizing and navigating within the provided map of the environment:
  - `roslaunch mie443_contest2 turtlebot_world.launch world:=1`
  - `roslaunch turtlebot_navigation amcl_demo.launch map_file:=/path/to/your/map_1.yaml`
  - launches the AMCL software with reference to the absolute map path that you provide as an argument:


turtlebot@turtlebot-ThinkPad-11e: ~
turtlebot@turtlebot-ThinkPad-11e: ~ 84x24

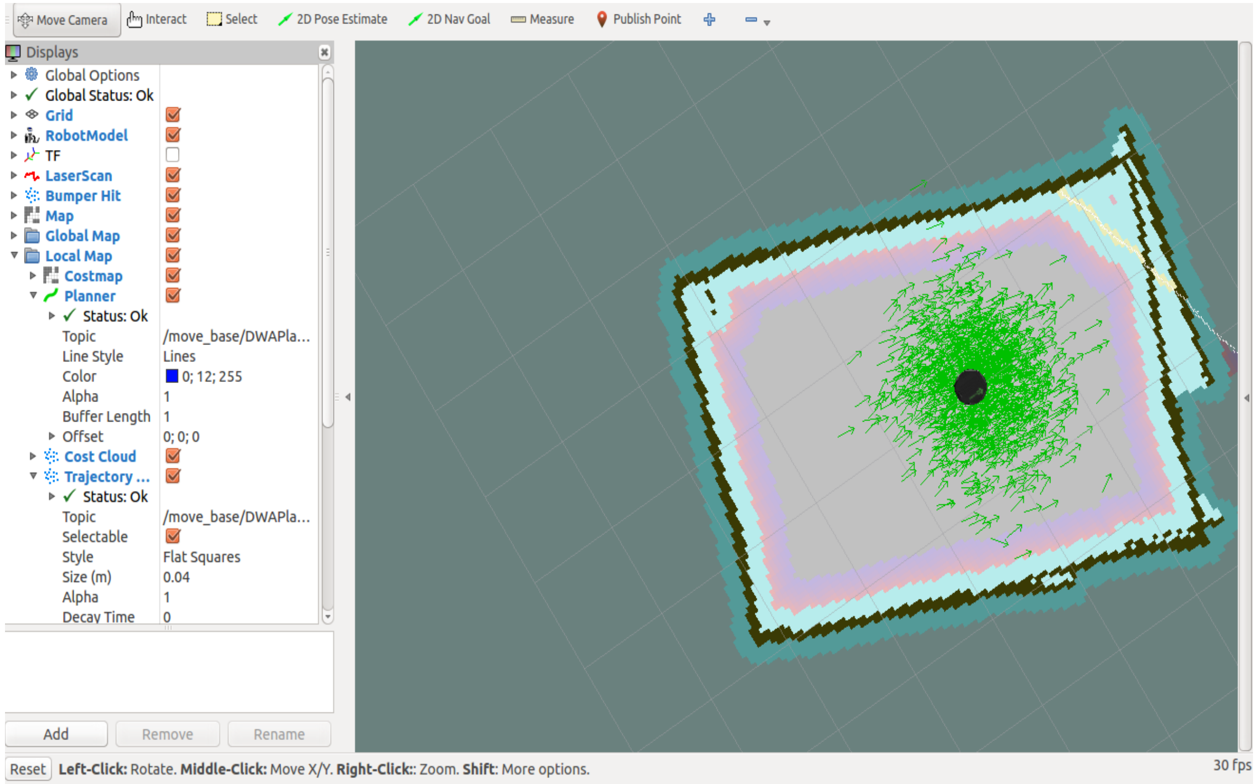
```
turtlebot@turtlebot-ThinkPad-11e:~$ roslaunch turtlebot_navigation amcl_demo.launch
map_file:=/home/turtlebot/catkin_ws/src/mie443_lab2/maps/map.yaml
```

Visualize Robot Location

- To visualize the AMCL location of the Simulated Turtlebot, Rviz is used again. Similar to gmapping, the following command is used to open Rviz:
  - `roslaunch turtlebot_rviz_launchers view_navigation.launch` – opens rviz with appropriate settings to visualize the robot

- The green arrows are the different particles that represent localization estimate from the filter.

