RoboND Project: Where am I

Chai Changkun

Abstract—In this project, two differential robot base were built from scratch into ROS package. These robots can locate its position using AMCL(Adaptive Monte Carlo Localization) algorithm and navigate around in the simulated jackal race world. The experiment was operated in Gazebo simulator. After tuning AMCL parameters, the simulation shows accurate localization results and robust performance.

Index Terms—Robot, IEEEtran, Udacity, Localization, Simulation.

1 Introduction

For any mobile device, the ability to navigate in its environment is important. Avoiding dangerous situations such as collisions and unsafe conditions (temperature, radiation, exposure to weather, etc.) comes first, but if the robot has a purpose that relates to specific places in the robot environment, it must find those places. This article will present an overview of the skill of navigation and try to identify the basic blocks of a robot navigation system, types of navigation systems, and closer look at its related building components. Robot navigation means the robot's ability to

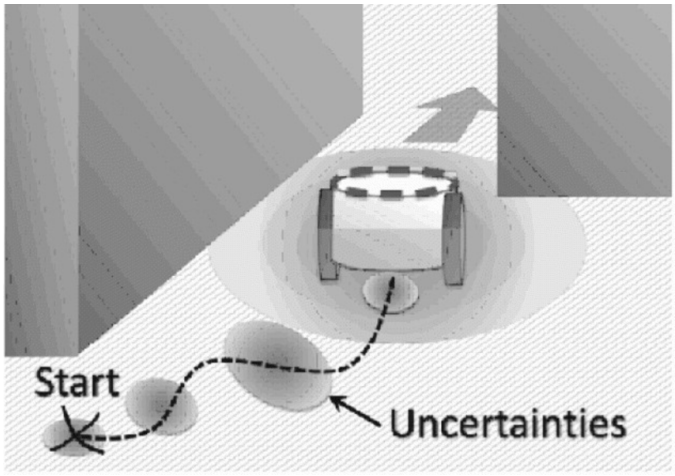

Fig. 1. Robot Revolution.

determine its own position in its frame of reference and then to plan a path towards some goal location. In order to navigate in its environment, the robot or any other mobility device requires representation, i.e. a map of the environment and the ability to interpret that representation.

In this project, a project called udacity_bot is built, two differential robots which contains links and joints are established in the urdf file. The simulation is done in Gazebo, which creates the robot and a jackal race world. The localization function is provided by AMCL package, which is a widely-used localization ROS package. The results showed a good localization accuracy and stable performance.

2 Robot Model

A new differential robot model is built in urdf file. It consists two wheels on the front of the bock and one caster wheel on the back. A hokuyo laser on top and a realsense camera at the front are equipped.

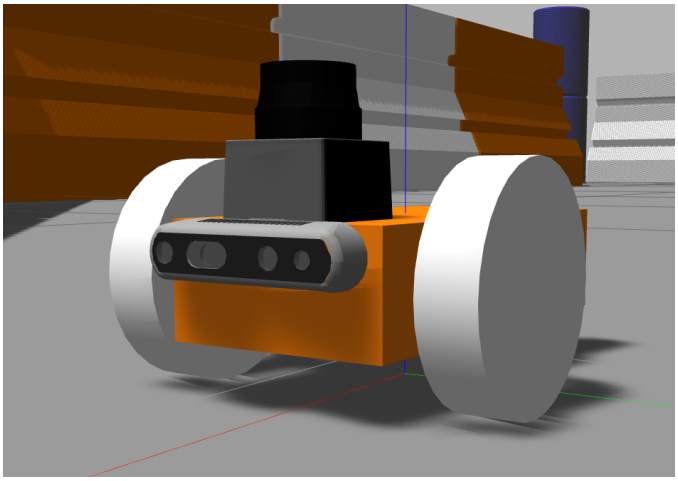

Fig. 2. Robot Model in Gazebo

Though the robot has 3 wheel(one driven wheel and two action wheels), it is actually a differential model, driven by the two.

To make the launch more convenient, the robot model(robot_description param loaded by urdf_spawner) and gazebo simulation are included in one launch file, as well as RViz. And it successfully launches these nodes.

