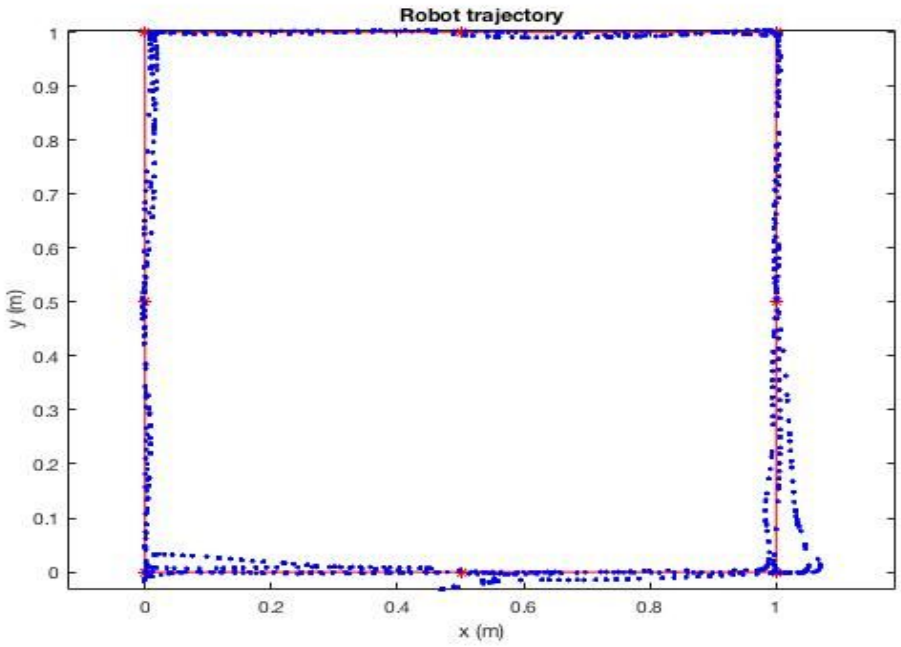
ME 144 Lab 4

Name : Jesus Gutierrez, Rebecca Silva

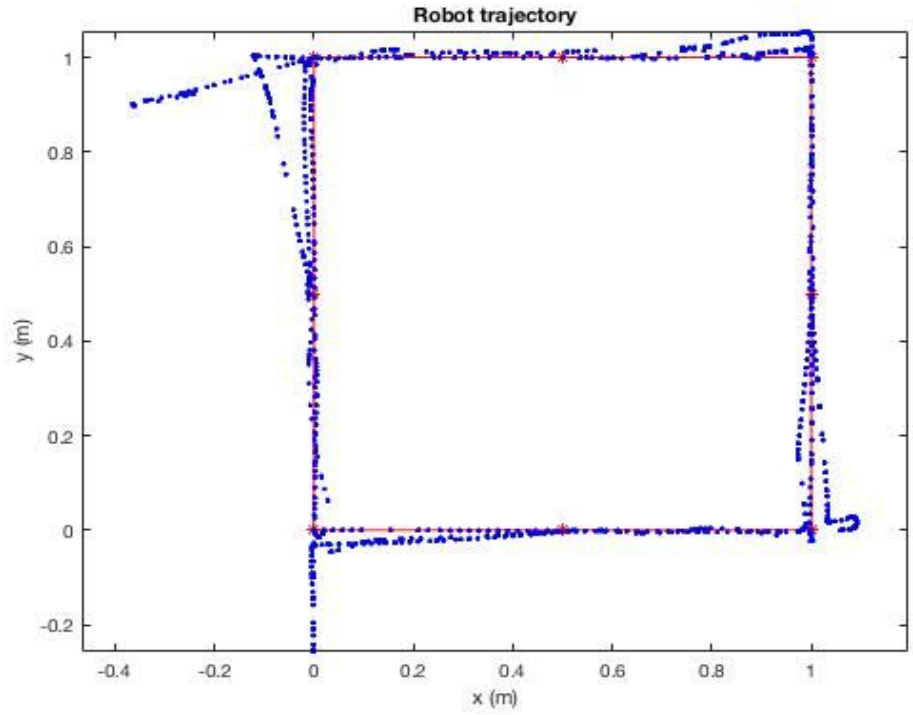
Date: 10th May 2017

Assignment 1:

In this assignment, we implemented waypoint navigation which caused a real world robot to move continuously in a square. The efficiency and accuracy of the program depends on on the K value; the control gain that is proportional to the rotational velocity and the threshold value Dthresh. We experimented with different values of k and Dthresh and found them to be 2.5 and 0.05 respectively, for the best performance. These values were identical to the ones we used for the simulation robot.

