A manipulation method based on odometry and MoveIt

Zeng Qingyi 1611472

Abstract: this document is a report for the Robotic Software Engineering course project. It explains how to grasp a raincoat in the "farewell" task in Robocup2019.

1. Introduction:

Our team intended to develop a solution to the Farewell task, which is a challenge for RoboCup 2019. Here is the scenario: some guests are tired, so they call the robot to retrieve their coat. It's raining outside and there is only one umbrella, so the robot takes the guests one by one to their cab and returns with the umbrella. However, due to the limitations of the manipulator, it is very difficult to hold an umbrella steadily while follow the guest. So we simplified the umbrella part. Therefore, our process of completing the task is shown as below.

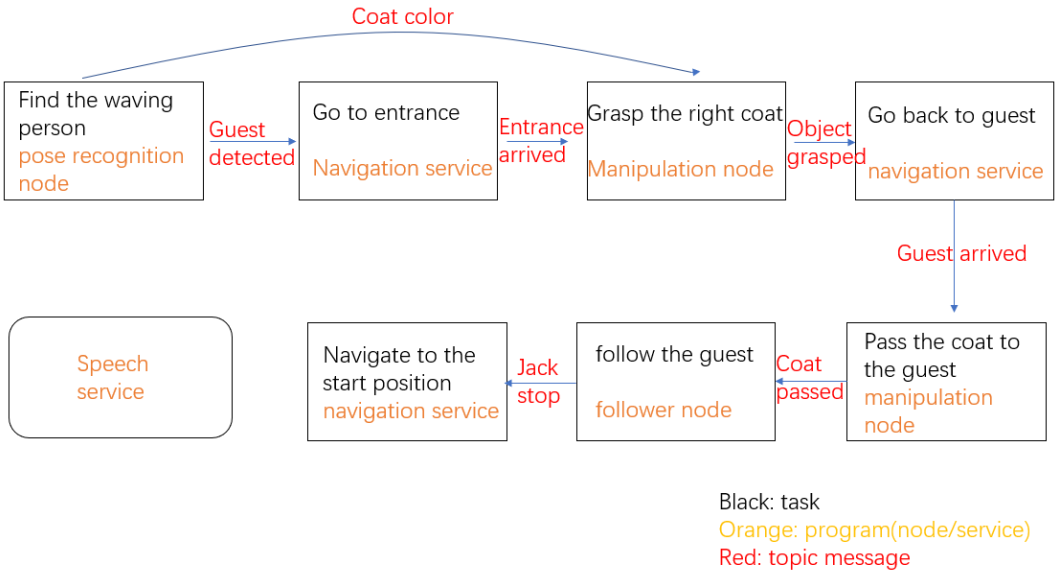

Fig. 1: farewell work flow

2. Overall system design:

2.1 Hardware design

Fig. 2: robot design(front view) Fig. 3: robot design(side view)

Based on the original design of TurtleBot2, we added a depth camera Astra for person and object detection, a microphone for voice recognition, a voice box for sound play and a manipulator for grasping task. What's more, since there is no much room left for a laptop, we used the Nvidia Jetson TX2 as master, and 3PCs as slaves to do all the control.

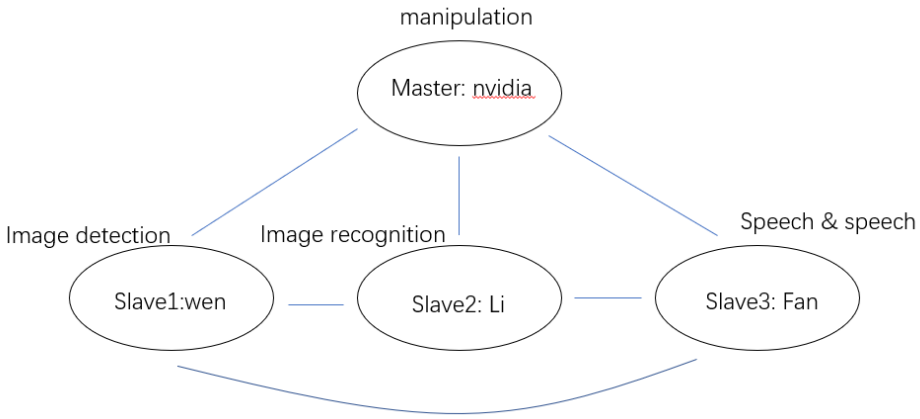

Fig. 4: communication network

2.2 Software design

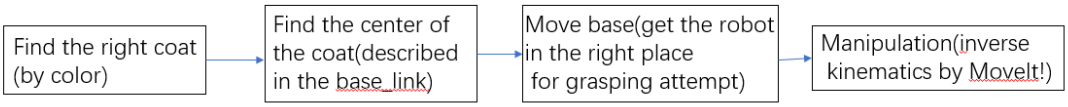

Fig. 5: One attemp to grasp

In the group task, I'm mainly responsible for grasping the right coat and pass it to the guest. The problem is to combine vision and TurtleBot arm. So I used odometry as a bridge. First I can find the right coat by some computer vision method, the color of the coat depends on who is going to leave. Then move the robot to a place suitable for grasping, and then try to grasp.

