Robotics Project : Where am I?

Shotaro Oyama

Abstract—The robot localization in simulation using ROS, Gazebo, and RViz. The robot finds and reaches the desired goal pose and orientation in the given situation. The Adaptive Monte Carlo Localization, AMCL in short, a Particle filter-based algorithm is used for the robot localization. This project also configures the specifiation of Robot by URDF file format and tweak the parameters in order for the robot to work well. The robot is implemented camera and hokuyo sensors node and the data from them is used for AMCL node and move_base node which is used for navigation of the robot.

Index Terms—Where am I project, Monte Carlo Localization,Particle Filter, Robotics Nanodegree, Udacity

1 Introduction

Localization is the challenge of determining the pose of robot in a mapped environment. There are three challenges, local localization, global localization, and the kidnapped robot problems. In local localization, or position tracking, robot knows its initial pose and the problem is to keep track of the robots pose as it moves. In global localization, the robots initial pose is unknown, and the robot tries to determine its pose relative to the ground truth map. In the kidnapped robot problem, robots initial pose is unknown, and to be kidnapped at any time and moved to another location of the map. These challenges assume the situations in which robot gets involved.

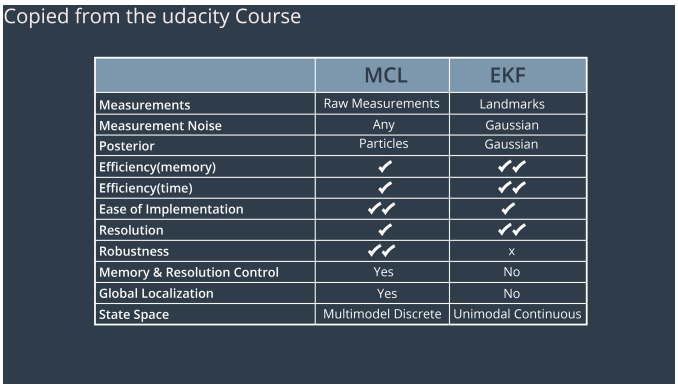
In this project, robot tries to solove global localization problem. The two robots traverse a known mapped environment then navigate to the desired goal, though robot does not know its initial pose. Using the sensor data, and mitigating the noise by the Adaptive Monte Carlo Localization algorithm, robot try to solve the problem.

2 Background / Formulation

Localization problems are mainly caused by noisy sensors and actuators. Instead of totally relying o the data from them, there are several approaches. In this project, Kalman Filter and Monte Carlo Localization (Particle filter) are focused.

2.1 Kalman Filter

Kalman Filter is an estimation filter technique used to estimate a value of a variable in real time as the data is being collected such as a robots pose. This filter is very prominent in control systems due to its accuracy and computational efficiency. However, Kalman Filter requires the assumptions that (1)Motion and measurement models are linear, State space can be represented by a unimodal Gaussian distribution. In the nonlinear problem, Extended Kalman Filter, which linearize a nonlinear motion or measurement function by Taylor Expansion.



	MCL	EKF
Measurements	Raw Measurements	Landmarks
Measurement Noise	Any	Gaussian
Posterior	Particles	Gaussian
Efficiency(memory)	✓	✓✓
Efficiency(time)	✓	✓✓
Ease of Implementation	✓✓	✓
Resolution	✓	✓✓
Robustness	✓✓	x
Memory & Resolution Control	Yes	No
Global Localization	Yes	No
State Space	Multimodel Discrete	Unimodal Continuous

Fig. 1. Kalman Filter and Particle Filter Comparison

2.2 Monte Carlo FIter(Particle Filter)

Particle Filter is the algorithm that randomly and uniformly spreads particle within the entire state space. Each particle represents a guess of the robots pose and orientation, and also contains a weight which is the difference between the actual and estimated. The value of the weight for each particle corresponds to the accuracy of each guess.These particles are re-sampled every time the robot movesby sensing the environment through range-finder sensors. After iterations, these re-sampled particles eventually converge with the robots pose, allowing the robot to know its location and orientation. Particle Filter is not limited to linear models.

2.3 Comparison

Extended Kalman Filter has advantage for time, memory efficiency and accuracy. However, it requires Linear Gaussian state space assumption. Monte Carlo Localization algorithm is easier to program and setup than Extended Kalman Filter, and it can be used to represent any model. This advantage makes it since the world cannot always be modeled by Gaussian distributions. (See Fig. 1).

