Where Am I?

Hsin-Wen Chang

Abstract—In this project we will apply Adaptive Monte Carlo localization algorithm utilize ROS packages to accurately localize a designed mobile robot inside a provided map(Jackal race world) in the Gazebo and RViz simulation environments then experiment and test with baseline robot. Working with the Navigation Stack(move base package) define a goal position for robot in the Jackal race world, and the robot will navigate to that goal position aims to solve robotic localization problem. successfully.

Index Terms—Robot, IEEEtran, Udacity, LaTeX, Localization, Adaptive Monte Carlo Localization (AMCL).

1 Introduction

Localization is the challenge of determining robot's pose in a mapped environment. By implementing a probabilistic algorithm to filter noisy sensor measurements and track robot's position and orientation. The robot will moving around taking measurement try to figure out where it can be positioned in a space. With the probabilistic model the robot might have a few guesses as to where it locate and over time it should narrow down it's location. There are four localization algorithms such as Extend Kalman filter localization is the most common Gaussian filter which estimating the state of non-linear models and Markov Localization maintains a probability distribution over the set of all possible position and orientation the robot might be located at. The grid localization is referred to as histogram filter because it's capable of estimating the robot's pose using grids and finally Monte Carlo localization also know as particle filter because it estimate robot's pose using particles.

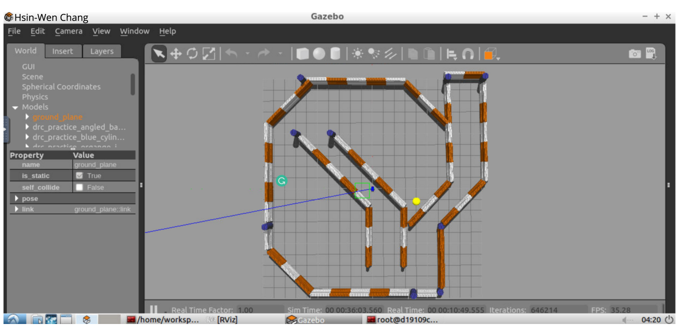


Fig. 1. Robot Revolution.

2 Background

In the book Probabilistic Robotic (Thrun et al. 2005) described there are several approach to help robot perform localization problem from Extend Kalman filter to Marco, to Monte Carlo and finally Grid. In this project, Jackal race world is a local localization problem which has information robot's initial pose relate to static environment collecting sensory information using range finder sensors from environment.

2.1 Kalman Filters

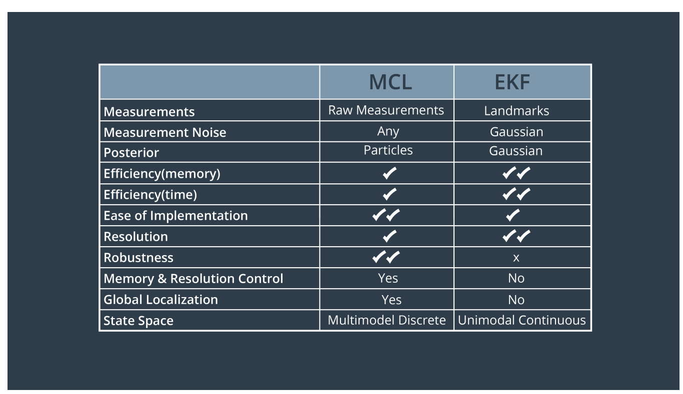
Kalman filters can take data with a lot of noisy or uncertainty in the measurements and provide very accurate estimate of the real value and can do it very fast don't need wait a lot of data to come. The Kalman Filter is applicable to problems with linear motion and measurement functions. This is limiting, as much of the real world is nonlinear.

2.2 Particle Filters

MCL can solve local and global localization problems represents non Gaussian distribution and can approximate any other practical important distribution not limited to linear model.

2.3 Comparison / Contrast

MCL is easy to program compare to EKF. MCL uses particle to localize robot's pose and can approximate almost any state space distribution. MCL uses particles to localize a robot and can approximate almost any distribution. Each particle has a position and orientation by changing the number of particles control computational memory and resolution.



	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. 2. MCL VS EKF.

3 Simulations

By utilize ROS packages to accurately localize a mobile robot inside a provided map in the Gazebo and RViz simulation environments. While Gazebo is a physics simulator, RViz can visualize any type of sensor data being published over a ROS topic like camera images, point clouds, Lidar data, etc. This data can be a live stream coming directly from the sensor or pre-recorded data stored as a bag file. RViz is one-stop tool to visualize all the three core aspects of a robot: Perception, Decision Making, and Actuation.

