Programming Assignment 1

Question 1:

Goal: Get familiar with a publisher. Write a small node that drives the robot forward for a specified distance. You should publish to cmd_vel topic (use rostopic info cmd_vel to find the message type) and you can time it so that the robot travels that specified amount of distance.

This Task has been implemented using a node that publishes a velocity (for example 0.2 m/s) to the robot for a specific period of time until the robot travels (1 meter). Then, the node publishes a velocity (0 m/s) to stop the robot.

The steps of the implementation are as follows:
1- Create a node.
2- Initialize a publisher to publish Twist message to /cmd_vel topic.
3- Set the linear velocity of the robot to 0.2m/sec.
4- Publish this Twist message.
5- Keep calculating the distance by multiplying the velocity by the time the robot spent moving until the distance is equal to the required distance (1m/sec).
6- Then, set the linear velocity of the robot to (0 m/sec) and publish the Twist message again to stop the robot.

Evaluation: The robot did exactly what it should do as it moved the specified distance (1 meter). However, because of the frictions and noise factors, the robot moved a little bit less that 1 meter. It moved (0.96 meters). To enhance the performance of the robot, a controller such as (PID controller) could be used.

P.S. Alberto helped me with specifying the location of the rospy.sleep(1) function in the code.

Question2:

Goal: Data collection. Turn on the robot and run the sensors using roslaunch, and record a bag file with all topics (images compressed), as we saw in class. Place the robot facing a wall at 2 m distance, use the ROS node written in point 1. to drive the robot 1 m towards the wall with a velocity of 0.2 m/s. Please when running this data collection experiment be careful for the robot to stop (or be ready to catch the robot). Measure also the distances with a measuring tape and record it on a CSV file.

This Task has been done using the node that has been implemented in the previous task. Using the previously implemented node, the robot has been driven 1 meter towards the wall with 0.2 m/s velocity. During the robot movement a rosbag command has been run to record some sensing information from the environment around the robot.

The steps of the implementation are as follows:
1- Place the robot 2 meters away from the wall.
2- Run: roslaunch rosbot-description rosbot-hardware.launch to run all the robot's sensors.
3- Start recording all the robot's topics except the uncompressed images using (rosbag record topics_names).

4- Then, run the previous node using rosrun first-programming-assignment PA1-a.py to drive the robot 1 meter towards the wall.
5- When the robot stops, stop the recording using ctl+c.
6- The bag file can be found in the current directory with all the specified topics recorded.

Evaluation: A bag file has been created successfully that includes all the sensing information about the robot. The traveled distance has been measure to (0.96 m) instead of (1.0 m) because this is a real experiment and there are many factors and noises that can affect the robot's performance.

P.S. The bag file and the CSV files are attached with the report.

Question 3:

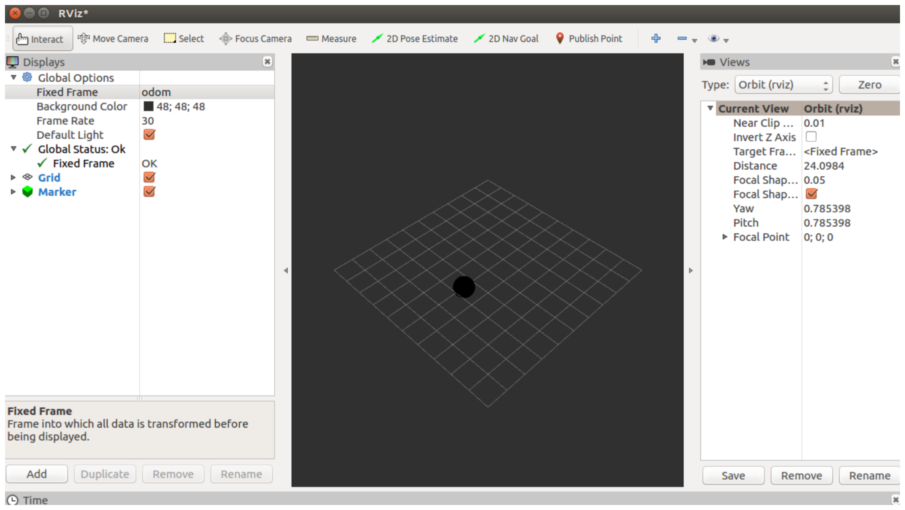
Goal: Get familiar with a subscriber. Write another small node that subscribes to the odom topic and plots the x,y position of the robot on a graph with axes x and y.

