Anirudh Topiwala

topiwala.anirudh@gmail.com, +1-240-302-3398, LinkedIn, GitHub: /anirudhtopiwala/

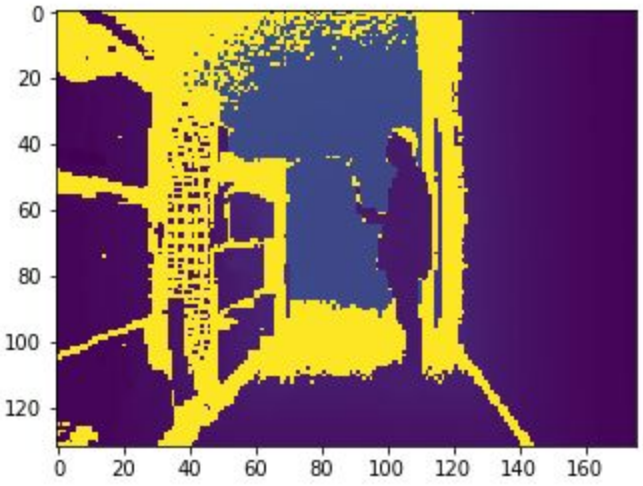
Braincorp Software Challenge - Robotic Perception: Machine Learning and SLAM - Software Engineer/Scientist position

Thank you for considering me for this position. I have tried my best to develop an algorithm to address the problem at hand. I will first explain what I did and then address on further improvements that can be used to make the algorithm more robust.

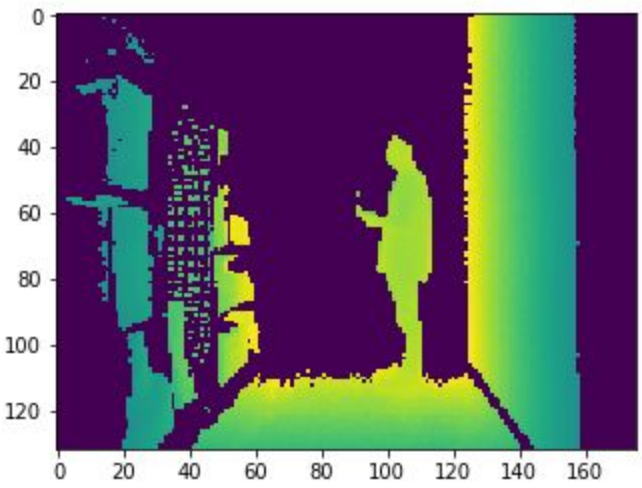
Methodology:

1) Extract the image from text file using np.loadtext().

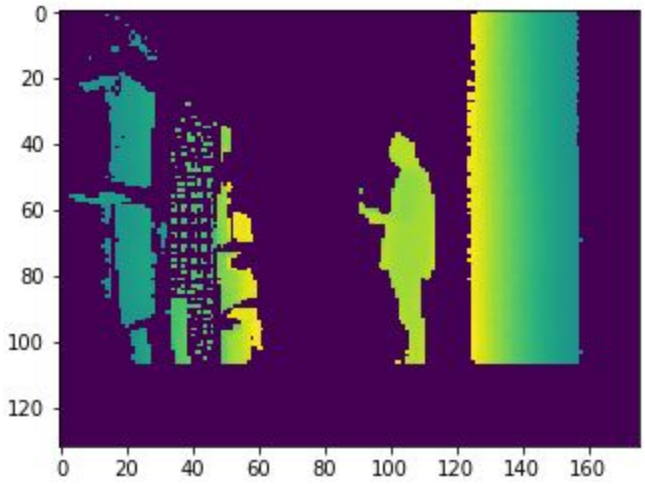
Output for ./human_corridor_0.txt



2) Threshold the image by removing distance values greater than 4m and less than 2m as it is given that the human is always approximately 2m away from the robot.



3) Remove the bottom 1/4th of the distance values. This is a reasonable assumption as human will always be above this region given that it is approximately 2m away from the robot.