3 Localization

In order to localize itself and make trajectory plan, 3 ROS packages are used: amcl, move_base and map_server .

amcl helps to locate the robot by comparing the laser scan with the known map.

move_base navigates the robot following the right path.

map_server is used for provide the global map of the jackal race world which has been provided.

Each node above contains a set of parameters as can be modified in amcl.launch file.

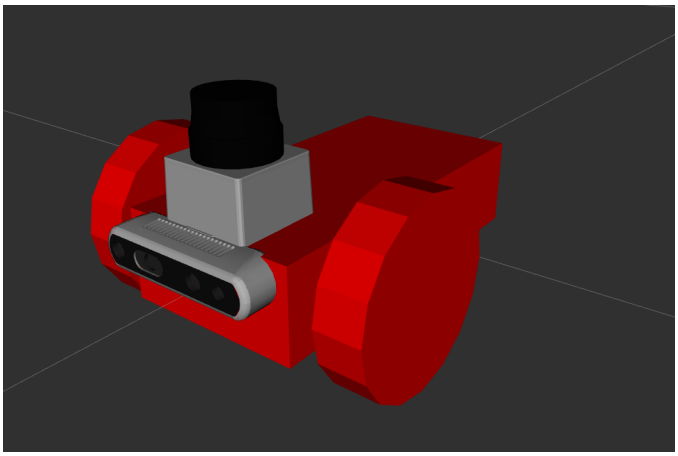

Fig. 3. Robot Model in RViz

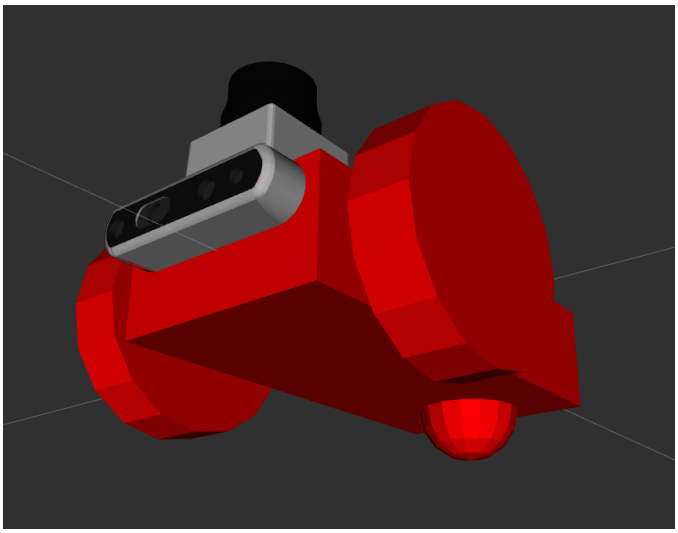

Fig. 4. Robot Model bottom view in Rviz

3.1 amcl

amcl is a probabilistic localization system for a robot moving in 2D. It implements the adaptive (or KLD-sampling) Monte Carlo localization approach (as described by Dieter Fox), which uses a particle filter to track the pose of a robot against a known map.

- odom_frame_id: odom
- odom_model_type: diff-corrected
- base_frame_id: robot_footprint
- global_frame_id: map
- min_particles: 10
- max_particles: 30
- kld_err: map
- update_min_d: 0.25
- update_min_a: 0.25
- resample_interval: 1
- transform_tolerance: 0.02
- laser_max_beams: 50
- laser_max_range: 12
- laser_likelihood_max_dist: 1.5
- laser_z_hit: 0.8
- lazer_z_rand: 0.2
- odom_alpha1: 0.02
- odom_alpha2: 0.02
- odom_alpha3: 0.02
- odom_alpha4: 0.02

If the particle numbers are set too large, it will be very slow computing. In the simulation, the odom covariance is very small(no place to set the noise), it is assumed to be precise. So the particle nums and odom_alpha can be set less.

