Lab0 - Introduction to ROS

Probabilistic Robotics

Songyou Peng
psy920710@gmail.com

February 28th, 2016

1 Questions Answer

In the lab, there are two questions listed in the lab0 pdf. They will be answered respectively.

1. The name of the topic used to send velocity commands to the turtle and the type of these messages
2. The name of the topic where the turtle position is published at, and the type of these messages

For the first question, the topic `/turtle1/cmd_vel` is used to send velocity to the turtle. `/turtle1` is the name of the current turtle. The message used is `geometry_msgs/Twist`, which contains the linear and angular velocity of the turtle in x, y, z directions. In our case, a turtle can only linearly move in the x direction and rotate in z direction. In order to let turtle move towards the goal, we should write a publisher and send messages containing current linear and angular velocity to the turtle.

For the second one, the position topic published by the turtle is `/turtle1/pose`, and the type of messages are `turtlesim/Pose`, which contain the 2D coordinates and angle of the turtle. In order to control the turtle, we should know the position information so a subscriber of this position topic is needed.

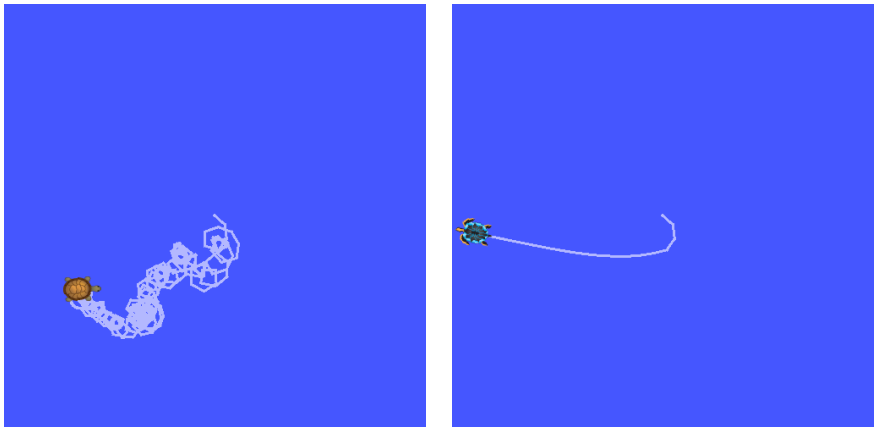
2 Problems faced and solutions

Due to the fact that most of us are new to ROS or even Python, there is no doubt that we faced many different problems. Here we would like to point out two main problems during coding.

First of all, if we want the turtle to reach the goal point accurately, the linear and angular velocities should be set very small. If increasing these two velocities, the turtle will easily miss the angle pointing to the goal or pass the goal point.

In order to increase the speed without losing the accuracy, three changes have been applied. First we decrease the updating time. Before, publishing velocity commands every 0.3 seconds, now we use 0.05 seconds. In this case, the turtle would get the velocity messages every 0.05 seconds, which significantly increases the accuracy. Besides, instead of giving priority to the rotation, we let turtle rotate and go straight simultaneously. Moreover, the linear velocity is proportional to the distance between turtle and goal, and angular velocity to the angle difference between turtle angle and the angle to the goal. This dramatically increases the moving robustness of the turtle.

After applying the above changes, the turtle works well in most cases. However, we notice that sometimes the turtle keeps rotating crazily and could not reach the goal point. The situation is shown in the Figure 1(a).



(a) Before (b) After

Figure 1: Before and after wrapping the angles, goal coordinate is $(1, 5)$

Our angular velocity is proportional to the angle difference between the angle from the turtle to the goal and the angle of the turtle itself. We find out that the reason of the problem is that, the angle difference hasn't been wrapped to the range $[-\pi, \pi]$. The turtle "dances" because the turtle has a wrong large velocity. After wrapping, the result is in Figure 1(b), which performs very acurately.