BSCH-AIR – Project

Name 1: Farrukh Jahangeer

Name 2: Khusmanda Devi Ramanjooloo

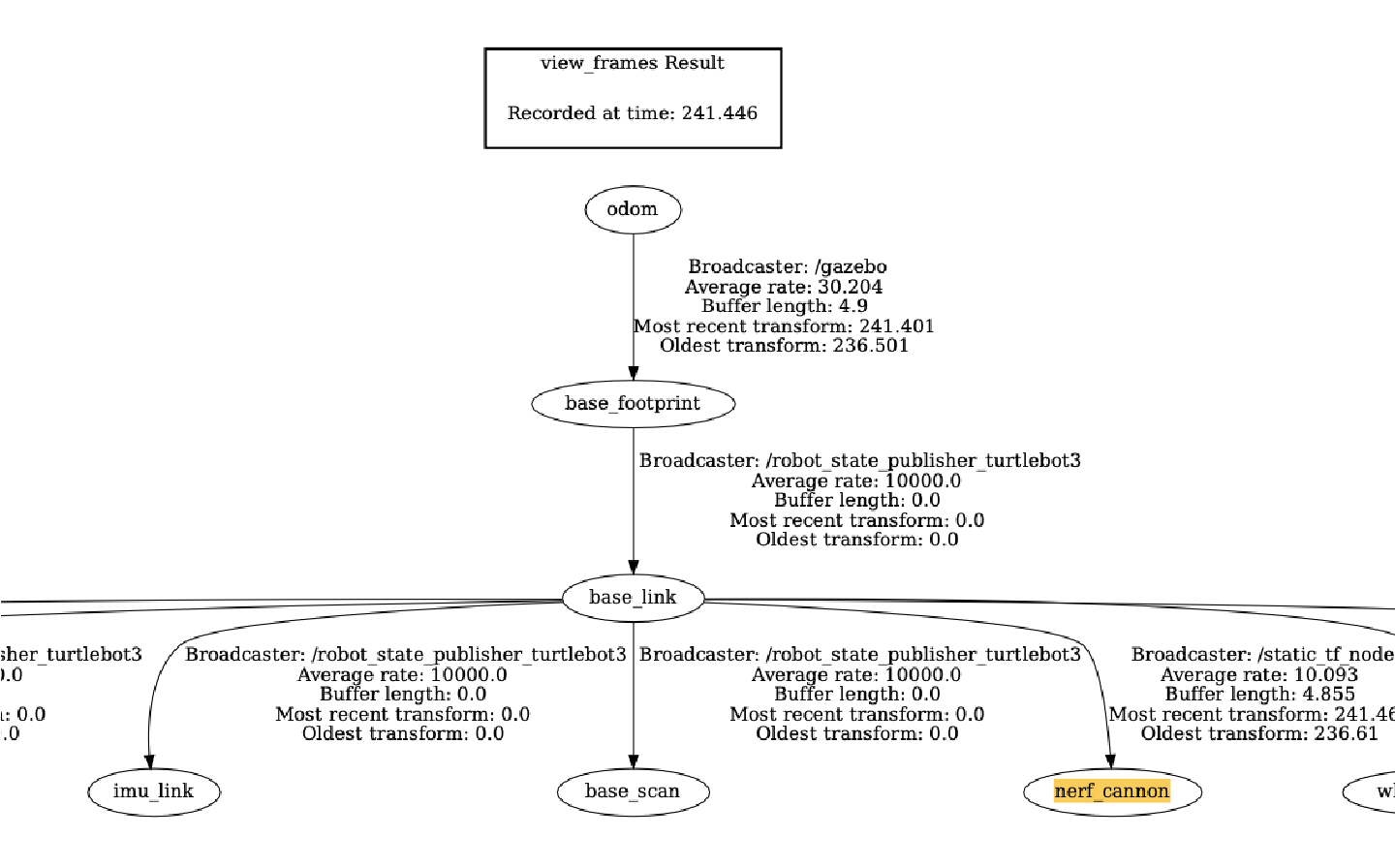
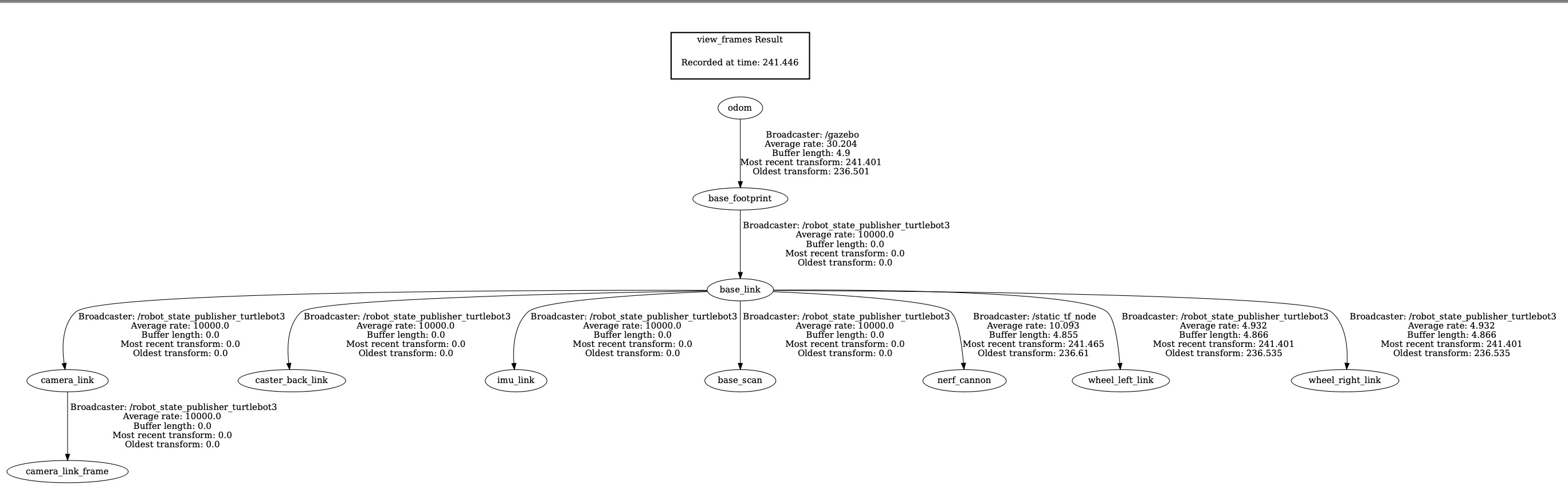
# Documentation & Screen Shots

## 1. project\_mapping(documentation & screenshots)

2.Creating frames.pdf

This is the image of the frame created. It has camera\_link, camera\_link\_frame, imu\_link, caster\_back\_link, base\_scan\_nerf\_cannon, wheel\_left\_link, and

wheel\_right\_link



Command use:

In Shell 1:

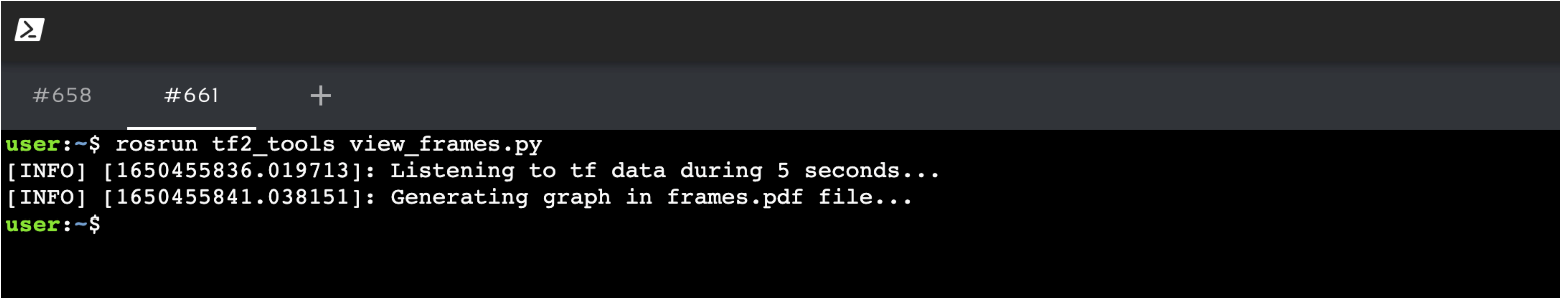
*roslaunch realrobotlab main.launch*

In Shell 2:

*roslaunch project\_mapping project\_mapping\_transforms.launch*

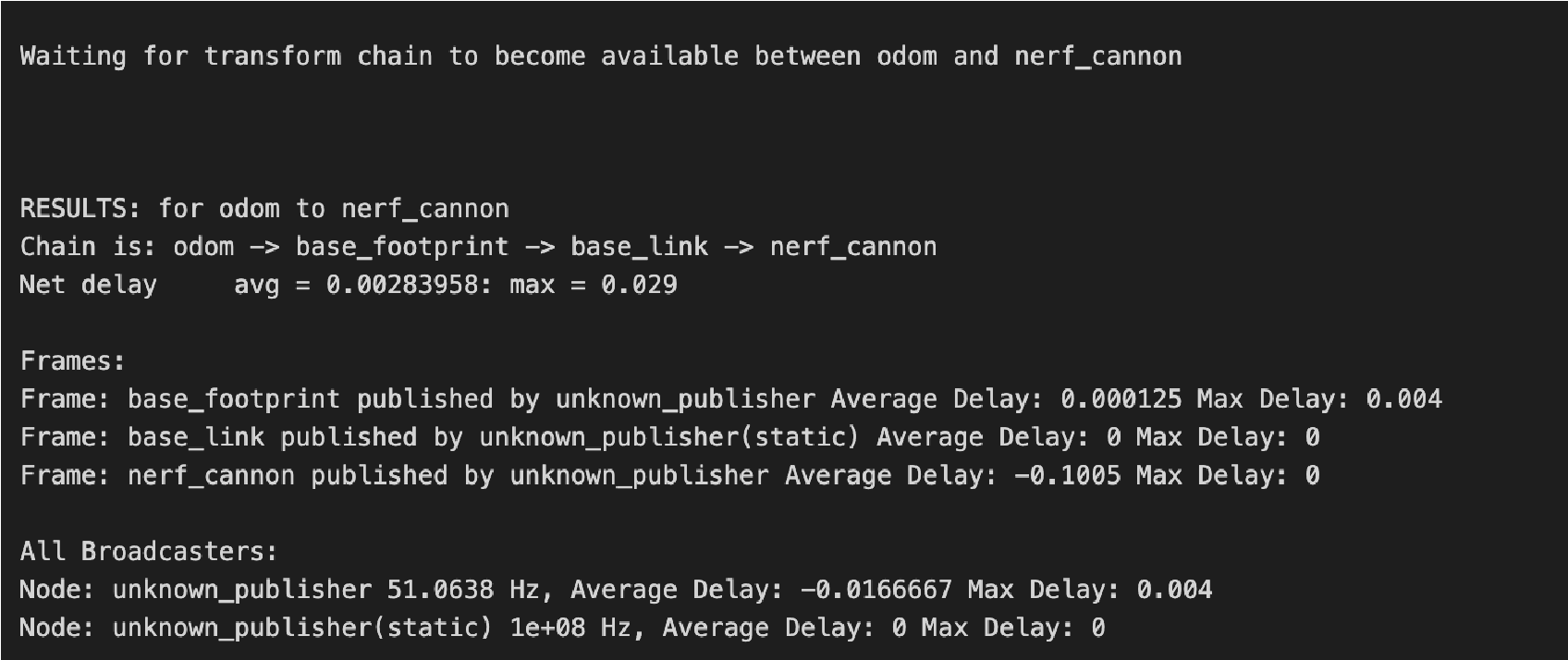
In Shell 3:

*rosrun tf2\_tools view\_frames.py*

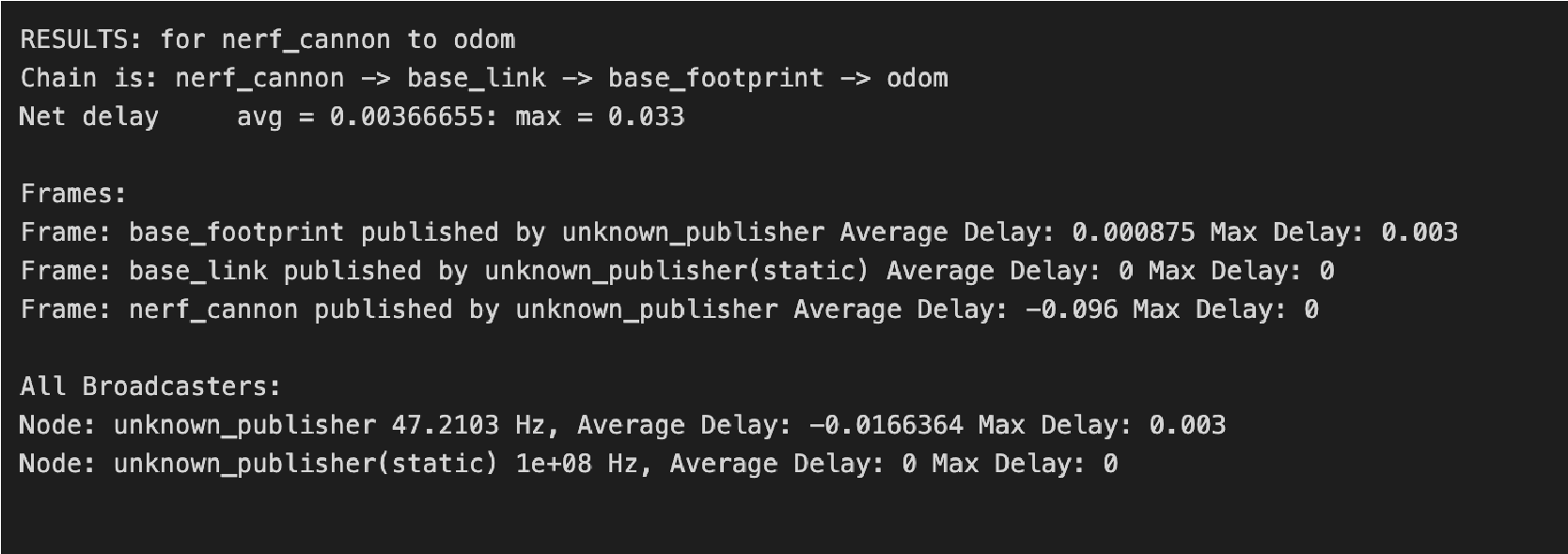


3.tf\_monitor

Transform between Odom and Nerf Cannon

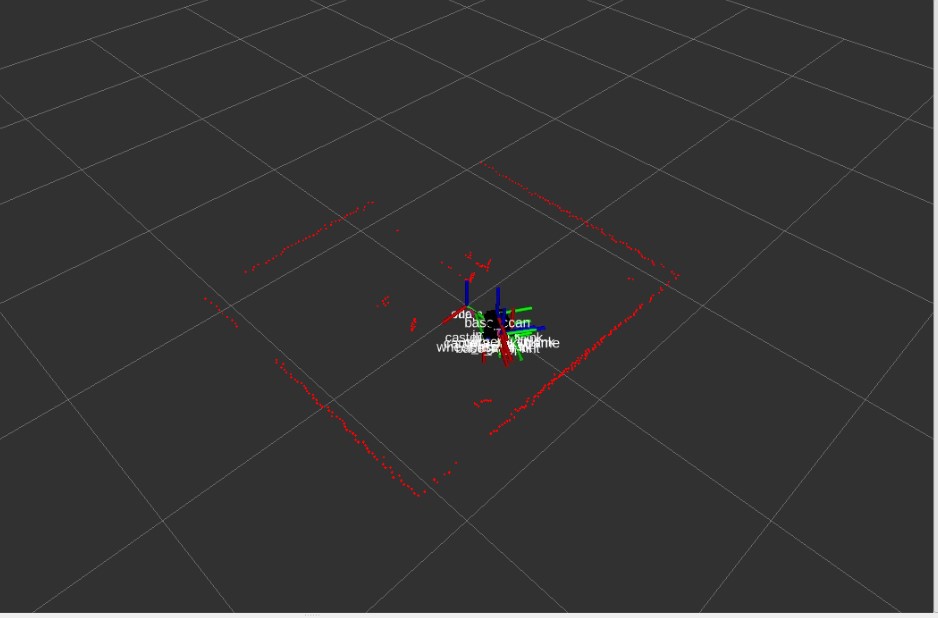


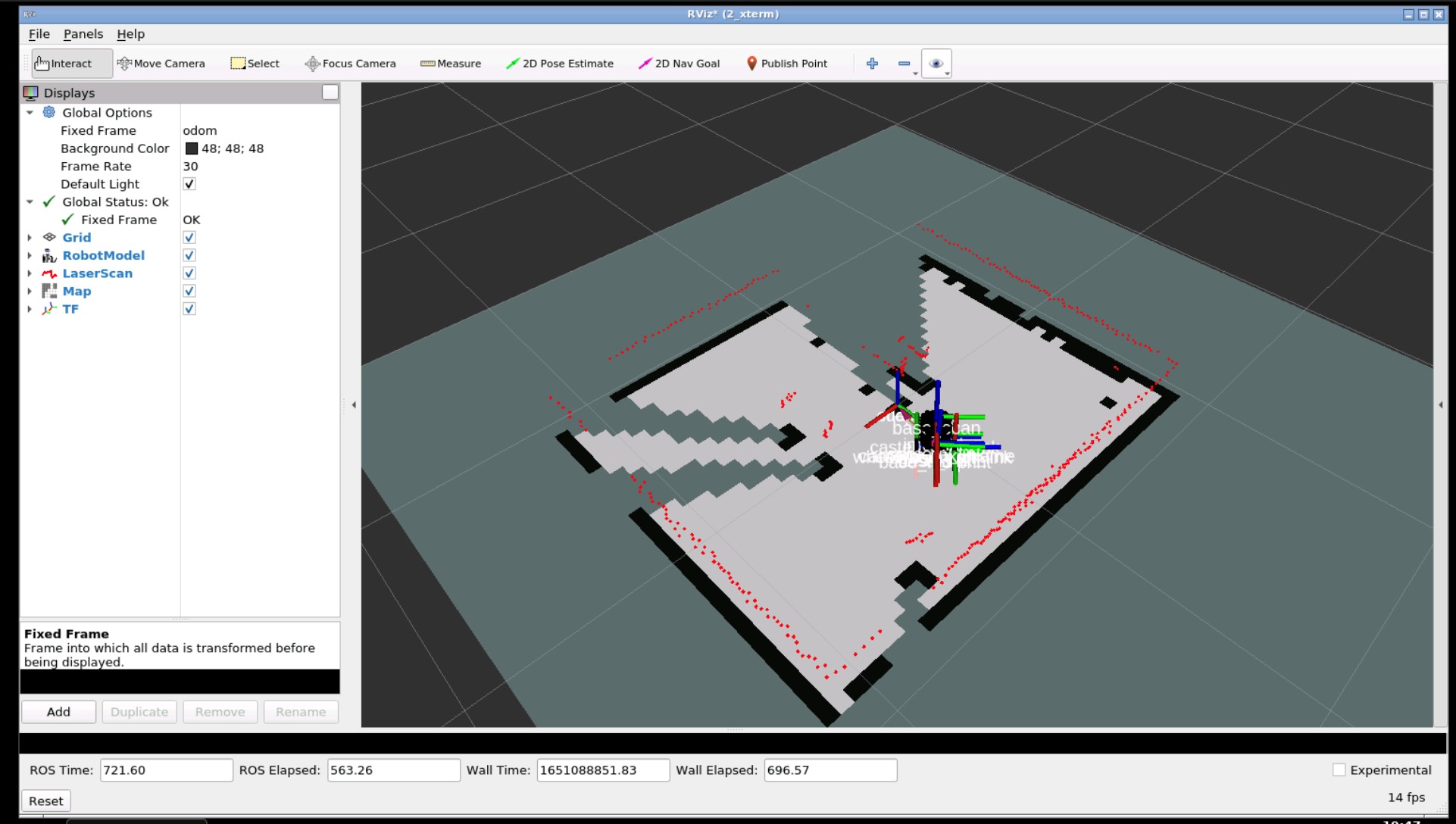
Transform between Nerf Cannon and Odom

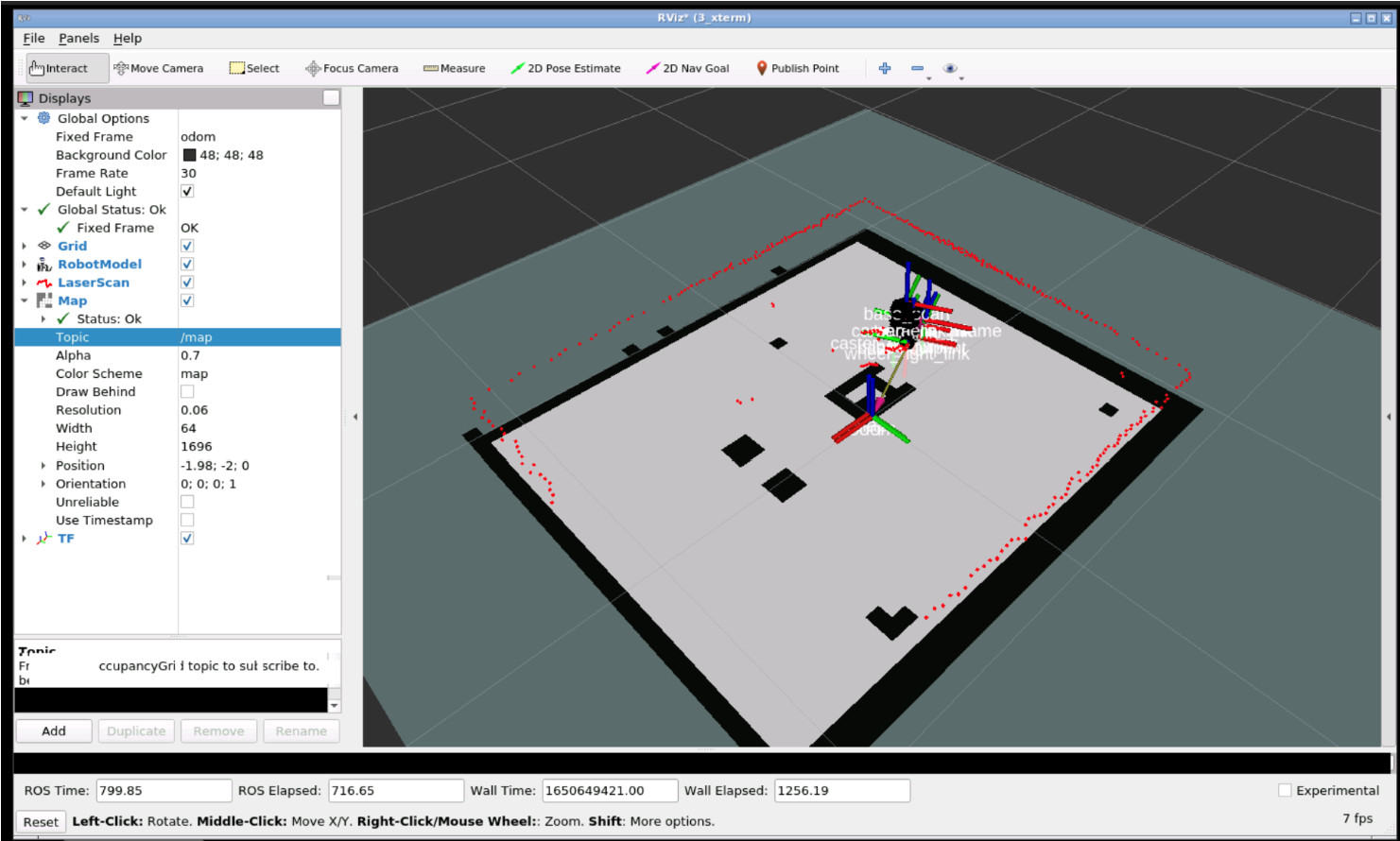


4. Rviz

The image of rviz configuration done while developing project\_mapping package. Laser scan, Robot model, TF and Map are the elements added in the rviz.







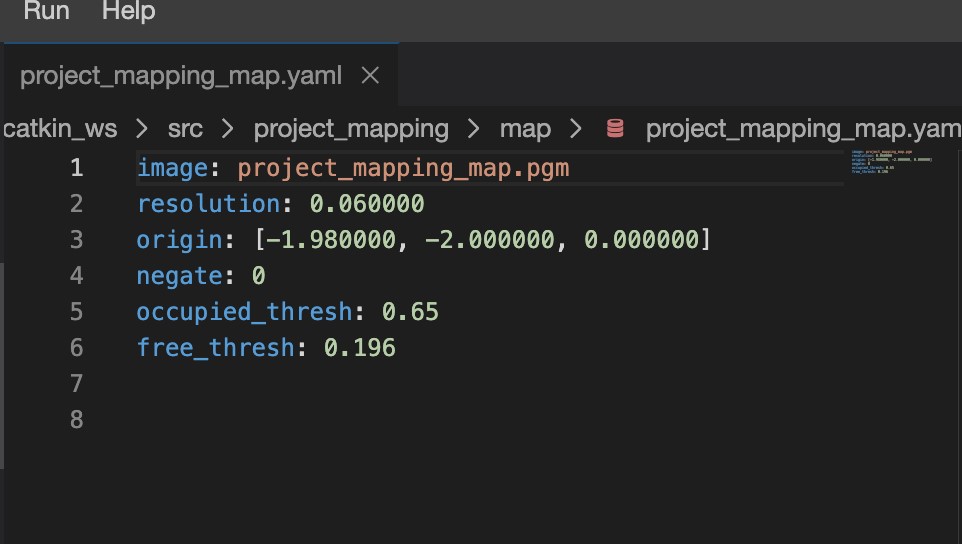
5.Project\_mapping.launch

* 1. maxUrange : sets the maximum usable range of the laser.
  2. maxRange : sets the max range of laser.
  3. map\_update\_interval : sets the time to update after every n seconds.