3.2 move_base

The move_base package provides an implementation of an action (see the actionlib package) that, given a goal in the world, will attempt to reach it with a mobile base. The move_base node links together a global and local planner to accomplish its global navigation task. There are a large number of parameters in move_base node. Some important parameters are picked as follows:

- base_global_planner: navfnNavfnROS
- base_local_planner: base_local_plannerTrajectoryPlannerROS
- robot_radius: 0.1
- inflation_radius: 0.2
- xy_goal_tolerance: 0.05
- yaw_goal_tolerance: 0.05

The move_base has two path planner, a global planner and a local planner. The navfnNavfnROS is chosen as the global planner and ase_local_plannerTrajectoryPlannerROS is chosen as the local one. These two helps the robot to navigate across obstacles and reach the goal. The robot radius is set to be 0.1 which is roughly the shape of the robot. To make it away from obstacle, inflation radius is set 0.2. And the acceptable position and orientation tolerance is set 0.05.

3.3 map_server

map_server provides the map_server ROS Node, which offers map data as a ROS Service. It also provides the map_saver command line utility, which allows dynamically generated maps to be saved to file.

The map of jackal race world is loaded as the static map.

4 Localization Accuracy

The performance of robot localization is very good. As is shown below, at the start, the particles are separated around the robot. After the goal set, the robot makes a plan and begins to move, then the particles begin to converge. Every particle represents a possible position and orientation, and most of them are very close to base_link point.

The process of robot moving and localizing is shown below.

5 Discussion

The tuning of parameters in amcl and move_base is quite time-consuming. The final result depends on many parameters. My robot has better performance, because of better amcl parameter settings, especially the particle numbers and odom alpha. My robot is around half the size of udacity bot, with left wheel and right wheel setup moved forward a little. The change doesn't had any affect on the accuracy.