- When AMCL launches there is always a large particle variance to try and encompass as many likely states as possible.

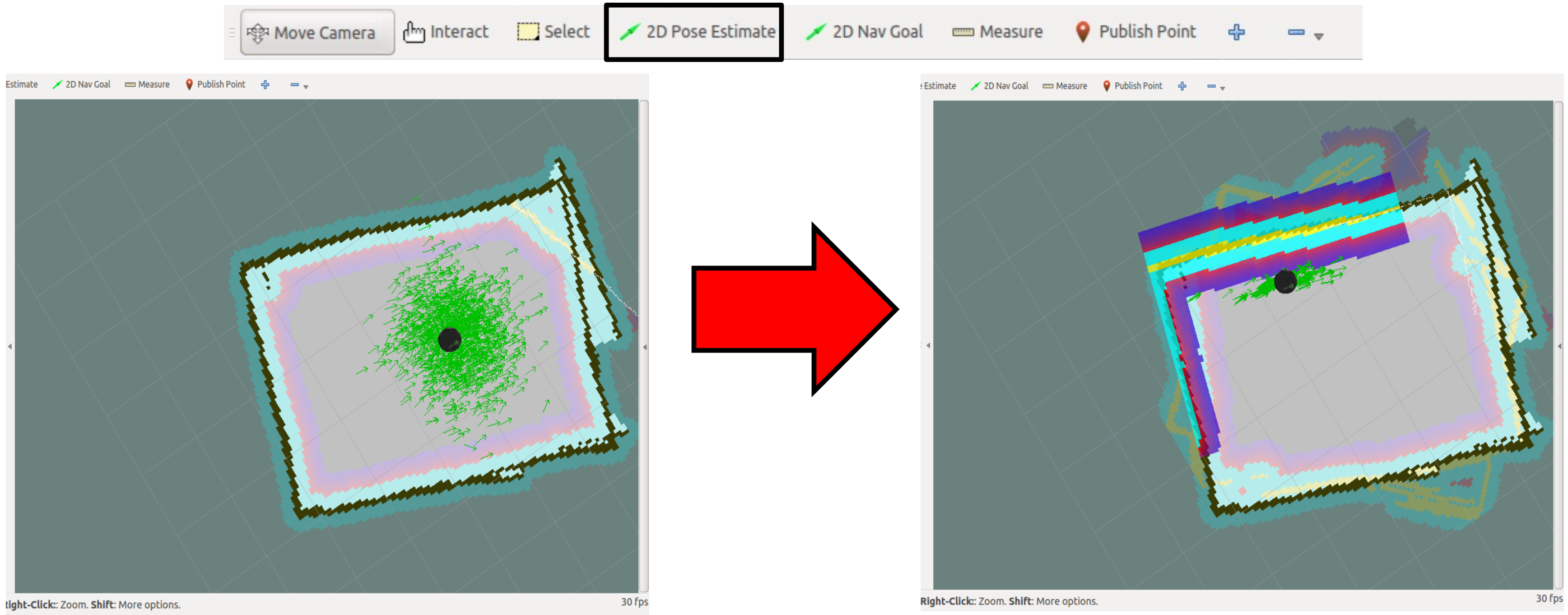


Global Options	
✓ Global Status: Ok	
Grid	☑
RobotModel	☑
TF	□
LaserScan	☑
Bumper Hit	☑
Map	☑
Global Map	☑
Local Map	☑
Costmap	☑
Planner	☑
✓ Status: Ok	
Topic	/move_base/DWAPla...
Line Style	Lines
Color	0; 12; 255
Alpha	1
Buffer Length	1
Offset	0; 0; 0
Cost Cloud	☑
Trajectory ...	☑
✓ Status: Ok	
Topic	/move_base/DWAPla...
Selectable	☑
Style	Flat Squares
Size (m)	0.04
Alpha	1
Decay Time	0

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click::** Zoom. **Shift:** More options.

