RoboND Project: Where am I

Milton Wong

Abstract—This project explores the process of building robot model using Unified Robot Description Format (URDF) with Gazebo sensors and actuators plugins that can be used for a further simulation. By utilizing ROS packages, such as Adaptive Monte Carlo Localization (AMCL) package and Navigation stack to accurately localize a mobile robot inside a provided map in the Gazebo, and navigate a robot towards the specified goal. For localization, the corner stone of the robot navigation, Adaptive Monte Carlo Localization algorithm was used and compared with the Kalman-Filter based algorithms. Tuning parameters strategy is described in detail and future possibilities discussed.

Index Terms—Mobile robot, Localization, Monte Carlo Localization.

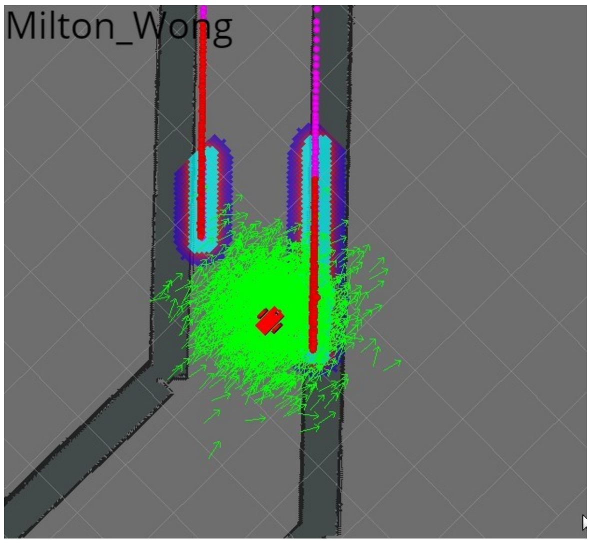

Fig. 1: AMCL localization

1 Introduction

The first important characteristic of a robot is to know its location in a surrounding space so it can reason about neighboring features and build plans for the movement goals. Two common types of localization are recognizes: local and global. Local localization is determining robot pose and orientation based on odometry and perception sensors data before any matching to the map. Global localization, on the other hand, combines output from local localization and matches it to the known map so to place the robot in exact position and orientation on the map. In other word it is describing the translation and rotation from odom frame to the map frame.

The goal of this project is to localize the robot in a given map. The specific tasks can be classified as : i). create a customized robot using URDF. ii). perform localization by AMCL algorithm iii). tuning parameters for better performance. (Fig. 1)

2 Background

Kalman filters [1] are considered as a sufficiently accurate algorithm that works great for small incremental errors that arise due to the sensors errors and motion models discrepancies (noise representation, wheel slippage etc.) like odometry estimation and local robot pose. However Gaussian assumption employed by Kalman filters is restrictive for global localization problem.

Monte Carlo Localization (MCL) [2] algorithms based on sampled particles and thus can represent any distribution function, that is not limited by a Gaussian model and linear case. MCL is also much simpler in calculation and implementation compared to Kalman filters algorithms. MCL is also a good choice when one desires to control memory and resolution of a filter, which is demanding for limited resources such as embedded devices.

3 Simulations

3.1 Standard robot model design

The standard robot was built based on the square platform with two wheels and one caster. Thus, control base is a standard differential drive.

Standard sensors are chosen:

1) Laser Scanner (Hokuyou), placed on the head in order to have a free space within 180 degrees in front of a robot.
2) RGB Camera, placed in front to see whats ahead (however, in this work camera sensors was not used for navigation purposes)

See Fig. 2 for the appearance of standard robot model generated by URDF.

3.2 AMCL Configuration

Adaptive Monte Carlo Localization (AMCL) [2], [3] algorithm applies variable number of particles depending on certainty, which helps to free computational resources. For the given task as small number of particles, e.g. 100. It works with satisfactory. However, for very uncertain scenario, it needs much more particles distributed over the space to represent

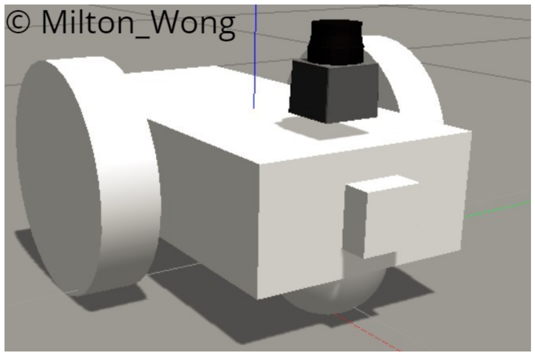

Fig. 2: Standard robot model

the diverse uncertainties. So in this project, we set the following parameters for the range of particles' quantity.

