# Technical Report: Turtlebot Obstacle Avoidance

The first step of the program is to detect the walls of the hallway and the human within the hallway. The second step is to eliminate the hallway using Hough transform in order to detect the human without the interference of the walls. The third step is to track the movements of the human including speed and anticipated destination. The last step is to move the robot accordingly.

The first part of the program is to detect the hallway and the person in the hallway. Every scan of the kinect camera, starting from -30° to 30°, provides 640 points of range data. An example is shown in Figure 1.

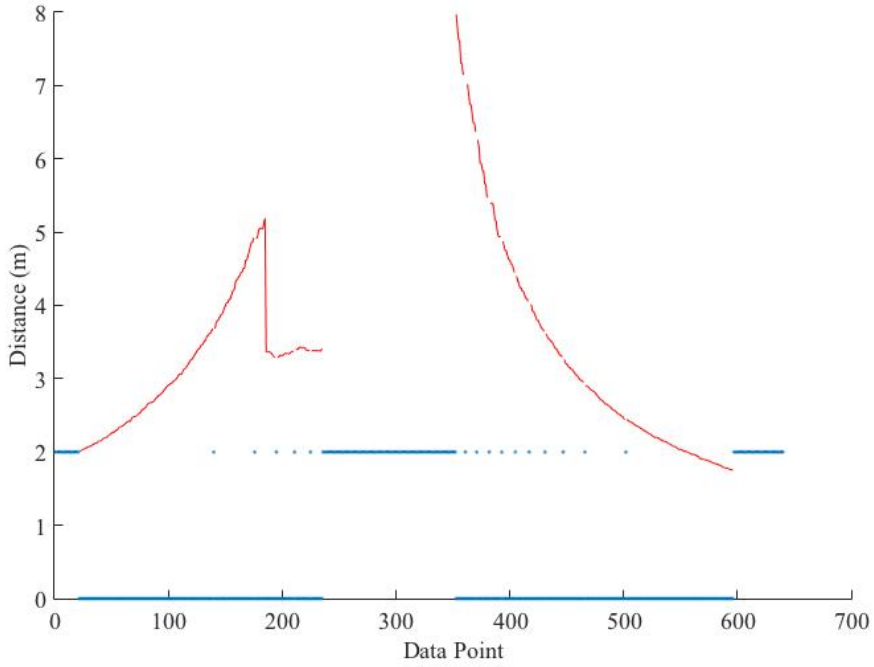


Figure 1

In this example, the person is standing near the left wall and about 4 meters in front of the turtlebot. Data in cartesian coordinate is shown in Figure 2.

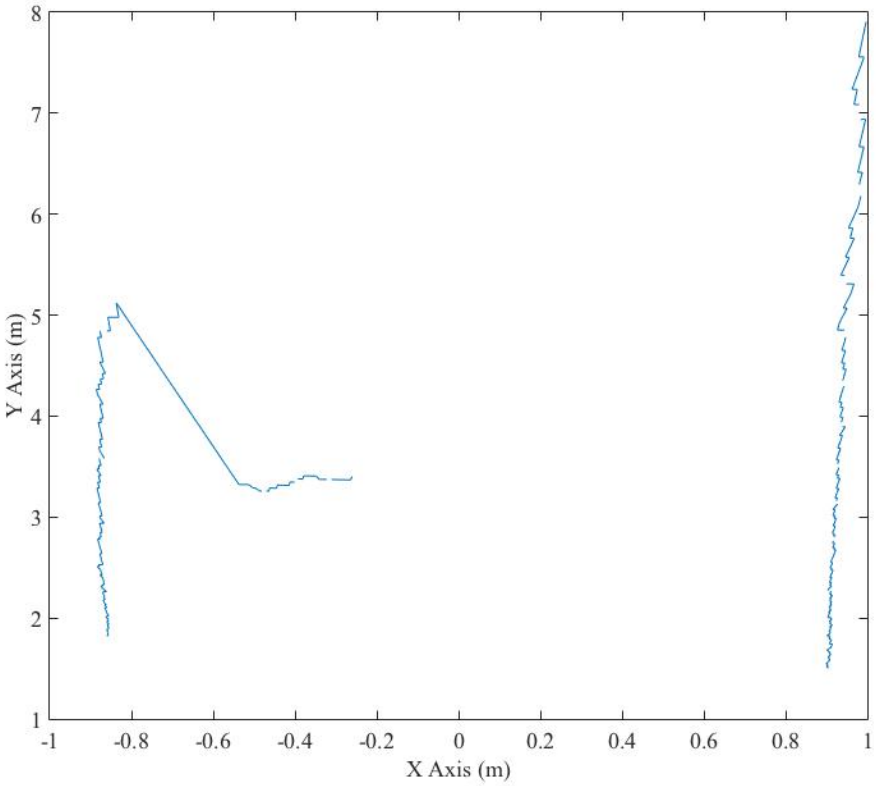


Figure 2

The person is represented as the horizontal line near the left vertical line. As the data points representing the walls are easier to distinguish, we decide to detect them first by using hough transform. The Hough transform is a feature extraction technique. In this case, the feature we are looking for is the vertical straight line from the walls of the hallway. The equation we use to describe the walls is shown:

$$\rho cos(\theta - \alpha) = r \quad (1)$$

in which $\theta$ and $\rho$ are the data points from laser scan; $\alpha$ and $r$ are the parameters being voted as shown in Figure 3.