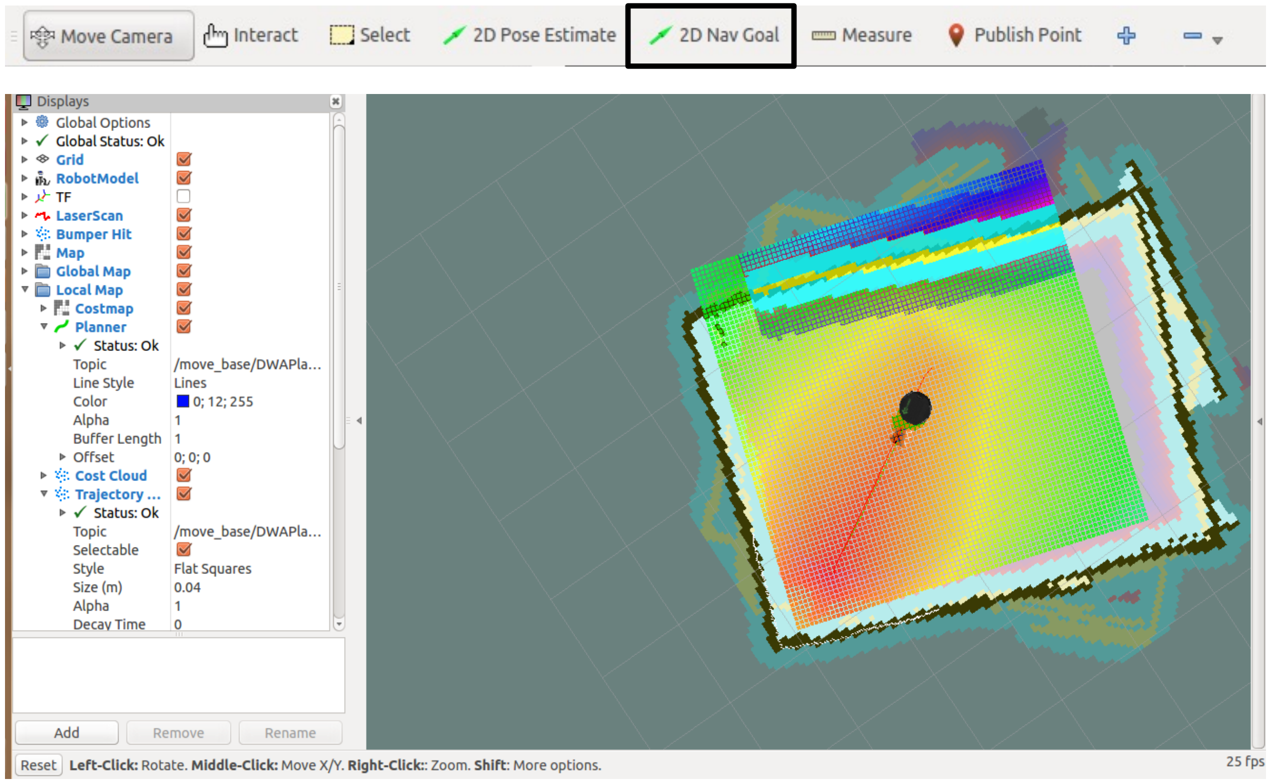
Filter Convergence

- Begin by clicking the “*2D Pose Estimate”* button in Rviz and select within the map to initialize the particle filter estimate.
- The particle filter algorithm mainly works as follows:
  - Each filter particle is compared to the simulated Turtlebot’s current sensor state and assigned a weight that is proportional to how well the particle and sensor state correspond to each other.
  - New particles are then spawned around the highest weighted particle.
  - The process is repeated indefinitely and eventually the particle state should converge to the true robot state.



Send Navigation Goals

- Navigation goals can be sent to the robot by clicking the “*2D Pose Estimate*” button in Rviz and then selecting a location within the map.
- The simulated robot will then attempt to plan and navigate a path to this goal while avoiding obstacles as you can see from the red line on the map.
- This command will fail if the simulated robot cannot find a valid path through open space.