3 Results

When My_robot is made initially, it had only 2 wheels, and it was unstable. Thus 2 casters were added to keep its

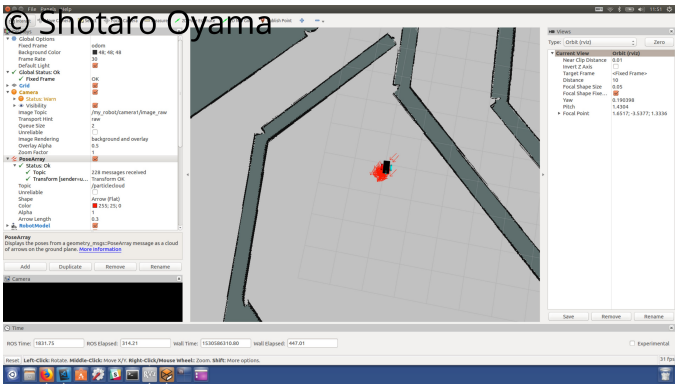


Fig. 2. Udacity_bot goal image

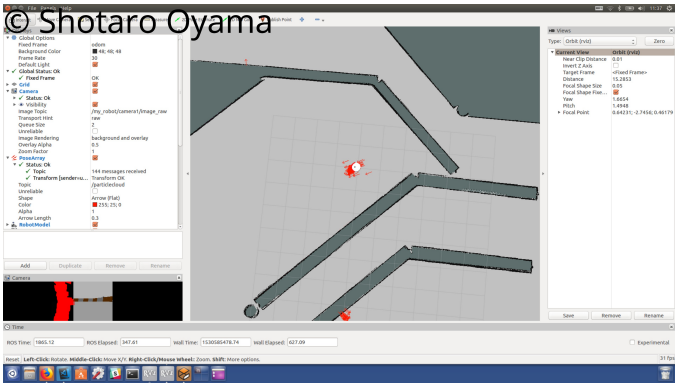


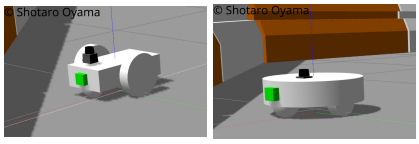
Fig. 3. My_robot goal image

posture. The initial parameters of My_robot were the same as Udacity_bot then it failed to reach goal. After tweaking parameters, finally My_robot also succeeded. (See Fig. 2 & 3)

4 Model Configuration

4.1 Design

Udacity_bot is a robot provided by Udacity course. My_robot, inspired by the home cleaning robot, is designed and based on the Udacity_bot. The design of My_robot is developed by creating a Unified Robot Description Format file (URDF). (See Fig. 4)



[]Udacity_bot [2]My_robot

Fig. 4. Design of Robots

4.2 Parameters

The parameter tweaked from several iterations. Especially, Costmap parameters are changed larger and others are kept as almost the same as Udacity_bot parameters. The final versions of parameters are as follows. (See TABLE. 1)

TABLE 1
Parameters for Robots

	Udacity_bot	My_robot
AMCL		
min particles	10	10
max particles	200	200
initial pose x, y, z	0	0
odom model type	diff-corrected	diff-corrected
odom alpha 1,2,3,4	0.010	0.010
Costmap Common		
obstacle range	2.5	3.5
raytrace range	3.0	5.0
transform tolerance	0.3	2.0
robot radius	0.25	0.4
inflation radius	0.5	0.6
Base Local Planner		
holonomic robot	false	false
yaw goal tolerance	0.05	0.05
xy goal tolerance	0.1	0.1
sim time	1.0	1.0
meter scoring	true	true
pdist scale	0.5	0.5
gdist scale	1.0	1.0
max vel x	0.5	0.5
max vel y	0.1	0.1
max vel theta	2.0	2.0
acc lim theta	5.0	5.0
acc lim x	2.0	2.0
acc lim y	5.0	5.0
controller frequency	15.0	20.0

5 Discussion

As a result, Udacity_bot and My_robot were able to reach the goal. Their performance were almost tha same in terms of the time spent to arrive the goal position. From the RViz window it looks that there were much noise or difficulty of pose assumption in My_robot because the robot kidnapped to far location more frequently than Udacity_bot case. However, spread particles were gathered to correct direction gradually in both cases, and resulted to reach the goal. It meant that the robot was able to understand and update the surrounding environment in real time, it would be helpful in warehouse automation / logistics industry like the robot of Amazon which is reported frequently in news article.

6 Future Work

Using the AMCL algorithm, both robots were able to reach the goal successfully. Through the project, it was observed that the warning message was alerted repeatedly for the controller cannot meet the required frequency. AMCL can implement easily and get accurate result but the next steps should be to find more correct parameters to mitigate the calculation amount or implement the algorithm to more powerful Hardware such as GPU / Jetson TX2.

Also, the robot may be able to implement additional sensors to find its pose more precisely. It will affect the computation amount and may take time to calculate. Thus, finding appropriate combination of sensors and tweaking parameters are very important.