Over all ROS aspect:

- Building a mobile robot for simulated tasks.
- Creating a ROS package that launches a custom robot model in a Gazebo world and utilizes packages like AMCL and the Navigation Stack.
- Exploring, adding, and tuning specific parameters corresponding to each package to achieve the best possible localization results.

3.1 Achievements

By tuning parameter in costmap common params.yaml the designed robot can localized itself and reach their destination. With navigation goal.cpp node will navigation robot to It's goal.

3.2 Benchmark Model: Udacity Bot

3.2.1 Model design

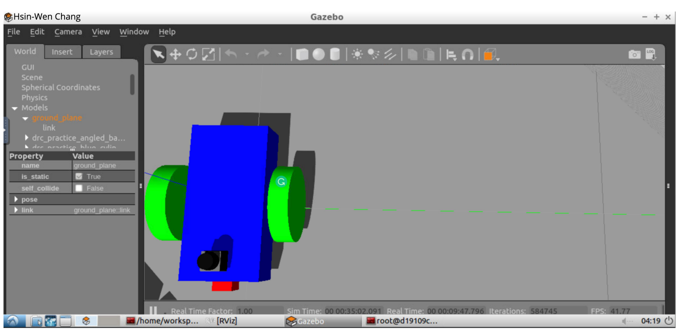


Fig. 3. Robot model with a camera and laser rangefinder.

In this section build a very basic mobile robot model named Udacity bot by creating its Unified Robot Description Format (URDF file). The robot designed geometry and parameters such as collision and inertia are set in configure. The Udacity bot has chassis size [0.4, 0.2, 0.1] with 2 casters provide balance with radius shape 0.0499 and two wheels with cylinder radius="0.1" length="0.05" on-board with camera and laser range finder.

Part	Geometry	size
Left and right wheel	Cylinder	length:0.05, radius:0.1
Chassis	Cube	0.4x0.2x0.1
Back and Front casters	Sphere	radius:0.0499
Camera	box	0.05 x 0.05 x 0.05

TABLE 1
Udacity bot specifications

3.2.2 Packages Used

In Robotics navigation, AMCL package and the Navigation stack package are crucial for localize and navigation robot. AMCL package can dynamically adjusts the number of particles over a period of time, as the robot navigates around in a map. This adaptive process offers a significant computational advantage over MCL.

Udacity Bot package

- config
- launch
- maps
- meshes
- src
- urdf
- worlds
- rviz

3.2.3 Parameters

Localization parameters in the AMCL node such as min and max particles is crucial as we can set from 10 to 100 to observe the performance and memory usage. As the particles increase same as the accuracy will increase. By tuning transform tolerance parameter will fix Costmap2DROS transform timeout indicative of the maximum allowed delay to either be not defined or to be too low for the system to compensate for. This maximum amount of delay or latency allowed between transforms is defined by the transform tolerance parameter. Except transform tolerance parameter several parameter was tested and tuned such as obstacle range, raytrace range and inflation radius. By increasing inflation radius will influence obstacles detect and the obstacles distance between the robot.

Costmap Parameters

Parameter	Global	local
global frame	map	Odom
robot base frame	robot footprint	robot footprint
update frequency	50.0	50.0
publish frequency	50.0	50.0
width	40.0	20.0
height	40.0	20.0
resolution	0.05	0.05
static map	true	false
rolling window	false	true

TABLE 2
Udacity bot specifications

AMCL node Parameters

min particles	100
max particles	5000
laser max beams	30

TABLE 3
Udacity bot specifications

3.3 Personal Model

3.3.1 Model design

By creating another Robot Description Format (URDF file) base on Udacity bot and the same package to perform publisher and subscriber .The designed a robot model named

Hsin bot has chassis size [0.4, 0.2, 0.1] with 2 casters provide balance with radius shape 0.0499 and two wheels with cylinder radius="0.1" length="0.05" on-board with camera and laser range finder which the camera is mount on front chassis has size [0.05,0.05, 0.05] and add three long and short back shield with size [0.1, 0.02, 0.12] and [0.1, 0.3, 0.02].

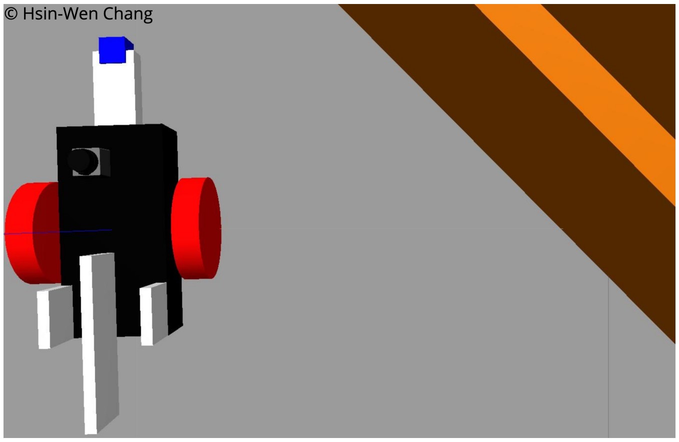


Fig. 4. Hsin bot right front.