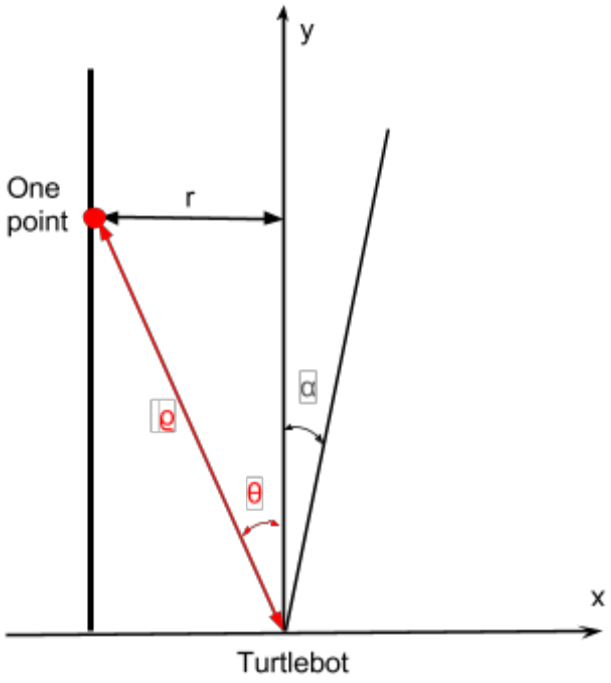


Figure 3

We create a 25 by 18 poll table: $\alpha$ ranges from -22.5° to 22.5° with 18 steps and $r$ ranges from 0 meter to 2.5 meter with 25 steps. By plug in $\theta$ and $\rho$ for every point from the laser scan in the equation, corresponding $\alpha$ and $r$ are voted in the poll table. Figure 4 is a poll table for the previous example.

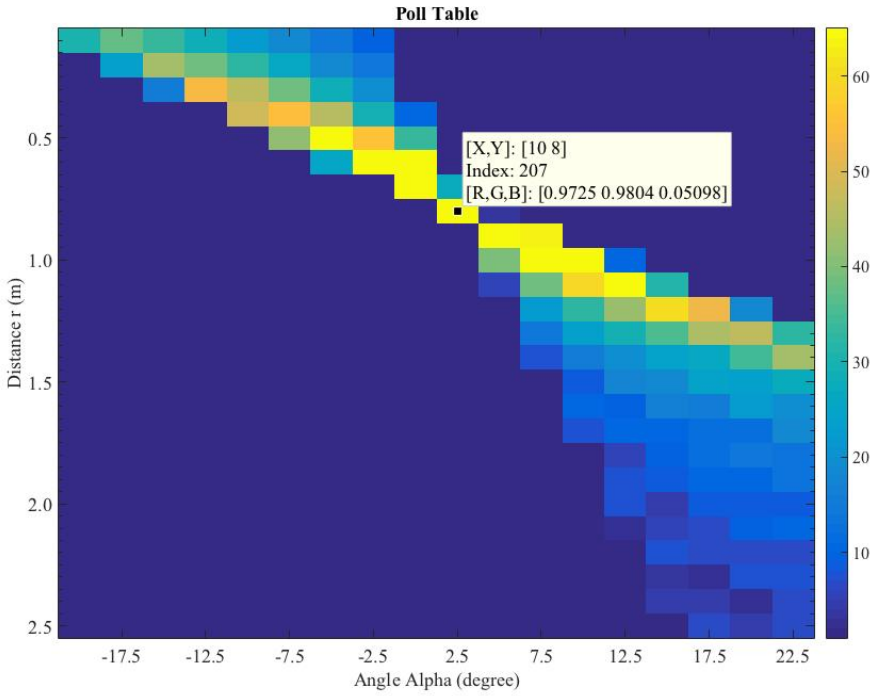


Figure 4

The highest vote describes the right wall is about 0.8 meter from the center with about 2.5 degree tilt.

After the voting process, we pick the $\alpha$ and $r$ values with the highest vote and eliminate all the points from this straight line in the scan data. This is done by checking whether Equation 1 can be satisfied for $\theta$ and $\rho$ of each data point. We also set a tolerance for the straight line so that any data point with slight offset from the straight line representing the wall will also be eliminated. The data after hough transform is shown in the Figure 5.

