Where Am I Writeup

Francisco Eduardo Ortiz

Abstract — The project was developed for robot location resolution using Monte Carlo algorithm in a simulated Gazebo / Rviz environment. A two-wheel robot was designed to traverse a map provided for execution within a ROS structure, using the AMCL package. If necessary adjustments are made to the cost map settings and the planner's home base, the robot can pinpoint and navigate to the target.

Index Terms — Udacity, Localization, Monte Carlo Algorithm, AMCL, EKF, Gazebo, Rviz, ROS.

1 INTRODUCTION

In robotics location is an interesting challenge as it is a capability that allows the robot to move accurately. Location challenges can be divided into:

- **Local location (pose tracking):** the robot knows its initial pose and needs to follow its pose as it moves.
- **Global Location:** pose of the robot is unknown and needs to determine its pose in relation to the inertial reference map, the uncertainty being greater than the local.
- **Robot kidnapped:** Robot's initial pose is unknown, and can be hijacked at any time, or moved to another location on the map.

The resolution of a hijacked robot problem also enables resolution in case of track loss or incorrect calculation of its pose.

In this project, the challenge of local location will be approached, via ROS packages using the AMCL package, through a map provided (Fig. 1).

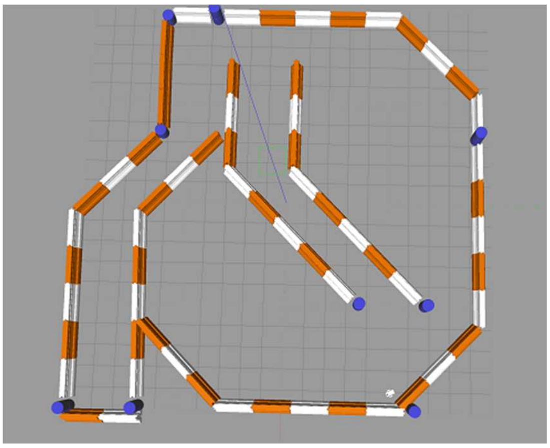

Fig. 1: Map Fornecido.

2 BACKGROUND / FORMULATION

2.1 Localization Problem And Algorithms

Localization resolution by a robot can be approached by the following common algorithms: Markov localization, Grid localization, Kalman Filter localization, and Monte Carlo Localization. In this work we explore a static environment map, and the robot can be located by means of a combination of measurements and movements by sensors, requiring the implementation of noise through a filtering, so we will approach the following algorithms:

- Extended Kalman Filter (EKF): is an algorithm that models the uncertainties of measurements of robotic sensors and motions as Gaussian noise, and uses iterations to update measurements and predict states to decrease variances / covariates of estimated quantities, which may include velocity, pose, orientation.
- Monte Carlo Localization (MCL) or particle filter: uses particles of varying poses to represent the probability distribution of the robot pose and re-samples of the particle set by the probability of each particle based on measurements of reference points each time the robot moves.

	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

Tab. 1: Algorithms comparison between MCL and EKF.

3 DATA ACQUISITION

3.1 Simulation

For simulation a robot was built in the ROS framework and it was executed and displayed on Gazebo and Rviz. The AMCL package was used for Navigation Stack, and to enable the robot with localization capabilities, along with a C ++ navigation node. All the work was developed in the Udacity workspace.

3.2 Objetivo

The robot achieved its navigation goal through the adjustments made to the costmap and base local planner parameters. The motion process was developed by the node in C ++.

3.3 Model

With the Udacity example model developed in URDF format Table 2 shows the specifications and in Figure 2 the final model.

Robot		
Part	**Geometry**	**Size**
Chassi	Cube	0.4 x 0.2 x 0.1
Back and Front Casters	Sphere	0.0499 (radius)
Front Wing	Cube	0.2 x 0.3 x 0.02
Left and Right wheels	Cylinders	0.1 (radius), 0.05 (length)
Sensors		
Camera Sensor	Link Origin	[0,0,0,0,0,0]
	Shape-Size	Box – 0.05 x 0.05 x 0.05
	Joint Origin	[0.025,0,0.025,0,0,0]
	Parent Link	chassis
	Child Link	camera
Hokuyo	Link Origin	[0,0,0,0,0,0]
	Shape-Size	Box – 0.1 x 0.1 x 0.1
	Joint Origin	[-0.05,0,0.1,0,0,0]
	Parent Link	chassis
	Child Link	hokuyo

Tab. 2: Robot and Sensors Specifications.

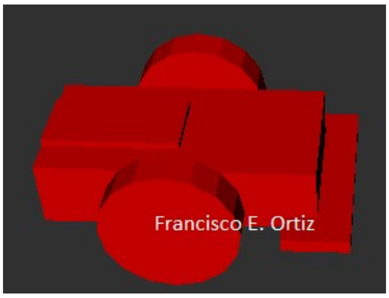

Fig. 2: Robot Visual.