3. Technical details

3.1 Object detection

In this part, we try to get the position of candide coat described in the base_link frame. There are two candidate coats, the feature of person determines which one to choose. We use the Drif algorithm to divide foreground and background, then we can get the position of coats on the RGB image view. Furthermore, we listen to the TF transformation between astra frame and base link, then transform the point from RBG image raw to exact 3D position.

3.2 Base adjustment

While controlling TurtleBot with fixed time and fixed speed cause considerable error, the odometry method is very accurate because of the existence of the underlying hardware such as the

code disk and counter. Therefore, I chose odometry to do some adjustment.

The idea is take the object position as the origin and make the robot facing the target.

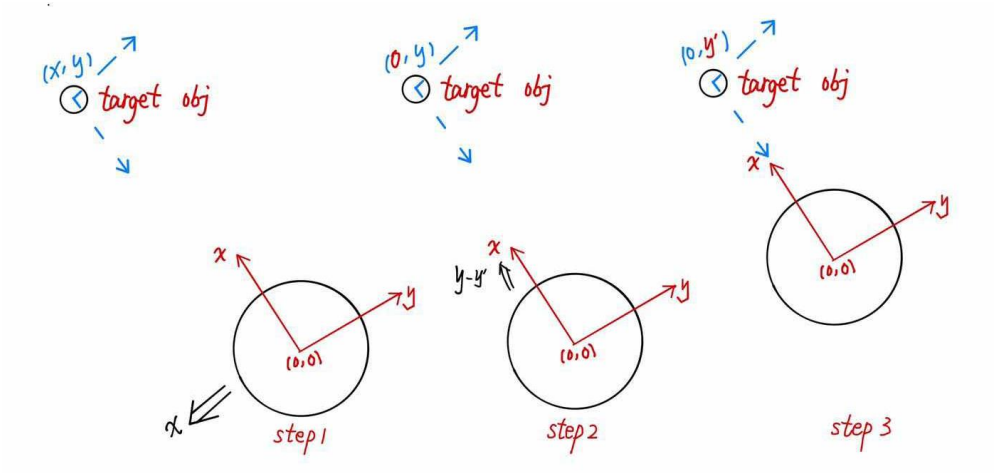

Fig. 5: method for base adjustment

In the book “ROS by example” there are two ways to move the robot base, one is by time and velocity and the other is by odometry. The latter is much more precise than the former because of its closed-loop control. The turtlebot move a very short distance each time and detect if its travel mileage achieves the goal distance.

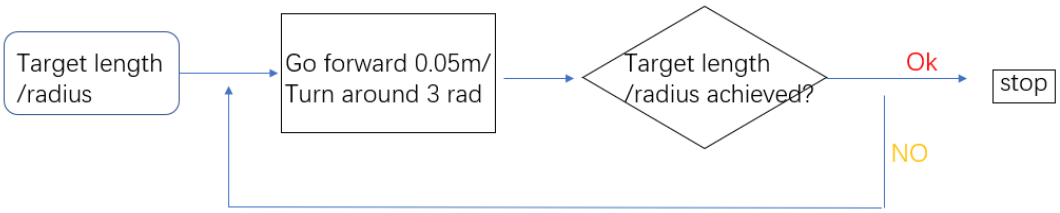

Fig. 6: odometry move base

3.3 Gasping

To solve the inverse kinematics problem, the MoveIt! open-source software is chosen because of its practicable. This package provides a smooth way for the robot arm when it’s given a target position. In the forward kinematics demo, it also perform better than just controlling motor by publishing topics.

Fig. 7: grasping

4.Conclusion

Manipulation is an important part in the farewell challenge and I've met many obstacles during the developing process. For example, grasping a coat is not like grasping a bottle, it depends a lot on the selection of target position on the coat. Also the coat must be light enough otherwise the motor will be overloaded. And the point cloud provided by Astra are not accurate and reliable enough, which leads to graping failure at last. So there are still some work to do to improve the robustness of this method.