Where n is value in seconds.

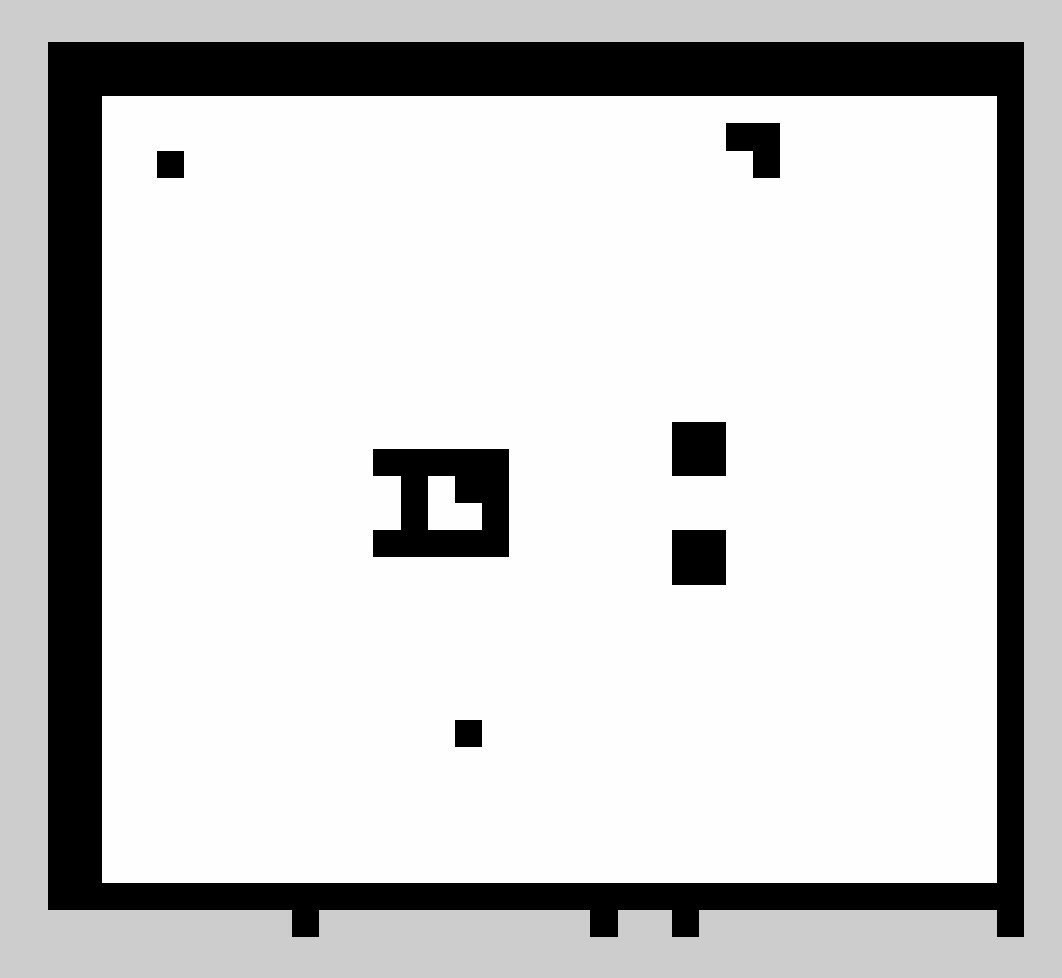
* 1. minimumScore : Sets the minimum score to consider a laser reading good.
  2. Delta : sets the resolution of the map. For ex – default is value is 0.05 which means each square in is 0.05 meters on each side.

7.Project\_mapping\_map.yaml



* **image**: Name of the file containing the image of the generated Map.
* **resolution**: Resolution of the map (in meters/pixel).
* **origin**: Coordinates of the lower-left pixel in the map. This coordinates are given in 2D (x,y). The third value indicates the rotation. If there's no rotation, the value will be 0.
* **negate**: Inverts the colours of the Map. By default, white means completely free and black means completely occupied.
* **occupied\_thresh**: Pixels which have a value greater than this value will be considered as a completely occupied zone.
* **free\_thresh**: Pixels which have a value smaller than this value will be considered as a completely free zone.

Project\_mapping\_map.pgm



8.Map\_server/ map\_provider.launch

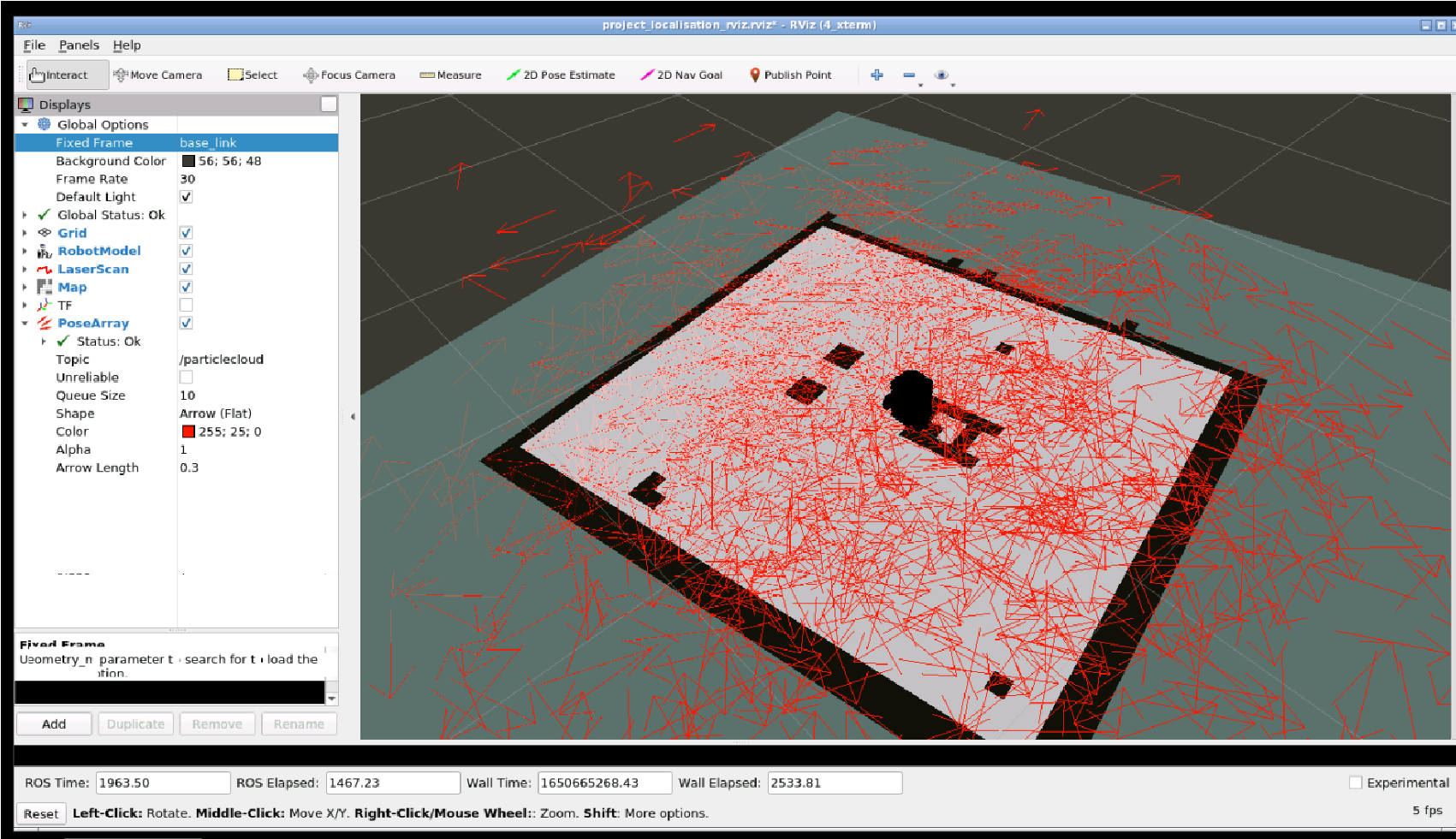
1. **pkg="package\_name"** : Name of the package that contains the code of the ROS program to execute
2. **type="map\_server ":** Name of the program file that we want to execute
3. **name="node\_name"**: Name of the ROS node that will launch our Python file
4. **arg = “local\_variable”** : sometime it is necessary to use local variable in launch files.