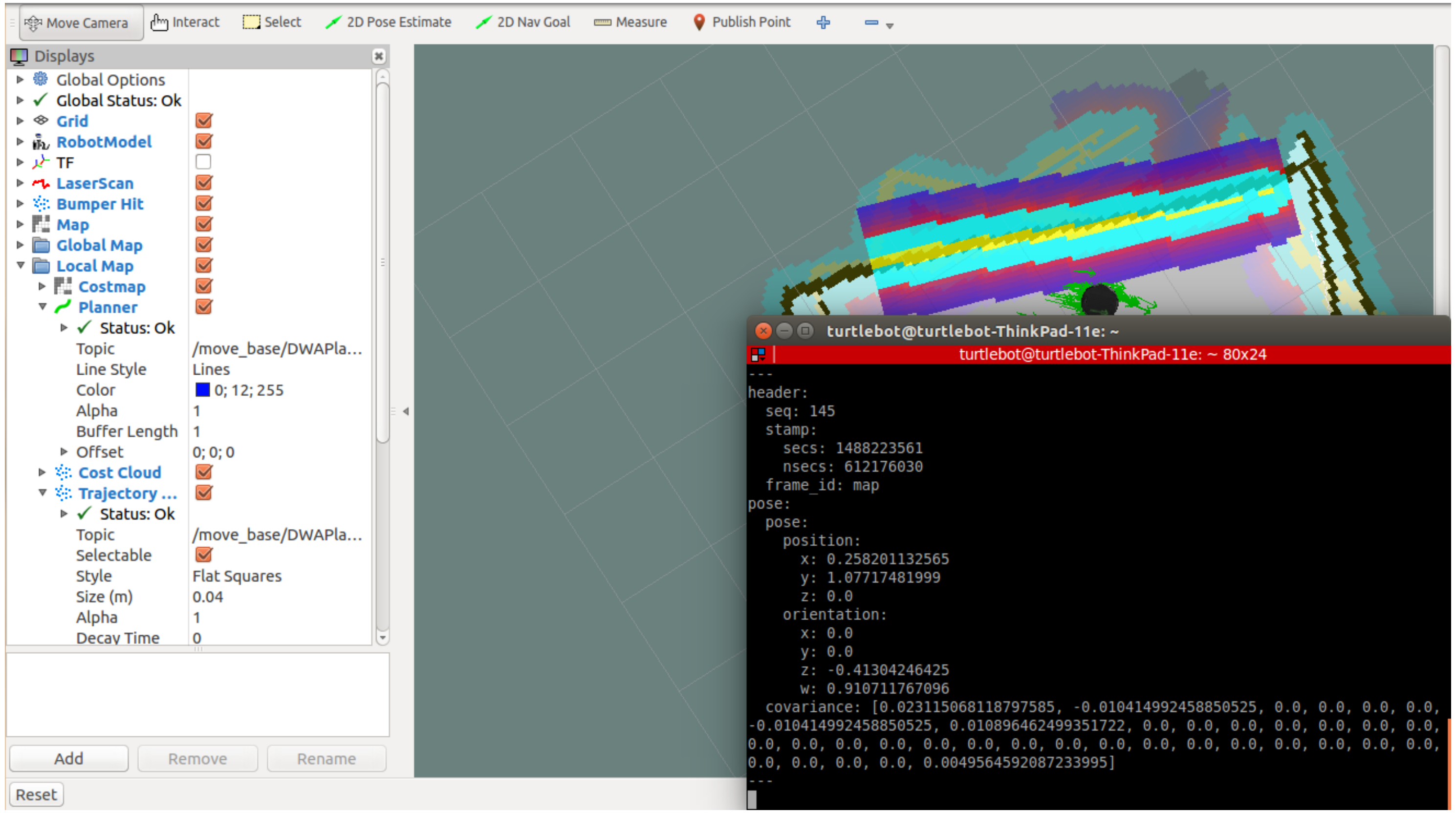


Displays
- Global Options
- ✓ Global Status: Ok
- Grid ☑
- RobotModel ☑
- TF □
- LaserScan ☑
- Bumper Hit ☑
- Map ☑
- Global Map ☑
- Local Map ☑
  - Costmap ☑
  - Planner ☑
    - ✓ Status: Ok
    - Topic: /move_base/DWAPla...
    - Line Style: Lines
    - Color: 0; 12; 255
    - Alpha: 1
    - Buffer Length: 1
    - Offset: 0; 0; 0
  - Cost Cloud ☑
  - Trajectory ... ☑
    - ✓ Status: Ok
    - Topic: /move_base/DWAPla...
    - Selectable: ☑
    - Style: Flat Squares
    - Size (m): 0.04
    - Alpha: 1
    - Decay Time: 0