4) Now as we can see in the above image, human is always in the middle region (column 45 - 122) as the other parts are covered by the wall and the shelves. Making this assumption, I extract the contour with the largest area and I get the centroid of the human.

5) Once I get the centroid, I can get the extreme left and the extreme right value on the row of the centroid to get the extreme pixel values of the human contour. This way I will be able to get the exact clearance.

6) We know that the horizontal field of view is 70` and the width of the image is 176, therefore angle per pixel is (70/176)`. [1]

7) Using the extreme pixel value of the human, I can get the angle it is making with the center of the image (where the robot is assumed to be) and using the distance value and using the sin() formula, we get the horizontal distance of human from center of the corridor.

8) This distance is then added to half the width of the corridor (given as 1.5m). Therefore we get the clearance value as = 0.75 + distance.

8) The left and right avoidance command can be calculated by checking if the extreme human pixel is on which side of the midpoint of 176, that is 88th pixel.

9) This way the outputs generated are as follows:

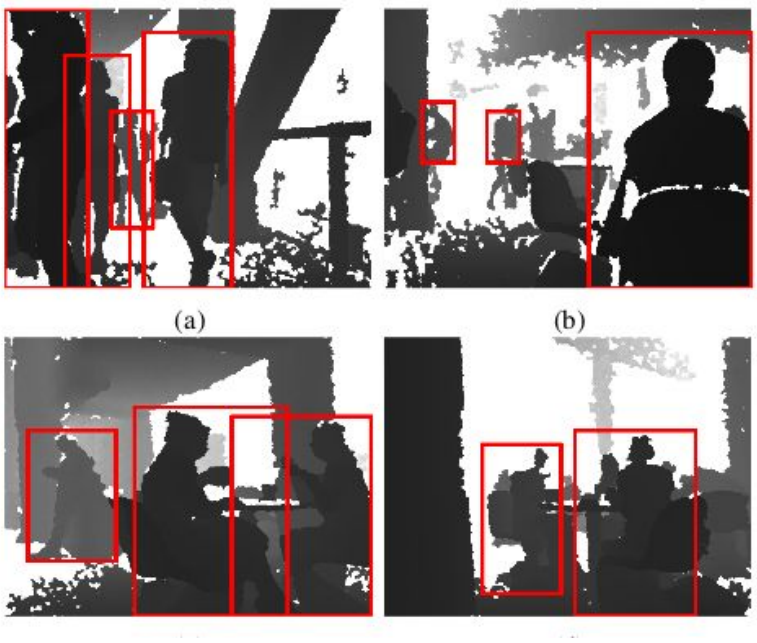
./human_corridor_0.txt = left 1.134049470219828
./human_corridor_1.txt = left 0.7994229625803765
./human_corridor_2.txt = right 1.0455388513621178

Drawback of this Algorithm

1) It assumes that the robot is approximately in the center of the corridor.
2) It uses contours to identify the human pixel, which is not always accurate, especially in cases of multiple humans or dynamic obstacles.

Improvements or Future Work

1) Using Histogram of Oriented Depth [2] to train a SVM to locate human will be much more accurate as it will not rely on detecting contours. It will also be able to track multiple humans together.
2) The paper [2] also talks about how the trained network will also avoid occlusions as it is trained to just detect the humans.
3) With the trained network, the person can be standing or sitting as well and therefore can be used as an holistic solution to solve the task at hand.
4) Below is an image taken from the paper [2]:



5) Combining this with semantic segmentation using UNET or MASK RCNN can help make this algorithm more robust.

References:

[1] "Mapping units of depth image and meters," *Orbbec 3D Club*. [Online]. Available: https://3dclub.orbbec3d.com/t/mapping-units-of-depth-image-and-meters/1600/9. [Accessed: 13-Feb-2019].

[2] B. Choi, Ç. Meriçli, J. Biswas, and M. Veloso, "Fast human detection for indoor mobile robots using depth images," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 1108–1113.