## 2. Project\_localization (documentation & screenshots)

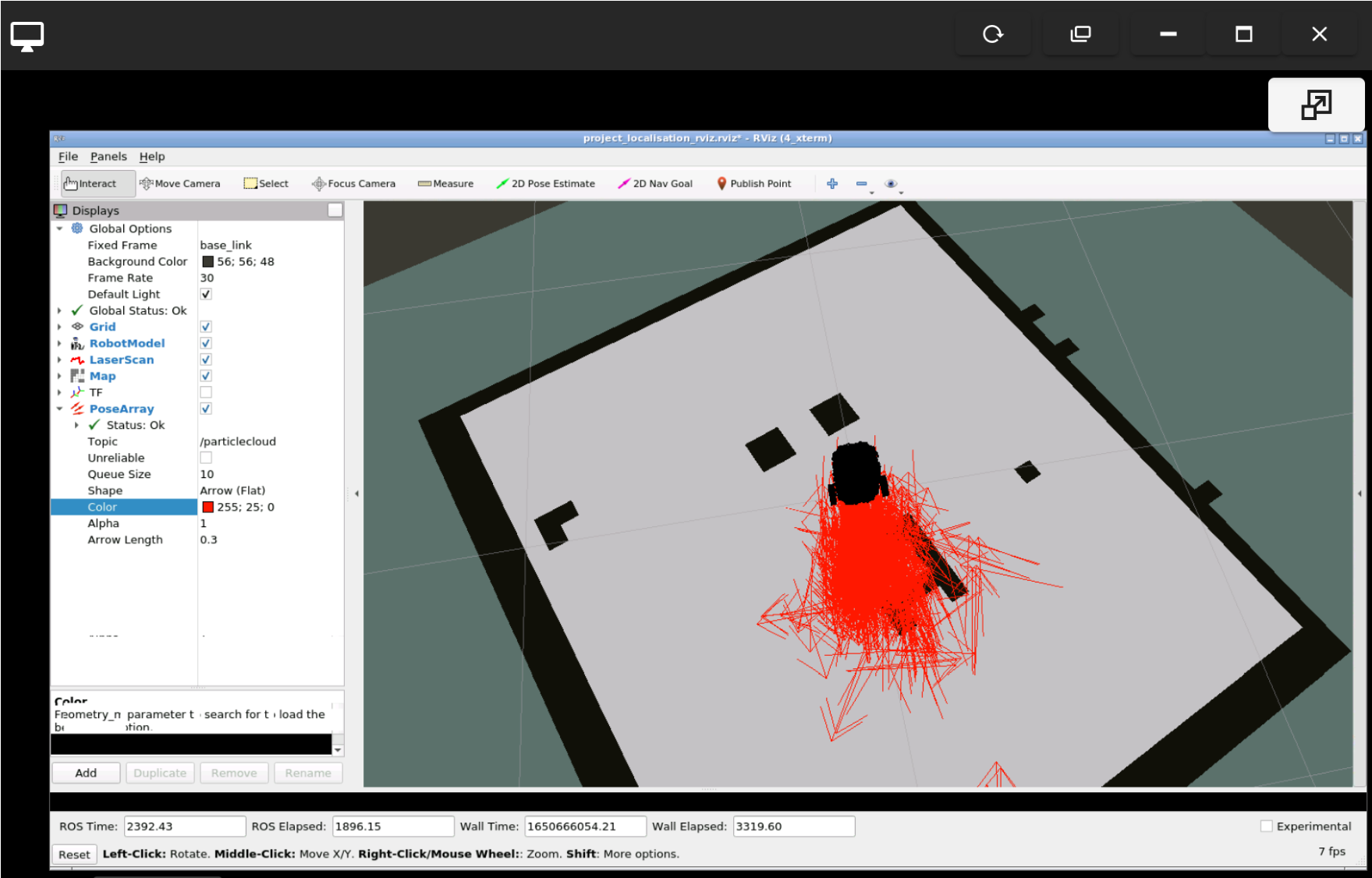
1.Project\_localization\_rviz

The image of rviz configuration while developing project\_localization package. Laser scan, Robot model, PoseArray, as any other appropriate items are the elements added in the rviz.

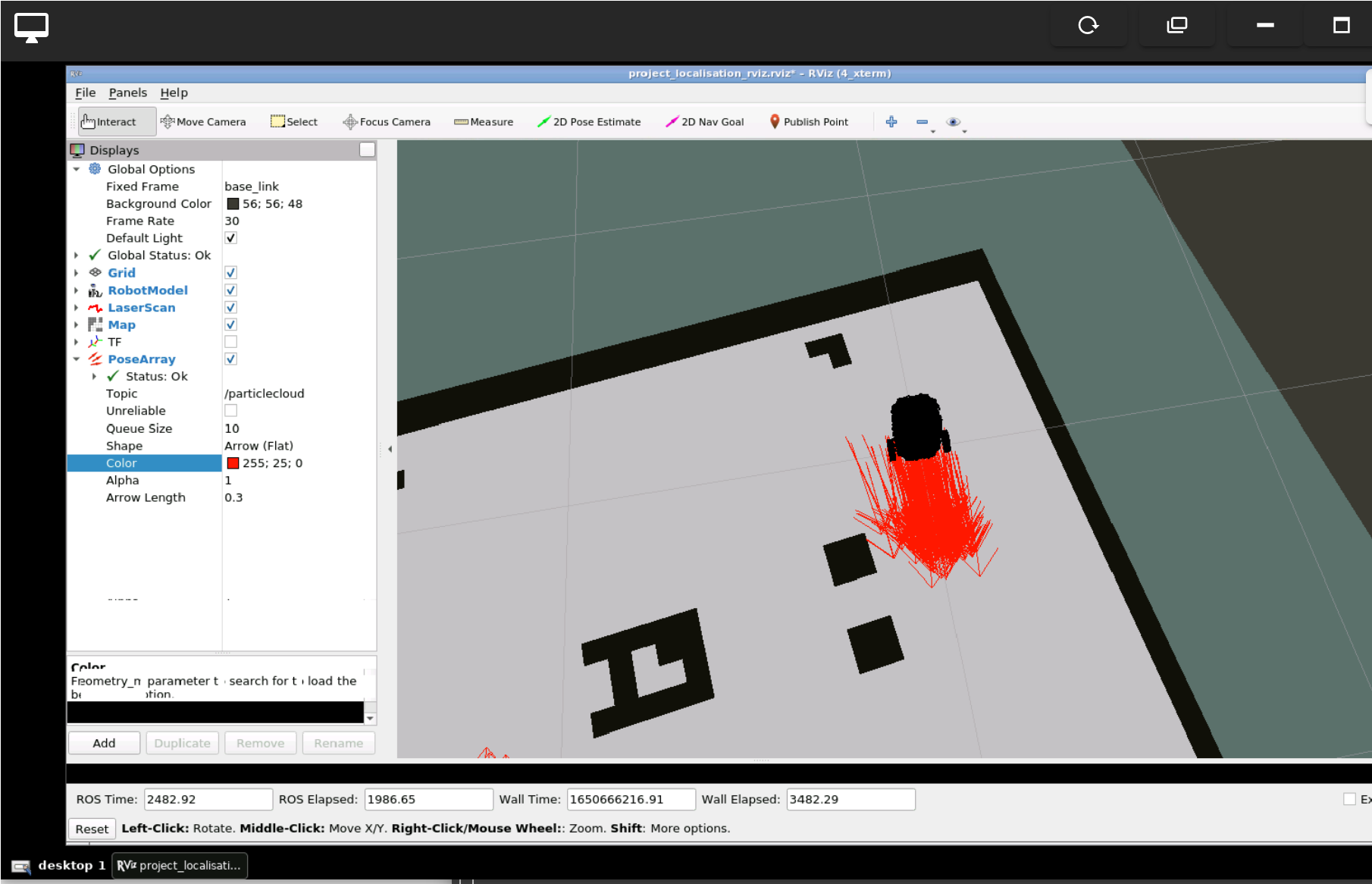
/particleclouds are dispersed at start. The robot does not know where its located.