$min\_particles = 50$

$max\_particles = 1500$

The parameter $transform\_tolerance$ was set up to 0.3 seconds by experiment.

We set our robot to start at the center of the map and its pose was set at the origin $(0, 0)$ and orientation as 0 for its initial pose. Here are the parameters for initial pose:

```
<param name="initial_pose_x" value="0.0" />
<param name="initial_pose_y" value="0.0" />
<param name="initial_pose_a" value="0.0" />
```

3.3 Move Base Configuration

Move base configuration includes the common parameters and local/global costmap parameters.

The main parameter $transform\_tolerance$ was set to 0.3 in accordance to the AMCL configuration. Also, parameter $inflation\_radius$ is lowered to 0.45 for better approximation to the robot shape. In $local\_costmap\_params.yaml$, we set $width = 0.5$ and $height = 5.0$.

To unstack, we should set params like min_vel_x , $min\_in\_place\_vel\_theta$ and $escape\_vel$ to overcome the friction and mass inertia.

To stop robot rotation and movement around the goal and not continuously overshoot it, parameters such as goal tolerance are also set as the following in $base\_local\_planner\_params.yaml$:

```
min_vel_x: 0.15
min_in_place_vel_theta: 0.3
escape_vel: -0.25
# Goal Tolerance Parameters
xy_goal_tolerance: 0.15
yaw_goal_tolerance: 0.05
```

4 Results

Present an unbiased view of your robot's performance and justify your stance with facts. Do the localization results look reasonable? What is the duration for the particle filters to converge? How long does it take for the robot to reach the goal? Does it follow a smooth path to the goal? Does it have unexpected behavior in the process?
For demonstrating your results, it is incredibly useful to have some watermarked charts, tables, and/or graphs for the reader to review. This makes ingesting the information quicker and easier.

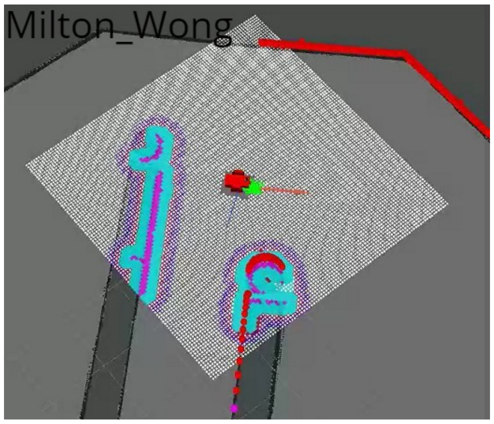

Fig. 3: AMCL localization result

4.1 Localization Results

Fig. 3 shows the final localization result using AMCL package, with the video link: https://youtu.be/7GodTcZbebQ

The robot almost follows the entire desired trajectory towards the goal, and the final orientation of the robot is aligned with the desired direction.

5 Discussion

For AMCL configuration, parameter $odom\_alpha$ was crucial in order to achieve a normal estimation of the robot pose, otherwise, the deviation will be high.

Monte Carlo Localization (MCL) can be used for the household robot with the known map at home or in the office space. However, dynamic objects such as moving people and furniture would pose troubles in the localization due to the changing map, which can be addressed by building dynamics maps of the environment.

Here, we also discuss the trade-offs in accuracy and processing time. i) AMCL number of particle filters: more particles give us more chances to correctly localize robot in unexpected places. But it will takes more time in computation. ii) By increasing **resolution** for global and local cost-maps, we can provide a higher granularity for paths calculation which gives us more accurate planers. Again, with the more cells to compute and search for a path we are increasing our computational load on the system. In summary we have a great control over computational efficiency and accuracy that is useful for deployment on smaller systems.

6 Conclusion / Future work

For the current environment and robot configuration, one of the possible improvement could be to replace a 180 degree LiDAR with a 360 degree LiDAR. It will cover more space and will provide more features for the AMCL algorithms. Other possible improvement will be to add a 3D structural depth sensor to obtain the depth map, which can be useful for another level of feature extraction and/or fine grained semantic object detections (like toys, fruits, missed wedding ring, etc).

References

[1] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[2] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," *AAAI/IAAI*, vol. 1999, no. 343-349, pp. 2–2, 1999.

[3] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.