This Task has been implemented using a node that subscribes to the /pose topic in rosbot to read the x and y values that the robot traveled and publishes them to the Marker topic to visualize them in rviz.

The steps of the implementation are as follows:
1- Create a subscriber that reads the PoseStamped messages from the /pose topic.
2- Create a publisher that publishes Marker messages to the /visualization_marker topic.
3- Set the marker id to be "odom".
4- Read the x and y values from odom and write them to the x and y values of the /Marker.
5- Finally, publish the marker message and add the Marker topic to rvize to visualize the changes of the robot location.

Evaluation: I manage to visualize the location of the robot while it moves in rviz from the position (0,0) to (13,0). See below some screen-shots of the robot locations on rviz.

P.S. Alberto helped me with adjusting the timing of the robot visualization on rviz.

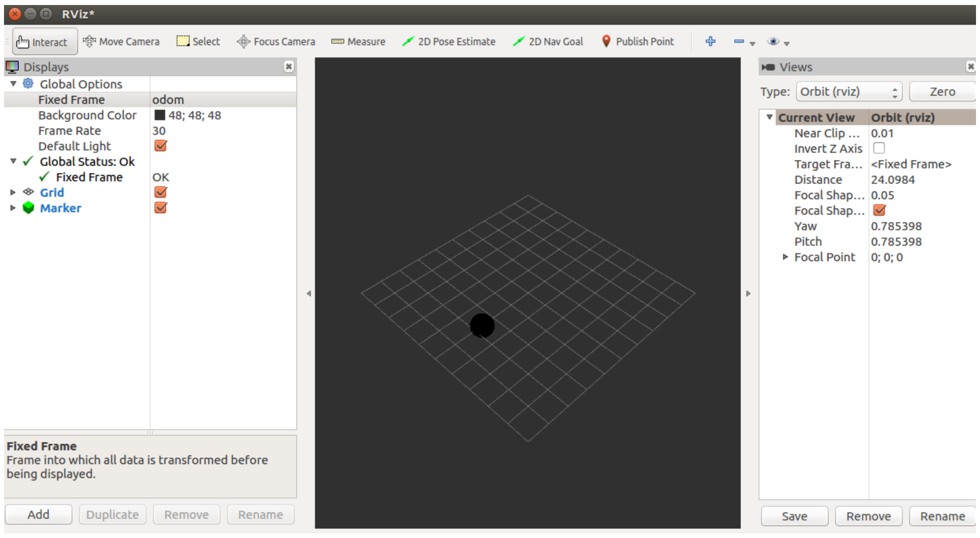


Global Options	
Fixed Frame	odom
Background Color	■ 48; 48; 48
Frame Rate	30
Default Light	✓
✓ Global Status: Ok	
✓ Fixed Frame	OK
Grid	✓
Marker	✓

Fixed Frame
Frame into which all data is transformed before being displayed.