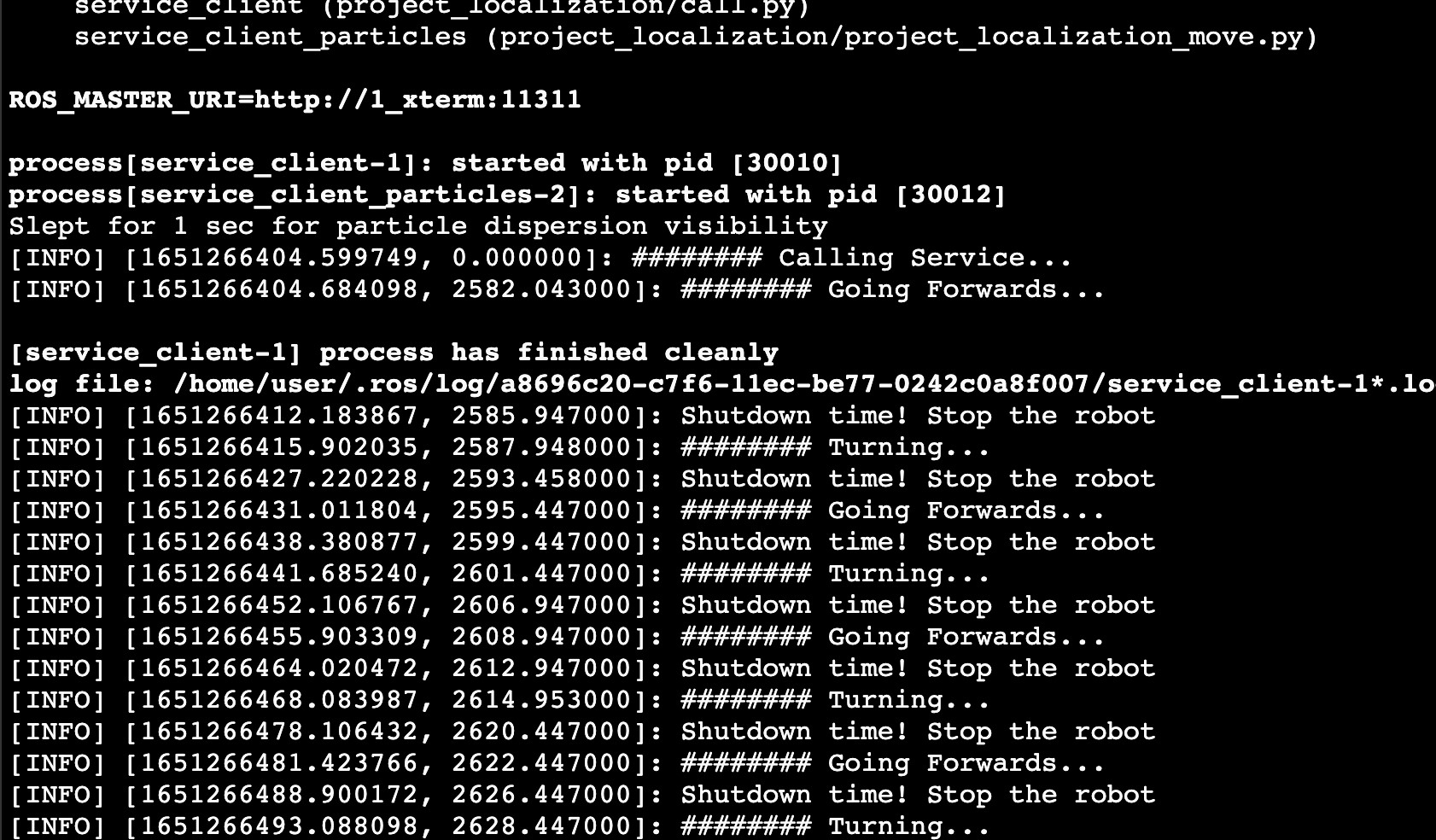
Moving the robot around makes it more localized.



The robot is able to identify where it is.



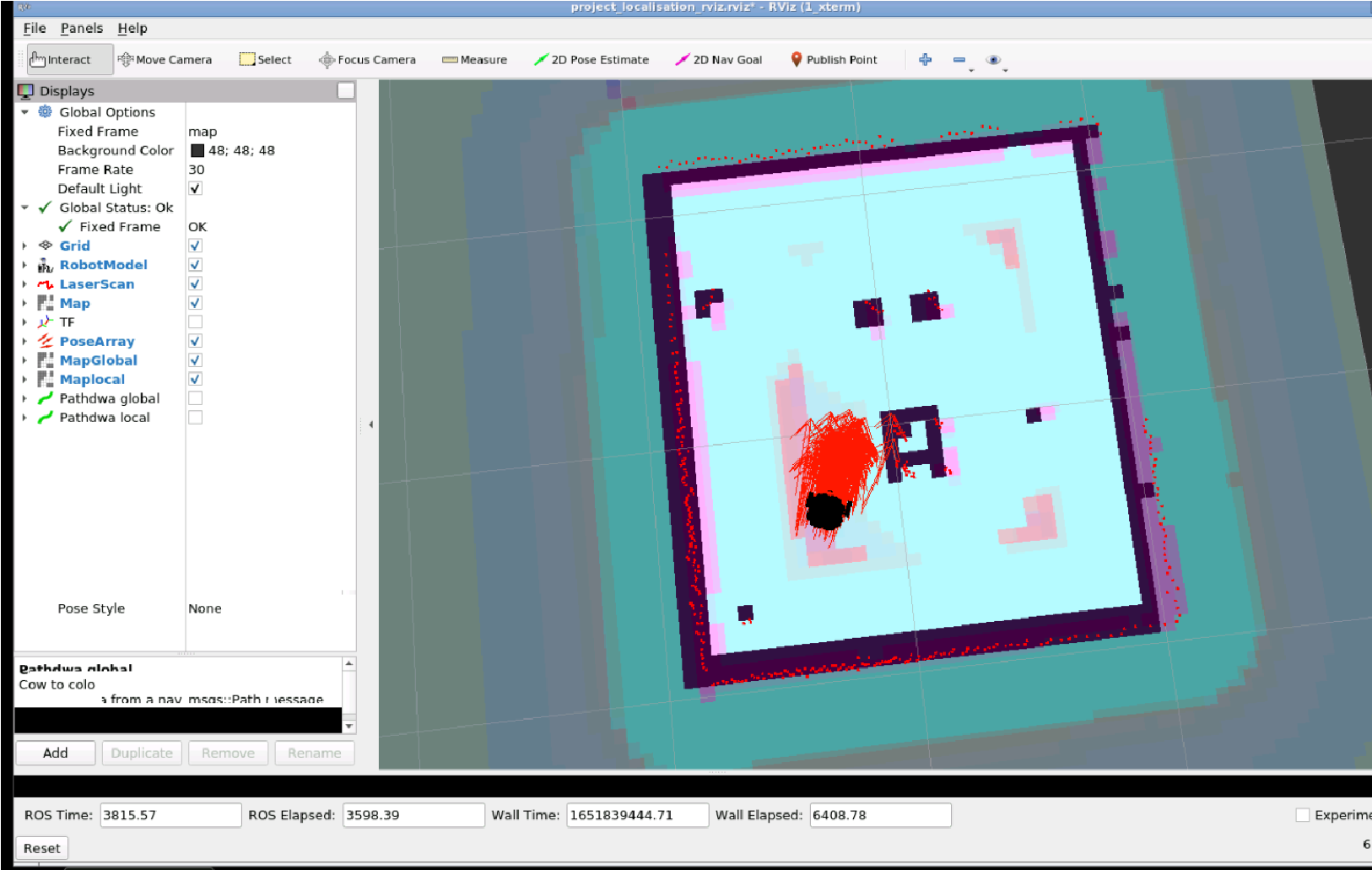
This figure shows the movement of the robot while it is trying to localized itself.