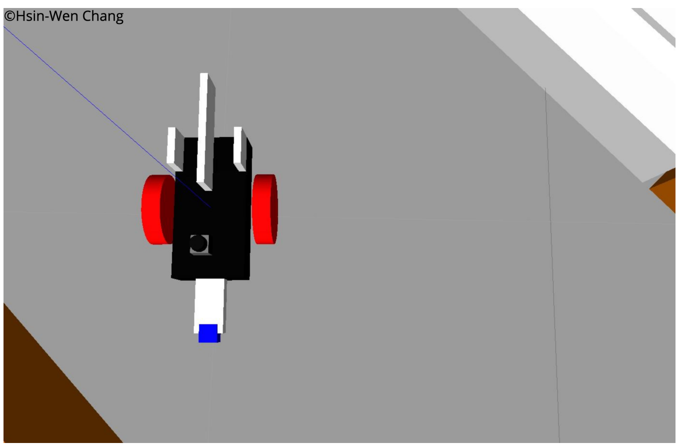


Fig. 5. Hsin bot top front.

3.3.2 *Packages Used*

HsinBot package

- config
- launch
- maps
- meshes
- src
- urdf
- worlds
- rviz

3.3.3 *Parameters*

Most of the configuration of parameters in Hsin bot is base on Udacity bot except laser max beams to 20 in each scan to be used when updating the filter. Minimum allowed 50 number of particles. Maximum allowed 200 number of particles.

Part	Geometry	size
Left and right wheel	Cylinder	length:0.05, radius:0.1
Chassis	Cube	0.4x0.2x0.1
Camera	box	0.05 x 0.05 x 0.05
Back and Front casters	Sphere	radius:0.0499
Back shield1	box	0.1 x 0.3 x 0.02
Left Back shield	box	0.1 x 0.02 x 0.12
Right Back shield	box	0.1 x 0.02 x 0.12

TABLE 4
Hsin bot specifications

Costmap Parameters

Parameter	Global	local
global frame	map	Odom
robot base frame	robot footprint	robot footprint
update frequency	50.0	50.0
publish frequency	50.0	50.0
width	40.0	20.0
height	40.0	20.0
resolution	0.05	0.05
static map	true	false
rolling window	false	true

TABLE 5
Hsin bot specifications

4 Results

After parameter tuning and testing both robots their particle filters will convergence after a few seconds through Navigation stack both robots were able to reach their destination. However when encountering obstacles both robots will slightly paused reaction for the particle filters transform but will Navigation it's way out and toward their destination eventually.

4.1 Localization Results

4.1.1 *Benchmark*

At very begging, the particle filters(green arrows) will emitted to the space from the Udacity bot while navigation to it's goal, the particle filters will convergence into the direction it is heading for extremely like human perception then reaction with focus. Fig 7 is Udacity bot enter it's finish line photo.

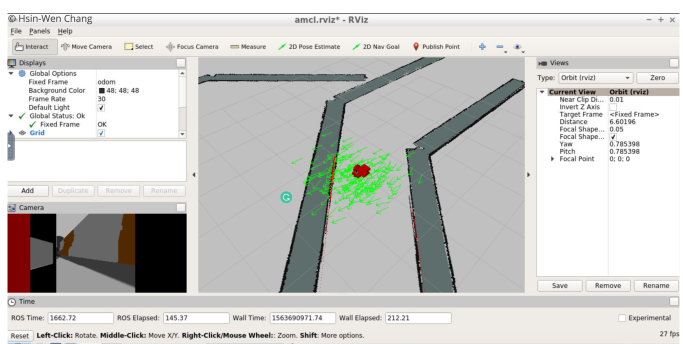


Fig. 6. Udacity bot model with a camera and laser rangefinder.

4.1.2 *Student*

At the very beginning, the particle filters(green arrows) will emitted to the space from the Hsin bot while navigation

AMCL node Parameters	
min particles	50
max particles	200
laser max beams	20

TABLE 6
Hsin bot specifications

costmap common params	
obstacle range	1.2
raytrace range	1.2
inflation radius	0.4
observation sources	laser scan sensor

TABLE 7
Hsin bot specifications

to it's goal, the particle filters will convergence into the direction it is heading for extremely like human perception then reaction with focus. Fig 8 is Hsin bot enter it's finish line photo.