3.3 Packages

The Navigation Stack and AMCL package was used for the project with the following structure in ROS:

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

3.3 Parameters

The parameters with the best results for minimum and maximum particles were 10 and 200, respectively, because adjustments with higher values would begin to directly impact the computational performance. It was also necessary to adjust the parameters of transform tolerance, inflation radius, robot radius and obstacle range, and it is presented in Tables 3 and 4.

Costmap Parameters		
Parameter	**Global**	**Local**
global_frame	map	odom
robot_base_frame	robot footprint	robot footprint
width	10.0	5.0
height	10.0	5.0
resolution	0.05	0.05
static_map	true	false
rolling_window	false	true

Tab. 3: Global and Local Costmap Parameters.

AMCL Node Parameters	
min particles	5
max particles	20
initial_pose_x	0
initial_pose_y	0
initial_pose_a	0
odom_model_type	diff-corrected
Costmap Common Parameters	
obstacle_range	4.0
raytrace_trace	4.0
transform_tolerance	0.2
update_frequency	10.0
publish_frequency	5.0
inflation radius	0.6
xy_goal_tolerance	0.05
yaw_goal_tolerance	0.01
always_send_full_costmap	true
observation_sources	aser_scan_sensor
Base Local Planner Parameters	
holonomic_robot	false
holonomic_robot	true
pdist_scale	1.0
gdist_scale	0.4
occdist_scale	0.01

Tab. 4: AMCL, Costmap and Base Parameters.

4 Results

4.1 Simulation

The results were reached after adjustments in the parameters, test results and simulations. The particle models converged after the start, it was found during the tests that the robot traversed with few interruptions or collisions with obstacles and successfully achieved their goal, adequately resolving the localization problem in a reasonably small time interval. The results are shown in the figures below.

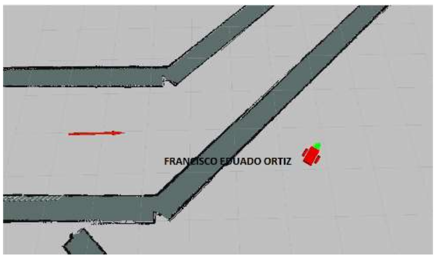

Fig. 3: Starting Simulation.

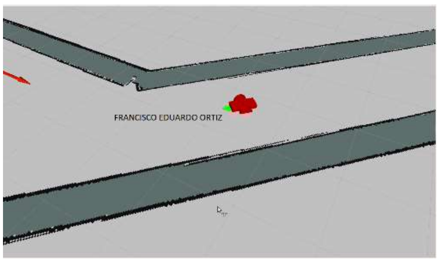

Fig. 4: Particles begin to converge.

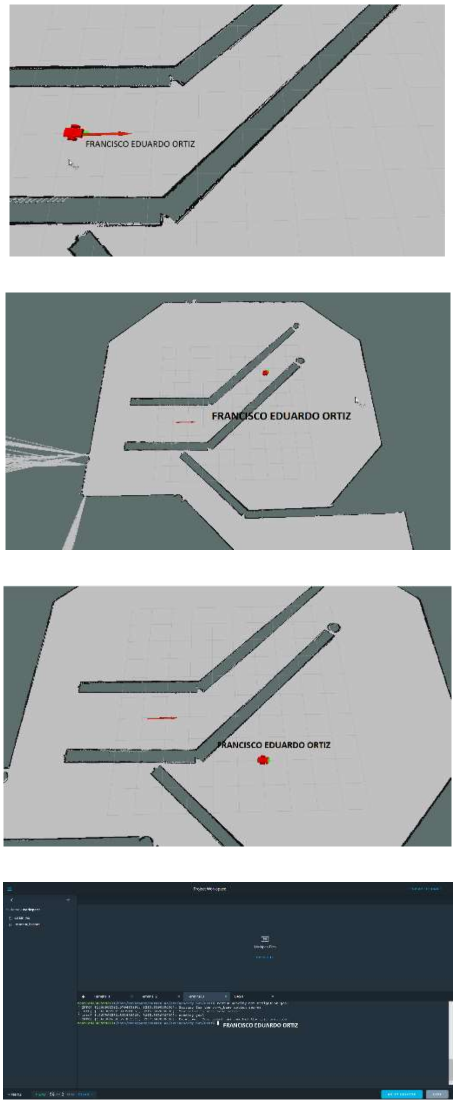

Fig. 5: Reaching Designated Goal.

5 Discussion

5.1 Simulation

In general with improved configuration it is possible that the results and accuracy will become more efficient for localization along the trajectory. The AMCL it becomes much better in closed and known space environments, but it manages to solve the problem of sequestered robot with a greater computational power.

6 Conclusion / Future work

In this work we can verify the power and ease of implementation of the AMCL framework. In future works will be realized real implementations and with a view to the use of the technique in more complex systems, such as drones.

References

[1]UKF-Wikipedia, https://en.wikipedia.org/wiki/Kalman_filter#Unscented_Kalman_filter. Last accessed 17 June 2019.

[2] Taylor series Wikipedia page, https://en.wikipedia.org/wiki/Taylor_series. Last accessed 17 June 2019.

[3] Documentation Robot Pose EKF, http://wiki.ros.org/robot_pose_ekf. Last accessed 17 June 2019.