## 3. project\_planning (documentation & screenshots)

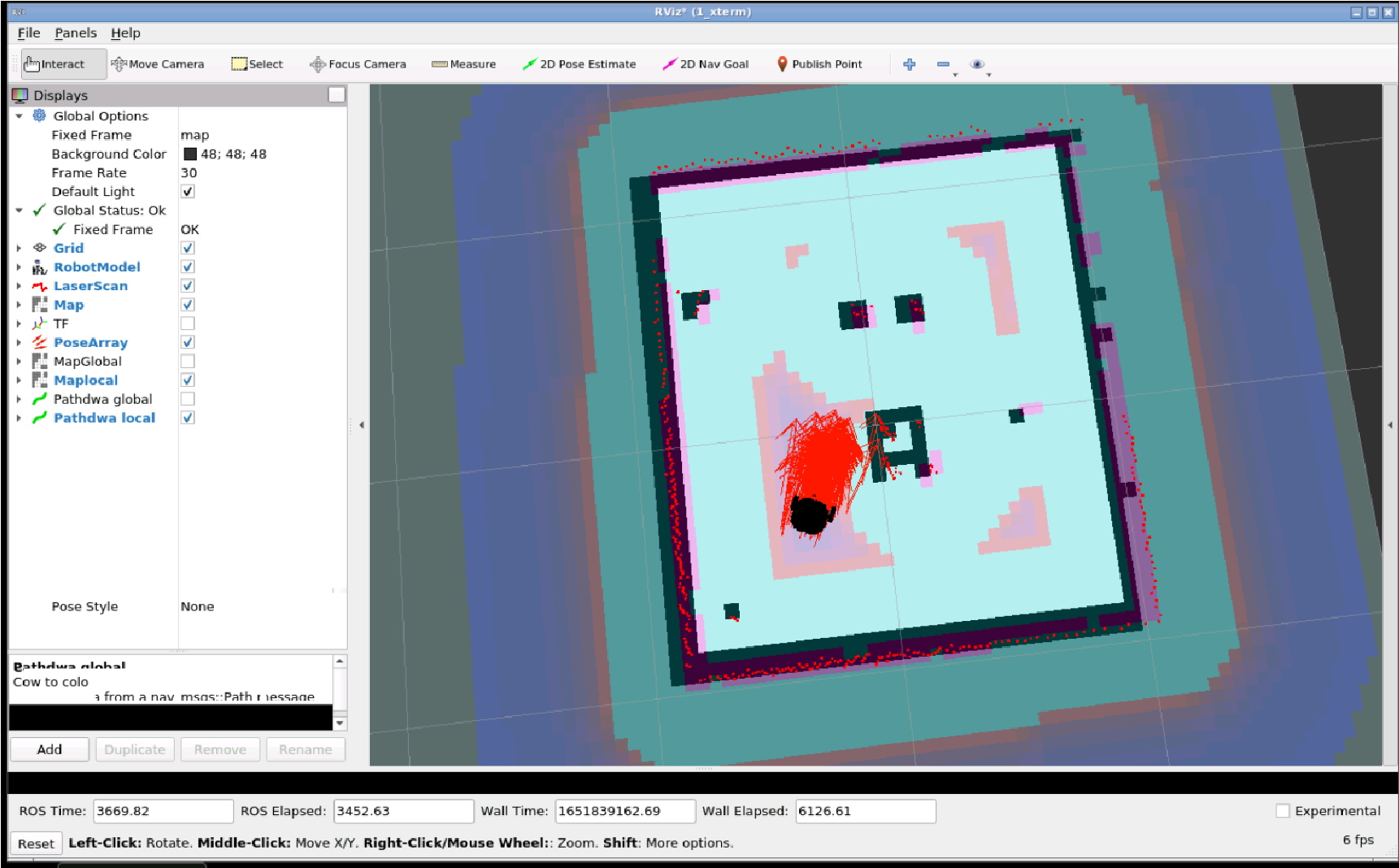
### Rviz Configuration

The figure below show the Rviz configuration using Local and global costmap. The pose array enable the robot to localize itself in the environment.



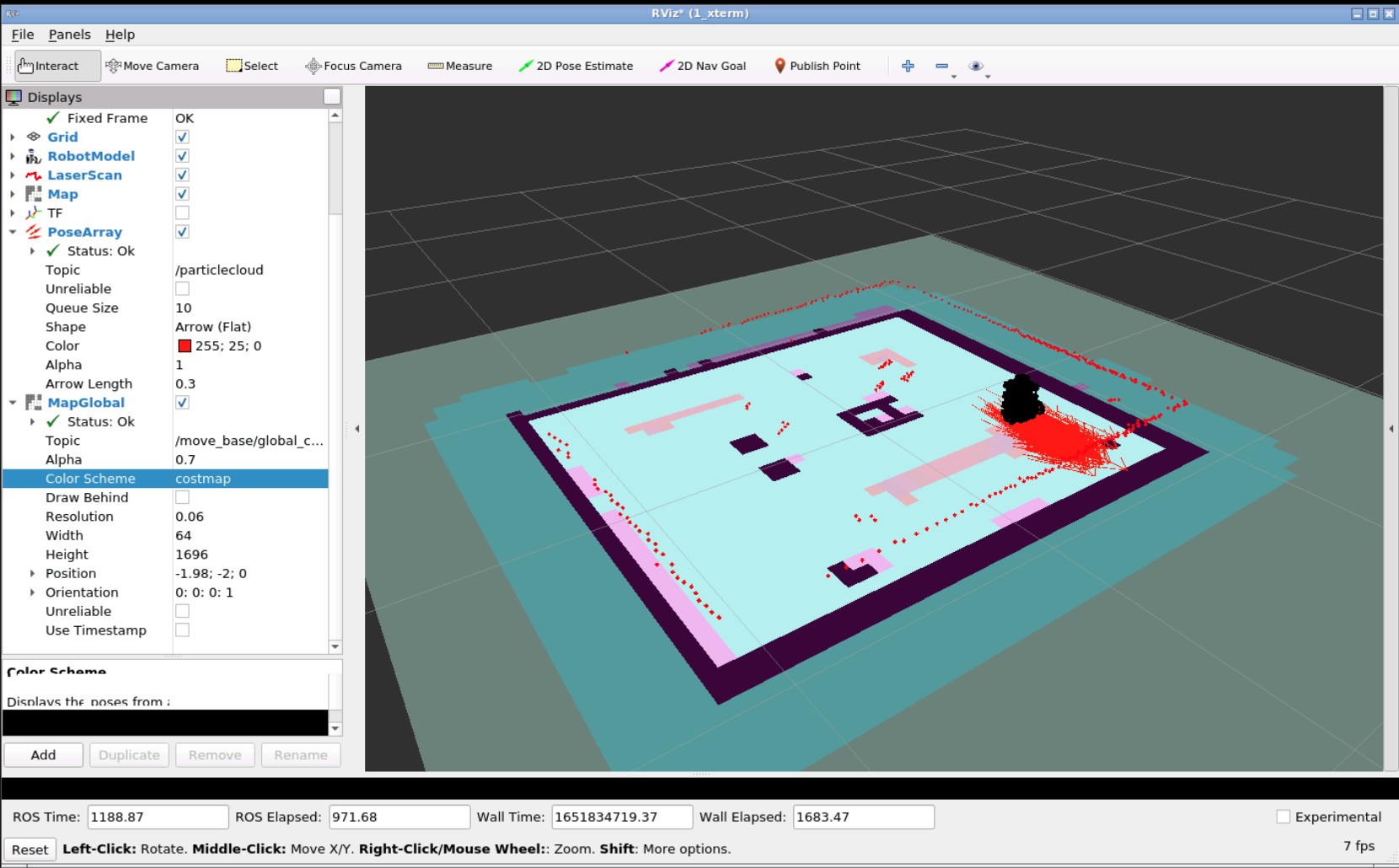
### Local Costmap

In local costmap we can visualize the part the robot can see using its laser. The side of the walls and obstacles are pink indicating it.



### Global costmap

In this figure below the map is set to global costmap to visualize the environment. The border of the map is indicated as pink lines along the border. The laser visualisation the environment including walls and obstacles.



### Launching project\_planning\_send\_goal.launch

Robot searches for the pose and move in that direction. After finding the correct pose a result of state 4 is received.