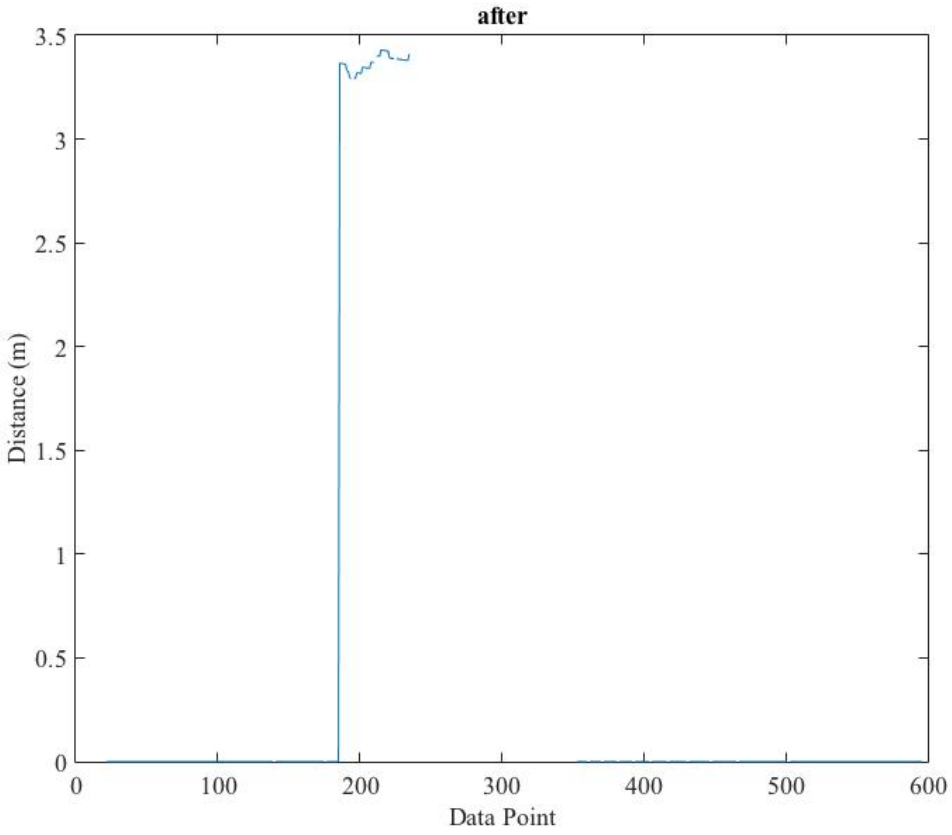


Figure 5

The only large group of data points cluster around the 200th data point and 3.5 meter, representing the person in the hallway.

The next step is a simple iteration of the processed data. The average distance and center of the large group of data points from the person can be calculated. The position of the person in the cartesian coordinate can be easily obtained.

After extracting position of the person for every scan, we implemented linear regression to calculate the movement. By using linear regression over fifty scans separately for x position and y position over time, we can find the intercept and the slope of the line of the object's movement. Then we can further estimate the speed of the object in x and y axis and the direction of the object. And at last we can estimate the destination of the object based on the data acquired.

We calculated the anticipated destination of the object by extending the existing trail of the object with the calculated speed and direction. Then the robot will determine if the destination is on the left side or right side of the center line and it'll make a move accordingly. We implemented simple moving strategies for the turtlebot to avoid the obstacle. After detecting the object in the hallway and after calculating the anticipated destination of the object, the turtlebot will turn accordingly, until the designated point from the scan data returns a certain distance. Then the turtlebot will follow the contour of the wall by keeping this point in the scan data a certain distance.

The working range of the program is shown in the figure below:

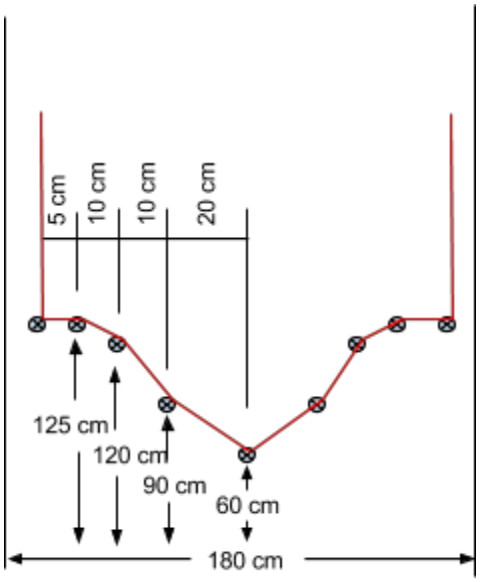


Figure 6

The test subject should walk within the area the red lines enclosed. This program does not work properly if the test subject is too close to the sensor or too close to the walls.

To run the program, follow the steps below:

1. put the files and folders in the workspace
2. roscd to the catkin workspace catkin_ws
3. run "catkin_make" in terminal to compile the code
4. run "roslaunch turtlebot_bringup minimal.launch" to launch the turtlebot
5. in a new window, run "roslaunch turtlebot_bringup 3dsensor.launch" to launch the kinect
6. in a new window, run "rosrun safe_turtle safe_turtlebot" to run the safe turtle program in the background
7. in a new window, run "rosrun crossing obstacle1" to run the obstacle avoidance program

The source code of the program can be found at:
**https://github.gatech.edu/wliu88/turtlebot.git**
The demo video of the program can be found at:
**https://www.youtube.com/watch?v=VqYyENpbYWw**