Type: Orbit (rviz)

Current View	Orbit (rviz)
Near Clip ...	0.01
Invert Z Axis	☐
Target Fra...	<Fixed Frame>
Distance	24.0984
Focal Shap...	0.05
Focal Shap...	✓
Yaw	0.785398
Pitch	0.785398
Focal Point	0; 0; 0



Displays

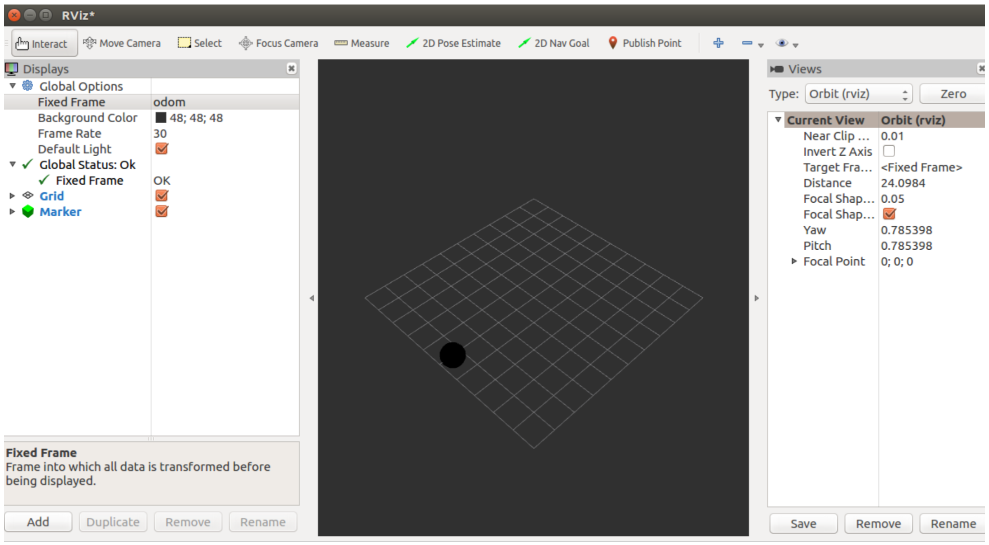
Global Options	
Fixed Frame	odom
Background Color	■ 48; 48; 48
Frame Rate	30
Default Light	✓
Global Status: Ok	
Fixed Frame	OK
Grid	✓
Marker	✓

Fixed Frame
Frame into which all data is transformed before being displayed.

Views

Type: Orbit (rviz)

Current View	Orbit (rviz)
Near Clip …	0.01
Invert Z Axis	☐
Target Fra…	<Fixed Frame>
Distance	24.0984
Focal Shap…	0.05
Focal Shap…	✓
Yaw	0.785398
Pitch	0.785398
Focal Point	0; 0; 0



Displays

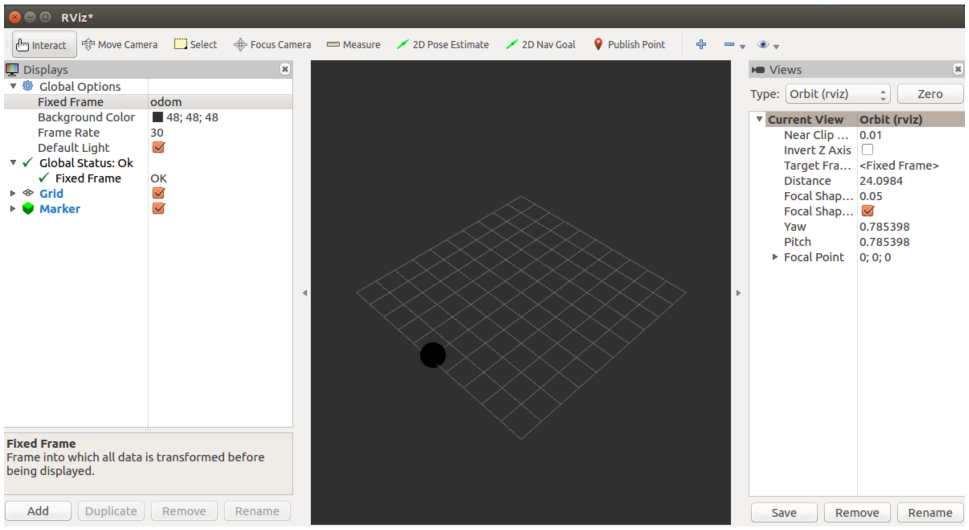
Global Options	
Fixed Frame	odom
Background Color	■ 48; 48; 48
Frame Rate	30
Default Light	✓
Global Status: Ok	
Fixed Frame	OK
Grid	✓
Marker	✓

Fixed Frame
Frame into which all data is transformed before being displayed.