## 4. Additional Task for Group of 3 - Add additional items to the environment (documentation & screenshots)

**Give permission command** *chmod +x box.urdf chmod +x object.urdf chmod +x cylinder.urdf chmod +x sphere.urdf chmod +x cylinder2.urdf*

**Running command** Red Box: *rosrun gazebo\_ros spawn\_model -file /home/user/catkin\_ws/src/box.urdf -urdf -x 0 -y 0 -z 0.5 -model box*

Blue Box: *rosrun gazebo\_ros spawn\_model -file /home/user/catkin\_ws/src/object.urdf -urdf -x 0 -y 0.5 z 0.15 -model my\_object*

White Sphere:

*rosrun gazebo\_ros spawn\_model -file /home/user/catkin\_ws/src/sphere.urdf -urdf -x -0.5 -y -*

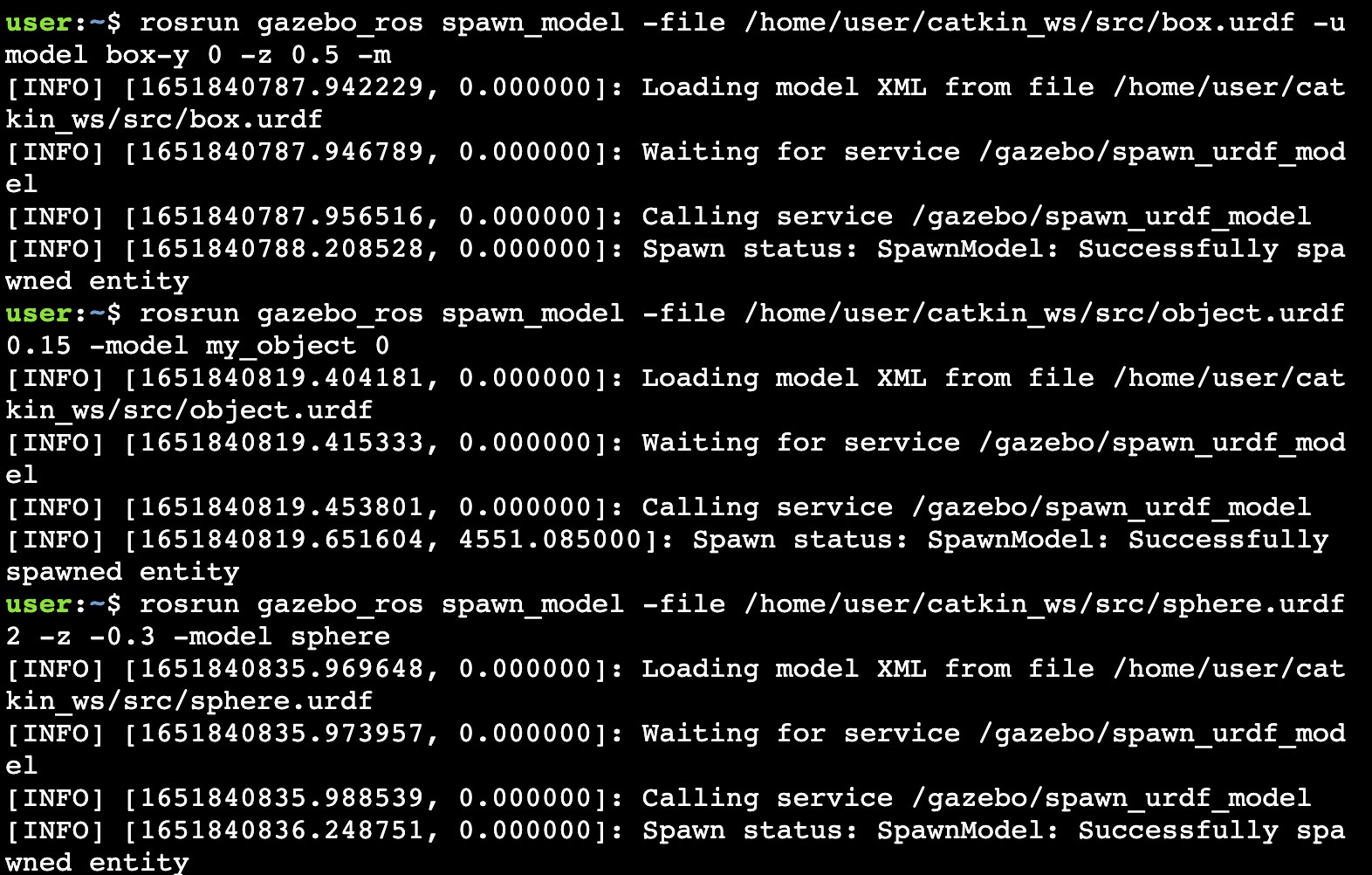
*0.2 -z -0.3 -model sphere*

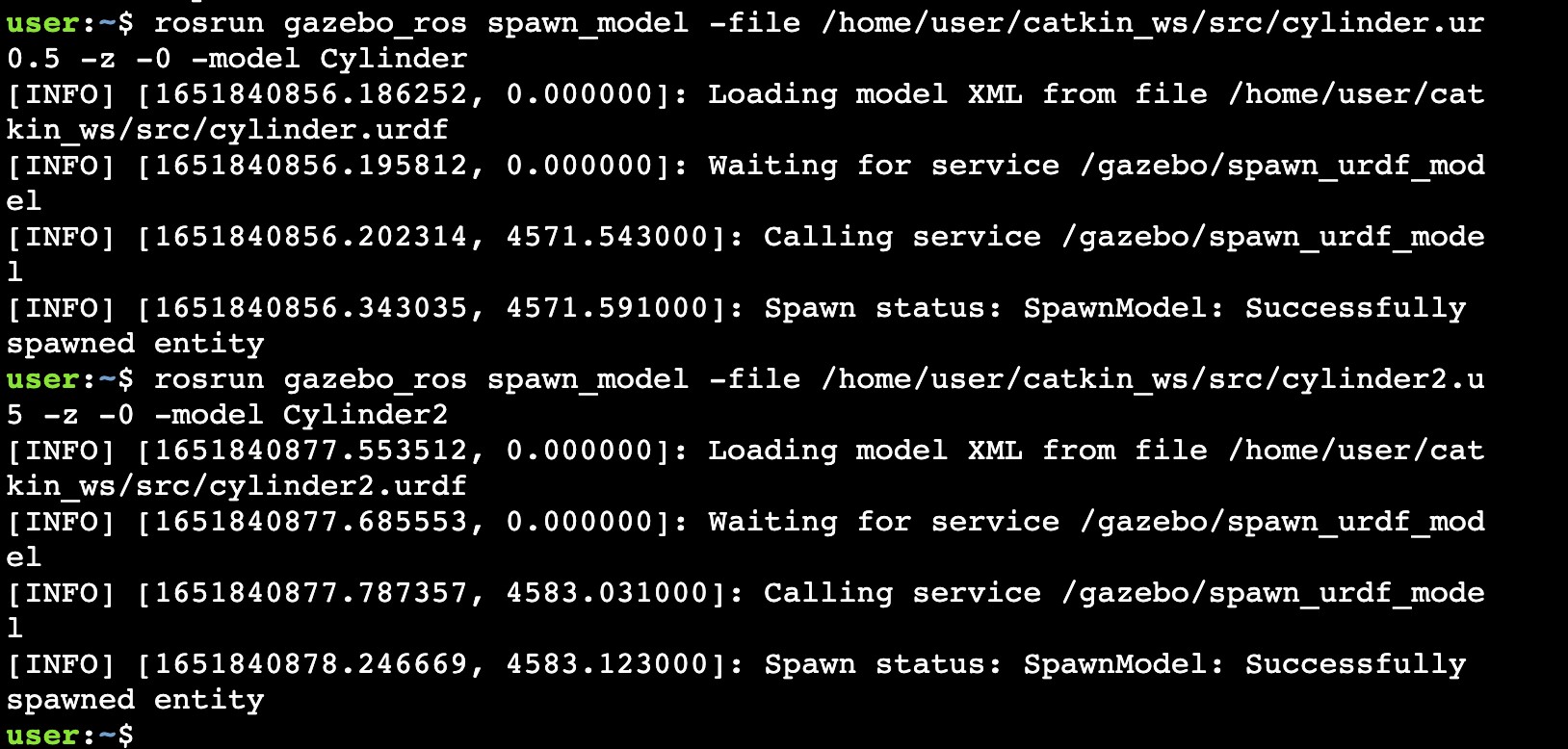
White Cylinder: *rosrun gazebo\_ros spawn\_model -file /home/user/catkin\_ws/src/cylinder.urdf -urdf -x -0.7 -y -0.5 -z -0 -model Cylinder*

Red Cylinder

*rosrun gazebo\_ros spawn\_model -file /home/user/catkin\_ws/src/cylinder2.urdf -urdf -x 0 -y -*

*0.5 -z -0 -model Cylinder2*





The robot moves around the object to move to a particular location that is around the location. Two boxes are on the edge of the course a blue cuboid and a red cube. One white cylinder is in the middle of the course. A white sphere is placed behind the two triangular border. A red cylinder is placed on the track near the red box.



