Opening Door Detection By Corobot
Student: Hongyu Zhao
Advisor: Zack Butler

Introduction

The aim of my project is to figure out if a door is open enough for the Corobot to pass. To answer this question, I developed an algorithm to find the door from an image and find the width of the open part. [1]

The Corobot is a capable, expandable and affordable mobile robot platform designed to minimize the complexity of robot development. Since my function needs to run on the Corobot, I also needed a communication function between my function and the ROS system, which is the base system of Corobot.

Background

Canny edge detection:
By the Canny edge detection function, I am able to smooth the image and have a auto-threshold for the different light-level image. This is because the Canny edge detection applies a Gaussian filter and double threshold. By doing this, it modifies the image and make it ready for the Hough transform function.

Problem, Research and Method

Problem:
1. Closed Shape:
   As shown below, the shape of door is not closed. It’s because the edges of the image comes from an edge detection function, which can miss some parts. In this case, the edge is hard to consist a closed body, which means the shape segmentation function won’t works well.

2. Object Location:
   Since there are some notices on the edge images, the image will include a lot of rectangles. My algorithm must find the correct door to examine. To do this, I need a matching function to catch some features from the door. In this case, the highest matching area will be the location of the door on the image. However, after trying SURF and SIFT to match the handle of the door, the results weren’t great.

3. Light:
   In the real world, the light will be a main element which affects the result of edge abstraction. For my project, when the light is too strong, the reflection area will be recognized by HT function.

Method:

Recognition function:
1. Canny edge detection
2. Hough transformation
3. Blind line extension
   3.1 loop the line and find the light pixel.
   3.2 count the pixel numbers and mark the line
   3.3 remove the noise line and build the light area
4. Recognition

Communication function:
1. image from camera
2. call back thread with the input image and recognition function
3. result summary and return the final result

Experiment and Result

Setting up the Corobot in front of the door and test the function basic on the different view point.

For each view point, I tested my function 100 times, and summarize the results as a present value.

Here is the summary of the result which comes from the experiments. I list them by view point and distance between door and Corobot. If the accuracy is more than 90%, then it’s good enough to say that my function works well for the current point. Otherwise, it may need upgrade or rebuild.

<table>
<thead>
<tr>
<th>D/V</th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>13%</td>
<td>91.11%</td>
<td>97.14%</td>
<td>98.57%</td>
<td>20%</td>
</tr>
<tr>
<td>D2</td>
<td>18.5%</td>
<td>87.09%</td>
<td>98.7%</td>
<td>91.80%</td>
<td>19.35%</td>
</tr>
<tr>
<td>D3</td>
<td>8.57%</td>
<td>47.14%</td>
<td>71.42%</td>
<td>60%</td>
<td>9.16%</td>
</tr>
</tbody>
</table>

Conclusion

As the table displayed, the closer to the door the Corobot is, the more accuracy my algorithm is. In addition, if the Corobot is facing the door at a good angle, the accuracy is increased. Considering the aim of my project and the functionality of Corobot, I think this result is good enough to say my algorithm is able to recognize the opening width for the Corobot to pass.

Reference