Left-Click: Rotate. **Middle-Click**: Move X/Y. **Right-Click:**: Zoom. **Shift**: More options.

Print AMCL Estimate

- To view the AMCL pose estimate you can print the AMCL pose topic to the terminal using the following command in a new terminal:
  - `rostopic echo /amcl_pose` – prints the data being published on the amcl_pose topic

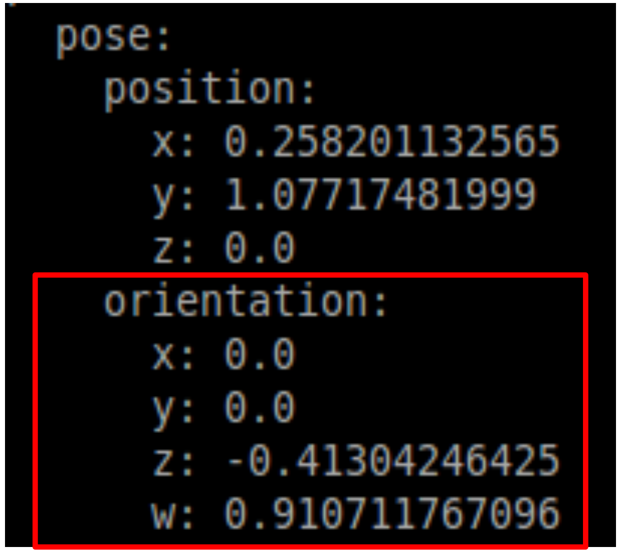


Global Options	
Global Status: Ok	
Grid	☑
RobotModel	☑
TF	□
LaserScan	☑
Bumper Hit	☑
Map	☑
Global Map	☑
Local Map	☑
Costmap	☑
Planner	☑
Status: Ok	
Topic	/move_base/DWAPla…
Line Style	Lines
Color	0; 12; 255
Alpha	1
Buffer Length	1
Offset	0; 0; 0
Cost Cloud	☑
Trajectory …	☑
Status: Ok	
Topic	/move_base/DWAPla…
Selectable	☑
Style	Flat Squares
Size (m)	0.04
Alpha	1
Decay Time	0

```
---
header:
  seq: 145
  stamp:
    secs: 1488223561
    nsecs: 612176030
  frame_id: map
pose:
  pose:
    position:
      x: 0.258201132565
      y: 1.07717481999
      z: 0.0
    orientation:
      x: 0.0
      y: 0.0
      z: -0.41304246425
      w: 0.910711767096
  covariance: [0.023115068118797585, -0.010414992458850525, 0.0, 0.0, 0.0, 0.0,
-0.010414992458850525, 0.010896462499351722, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0049564592087233995]
---
```

Introduction to Quaternions

- Intuitively, orientations are represented using Yaw, Pitch and Roll (YPR).
- However YPR can suffer from an issue called gimble lock, which is the loss of degrees of freedom when two or more axes align.
- Therefore, ROS represents orientations uses quaternions which is an alternative representation of an object's orientation using complex notation (ie. w + xi + yj + zk).
- Quaternions are out of the scope of this course and so the conversion between YPR and quaternions is handled in the contest code for you.
- Further resources about quaternions can be found at this link: http://www.cprogramming.com/tutorial/3d/quaternions.html



```
pose:
  position:
    x: 0.258201132565
    y: 1.07717481999
    z: 0.0
  orientation:
    x: 0.0
    y: 0.0
    z: -0.41304246425
    w: 0.910711767096
```

Questions