Task 1 Documentation

Ajsal Ali

***Abstract*— The first task is composed of 2 parts. The first part dealt with edge detection while the second part was to create a depth map using python tools like OpenCV and numpy**

In this project, we utilized computer vision techniques, specifically edge detection and depth mapping, to enhance safety measures and navigation aid, particularly in robotics. For edge detection, the Laplacian kernel was employed to determine the edges in the image.Ransac and Hough Transform algorithms were applied to find the line passing through these edges.In the second part depth mapping technique was used to plot a color-coded map implicating object proximity so that to avoid obstacle collision

I. INTRODUCTION

The problem statement involved ensuring safety and navigation for a bot.In first part initially I tried using Sobels kernel to find the edges in the image. Efforts were made to fine-tune the edge detector by varying the thresholds to find the correct edges in the image , However, it yielded incorrect results. Laplacian edge detection was employed followed by Ransac and Hough Line algorithm to draw the edge

II. PROBLEM STATEMENT

The first task comprised of 2 problems.

Problem 1: Edge detection and marking of edge

Fig. 1. Example of a table edge detection

The first problem was to detect edges and mark the edges of a table to prevent the robot from falling off the table. The edge-detection is to be done with any available edge-detection techniques coded from scratch. Then binarize the edge detected and apply Hough transform or any other line drawing algorithm to draw the edge of the table in the table.png image.

Problem 2:Plot depth map using the concept of stereo Vision

Fig. 2. Left image

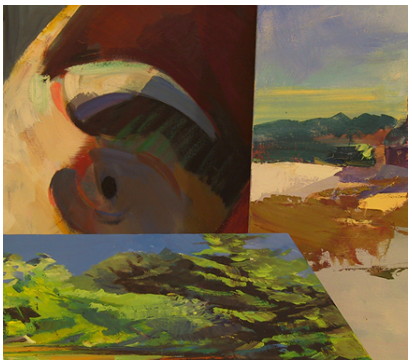

Fig. 3. Right image

The objective of the second problem was to write a python or c++ code to plot a depth map that accurately represents the proximity of objects in the bot environment using the concept of stereo vision. This can be done by measuring the shift of block of pixels in the right and left image(Fig.2 and Fig.3) captured by 2 identical cameras placed in the left and right sides of the bot.

III. INITIAL ATTEMPTS

In the case of the first problem, I used Sobel edge detection,but it yielded an inaccurate result. It was impossible to identify the table edge from the edges found(Fig.4)

IV. FINAL APPROACH

Problem 1

*Write anyone who might have helped you accomplish this eg any senior or someone

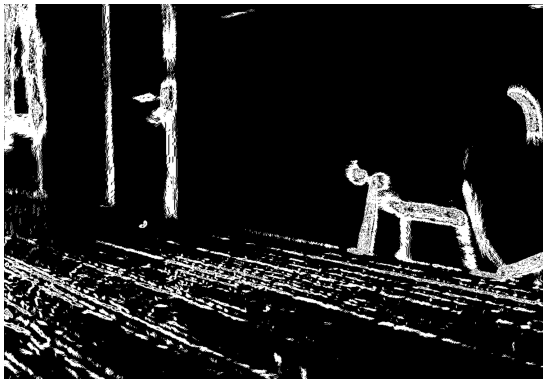

Fig. 4. Edge detected by sobel edge detection

0	−1	0
−1	4	−1
0	−1	0

Fig. 5. Laplacian kernel

The Laplacian edge detector coded from scratch was used to determine the edges in the image

The above kernel (Fig. 5) with applying a threshold of 6 gave a good result(fig.6)

Fig. 6. Final edge

Thereafter Ransac algorithm coded from scratch was employed to find the edge of the table.

Problem 2

a function **stereo match()** was defined to return the disparity map by taking left and right images, block size, and maximum offset as input. The function first takes a block of pixels with the block size on the left image and matches the position of the block on the right image. **The sum of Squared Differences** was used to reduce the error in matching the block. The **disparity** was calculated for all blocks in the images and finally, the function returns the disparity map of the image. Finally, the disparity map is normalized by dividing all element of the map by the maximum in it and plotted depth map by using a pre-defined color map 'jet' from matplotlib(Fig.8) .I also plotted a depth map based on color gradient in red and blue(Fig.9)

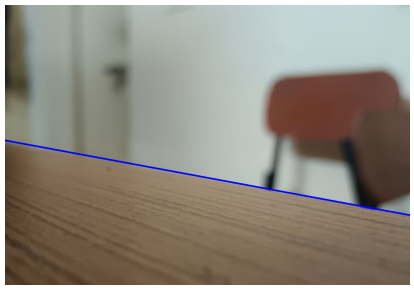

Fig. 7. Final image

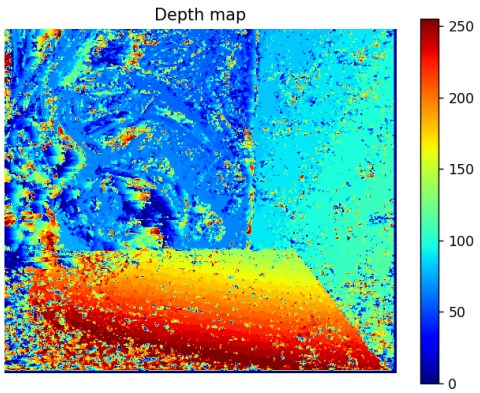

Fig. 8. Depth map

REFERENCES

[1] https://medium.com/analytics-vidhya/distance-estimation-cf2f2fd709d8.

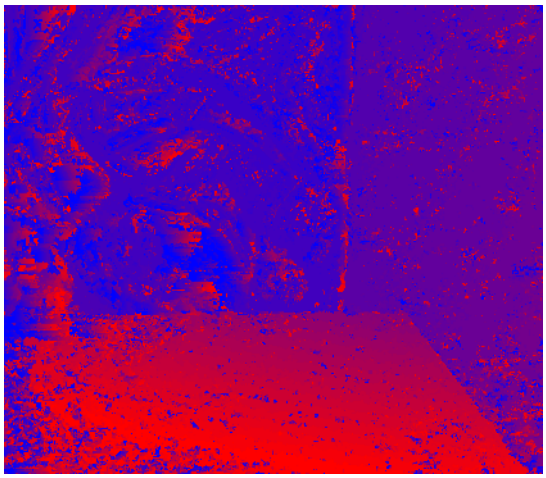

Fig. 9. Depth map