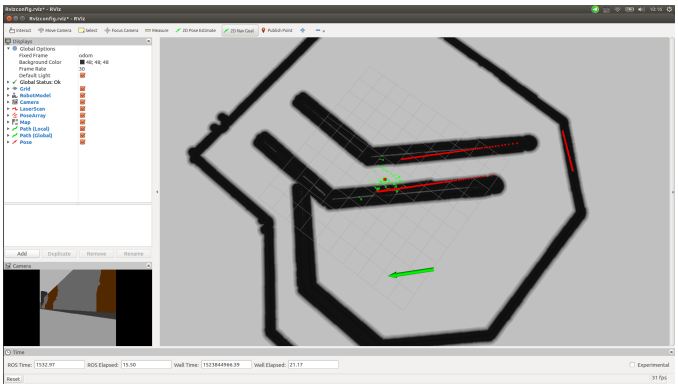

Fig. 5. set goal

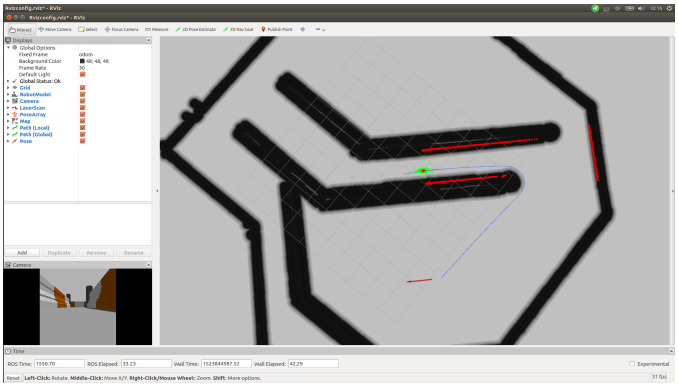

Fig. 6. start to move

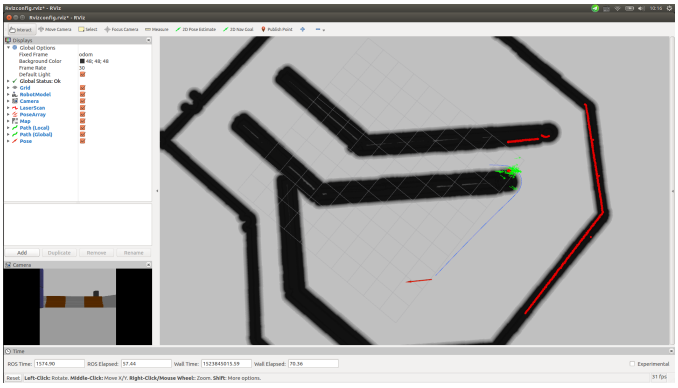

Fig. 7. turning

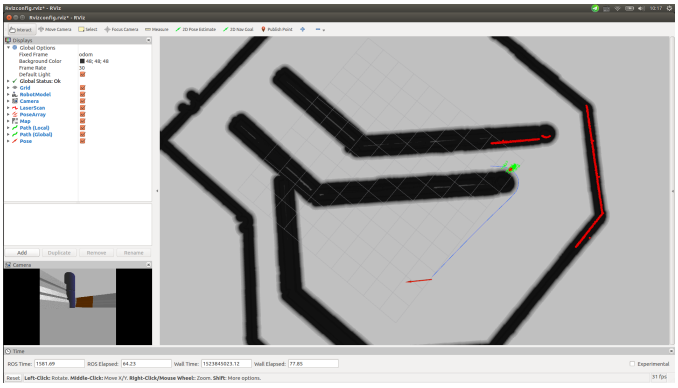

Fig. 8. particles converging

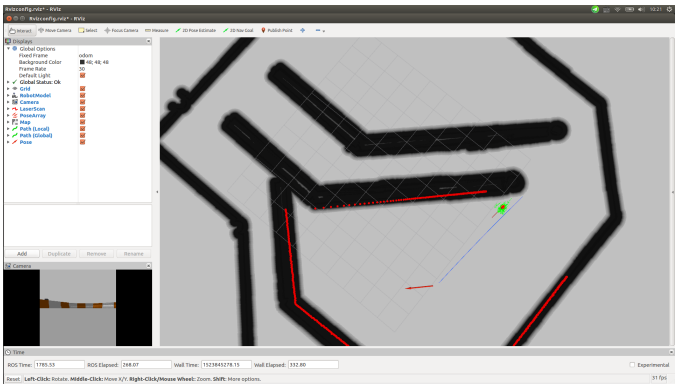

Fig. 9. moving

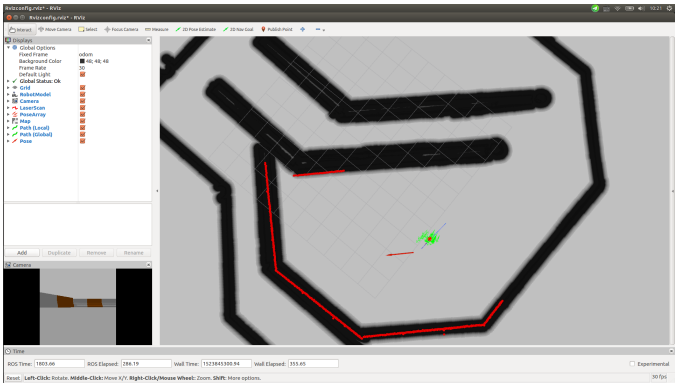

Fig. 10. moving

6 Conclusion / Future work

To summarize, the project built a two wheel robot with laser and camera. It can do localization and path-planning in a known map with amcl node and move_base node running background. The result showed that it always locates correctly. For a lack of time, more sensors cloud have been added such as RGBD camera, ultrasonic, IMU and infrared ranger. These plugins are available in Gazebo simulation and would be added to my robot. In addition, sensor fusion could be implemented. robot_pose_ekf packages can takes odom, visual odom and IMU as input, and outputs the combined odom, which would be much more precise.

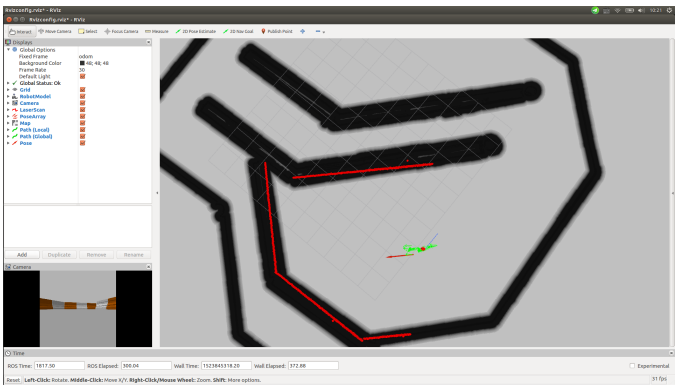

Fig. 11. goal reached