4.2 Technical Comparison

The difference between Hsin bot and Udacity bot is the layout of Hsin bot has back shield and longer front chassis where mount on camera can observe more forward obstacles to response earlier. The back shield design is imitate from rocket but might need more test because the aerodynamic in high speed might lead the robot drift for stable the robot but in lower speed it might not need for consideration. By tuning parameters in move base, amcl lauch file and costmap common params file to update particles in higher frequency.

5 Discussion

The Hsin bot is a little bit faster than Udacity bot base on it's layout design and higher update particle filters frequency rate. AMCL package can quickly retrieve locality of the robot but still need fine tune the Navigation stack and move base parameter. In the 'Kidnapped Robot' problem base on AMCL use number of particles to adapt certainty of robot localization not on landmark but on laser base maps, laser

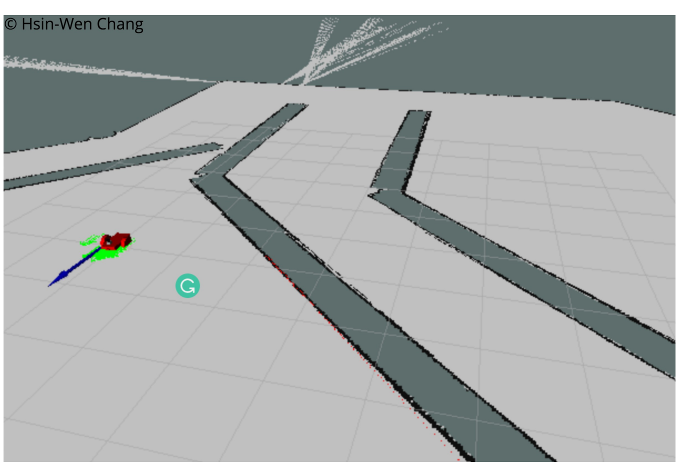


Fig. 7. Udacity bot reaching the goal position.

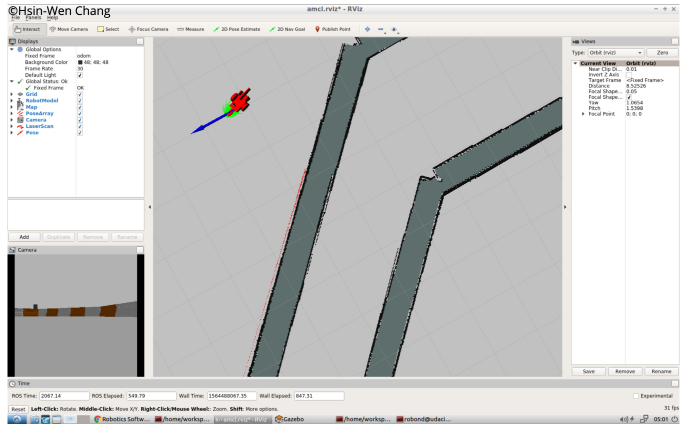


Fig. 8. Hsin bot enter the goal.

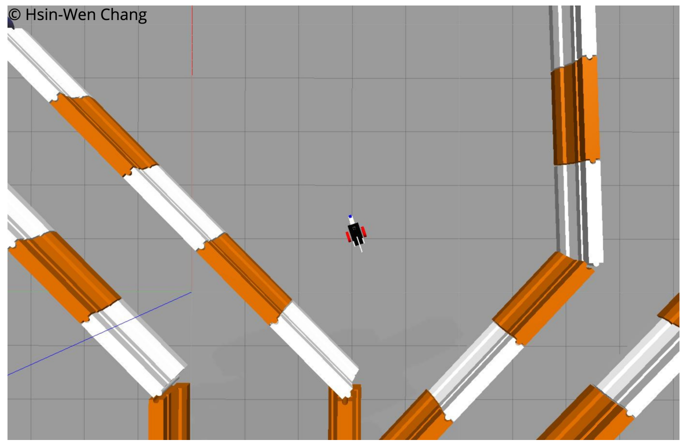


Fig. 9. Hsin bot enter the goal.

scan to transform via message and topic to out put pose estimation. Base on no need rely on landmark, AMCL will have advantage over EKF in non statistic environment(unknown map).

6 Conclusion / Future work

For further improve localization accuracy with more computational resources need to update particle filters frequency slightly higher to gain higher accuracy in localization problem. A LIDAR can be add to provide more details map of the environment surround the robot.

References

[1] Robust Monte Carlo Localization for Mobile Robots Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dellaer, in Artificial Intelligence, Summer 200.

[2] http://wiki.ros.org/amclParameters