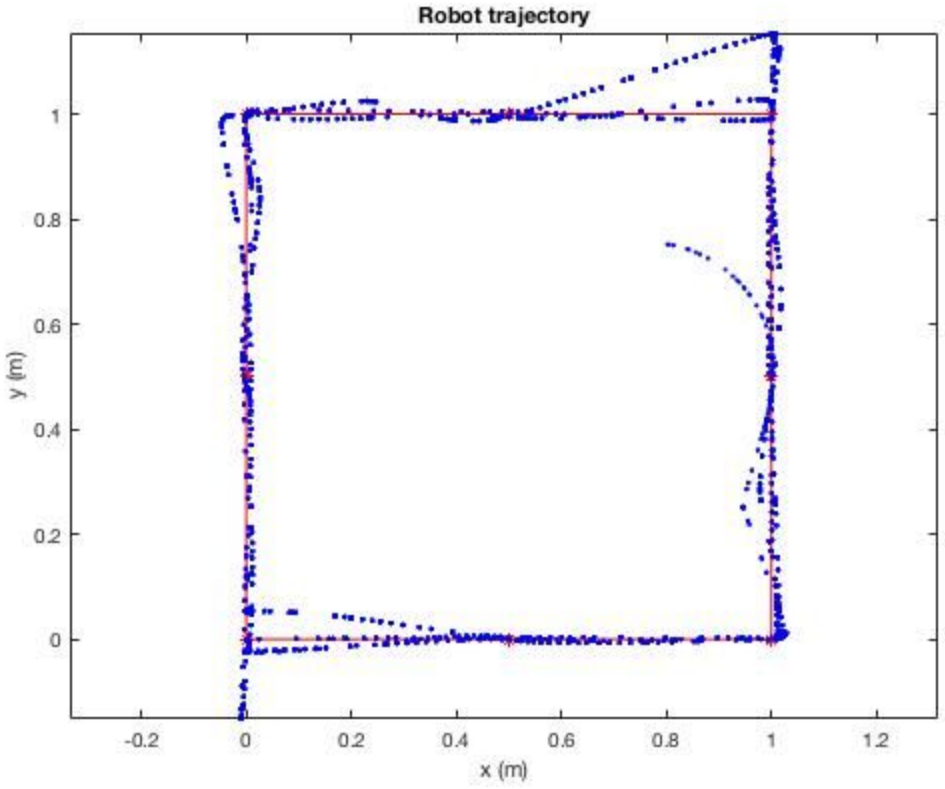


K = 2.5, Dthresh = 0.05



K = 2.75, Dthresh = 0.06

When we increased k value, the bot turned too quickly, and as a result, and overshot its target point. However, increasing threshold value but a tiny amount caused the bot to turn more efficiently.



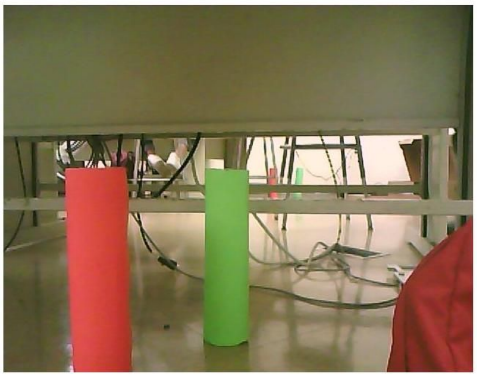
K = 2, Dthresh = 0.03

The bot got closer to each target point depending on the threshold value. If the threshold value is decreased, the K value should also be small in order for the program to work.

For optimal values of k and Dthresh, the matlab plot of the bots trajectory was quite similar to the trajectory of the bot in the real world. The motion of the bot however, was a lot more jerky in the real world than in its matlab trajectory. This is because the matlab plot only displays the robot's motion in a 2d plane, whereas the jerky motion was in a perpendicular plane. The bots motion is not smooth. It stop and accelerates very fast.

Assignment 2:

In this assignment, we implemented colour- based tracking to make the bot detect and move towards the cylinders. We found the upper and lower threshold values for the Hue in the HSV colour space to be 0.2335 and 0.2875 respectively, and created a binary image that would only detect pixels that belonged to the green cylinder. While trying to detect the orange cylinder, the hue value range of $0 < Hue < 0.075$ were not accurate enough and would yield a binary picture with too many false positives. In order to improve performance, we found the saturation value range to be $0.68 < Sat < 0.90$. We then combined both of these matrices to get the best binary image, with minimal noise.

<table>
<tr>
<td> 
Original Image</td>
<td> 
Hue Values</td>
</tr>
<tr>
<td> 
Saturation Values</td>
<td> 
Hue and Saturation Combined</td>
</tr>
</table>

We used a P-controller to make the robot rotate in place till it detected the green cylinder and then move towards it. If it got too close, ie, if its area on the image was too large, it would back up, so as not to hit the cylinder.