Views

Type: Orbit (rviz)

Current View	Orbit (rviz)
Near Clip …	0.01
Invert Z Axis	☐
Target Fra…	<Fixed Frame>
Distance	24.0984
Focal Shap…	0.05
Focal Shap…	✓
Yaw	0.785398
Pitch	0.785398
Focal Point	0; 0; 0



Displays

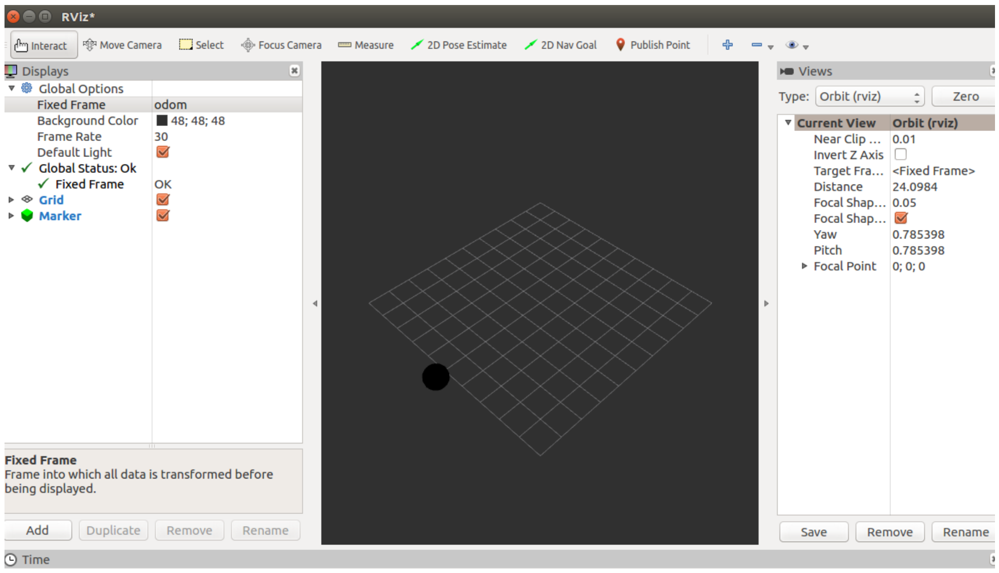
Global Options	
Fixed Frame	odom
Background Color	■ 48; 48; 48
Frame Rate	30
Default Light	✓
Global Status: Ok	
Fixed Frame	OK
Grid	✓
Marker	✓

Fixed Frame
Frame into which all data is transformed before being displayed.

Views

Type: Orbit (rviz)

Current View	Orbit (rviz)
Near Clip …	0.01
Invert Z Axis	☐
Target Fra…	<Fixed Frame>
Distance	24.0984
Focal Shap…	0.05
Focal Shap…	✓
Yaw	0.785398
Pitch	0.785398
Focal Point	0; 0; 0



Global Options
Fixed Frame odom
Background Color ■ 48; 48; 48
Frame Rate 30
Default Light ✓
✓ Global Status: Ok
✓ Fixed Frame OK
Grid ✓
Marker ✓

Fixed Frame
Frame into which all data is transformed before being displayed.

Type: Orbit (rviz)

Current View	Orbit (rviz)
Near Clip …	0.01
Invert Z Axis	☐
Target Fra…	<Fixed Frame>
Distance	24.0984
Focal Shap…	0.05
Focal Shap…	✓
Yaw	0.785398
Pitch	0.785398
Focal Point	0; 0; 0