Aerial Robotics Kharagpur Documentation Task 1

Abstract:
In this task, I employed the technique of Hough Transform and Canny edge detection method followed by Stereo Block Matching (BM) algorithm in b part of the task.

Different fields of operation of these algorithms in ARK as:
1)Detection of obstacles
2)Decision of the aerial vehicle should go further or not(Depth Map).

I. INTRODUCTION

In this task, I employed the Hough Transform and Canny edge detection method followed by the Stereo Block Matching (BM) algorithm in the b part of the task.

II. PROBLEM STATEMENT

The assignment involved detecting table edges and creating a depth map from two robot camera images. I began by implementing Canny edge detection. First, I read and converted the table.png image to grayscale for better Canny performance. To detect edges, I applied filters that highlight pixel intensity changes in horizontal (x) and vertical (y) directions. By adjusting two trackbars, I set the Canny detector function's threshold values to (88,125). I then used the Hough transform with the parametric equation $\rho = x\cos(\Phi) + y\sin(\Phi)$ to draw red lines extending up to 1000 pixels.

For task B:

- The process initiates by outlining the paths to the left and right stereo images.
- Next, it invokes generate_depth_map with the image paths to construct the depth map.
- Subsequently, generate_heatmap is called to create a heatmap based on the depth map.
- Lastly, visualize_heatmap is utilized to present the heatmap visually.

INITIAL ATTEMPTS

I was doing tutorials in openCV and fundamental Python and was working task-oriented since I had very little time.
Didn't get a lot of errors during this time.

V. FINAL APPROACH

My final approach was as mentioned above
I would elaborate more on them.

Import Libraries: NumPy, OpenCV (cv2), and pyplot from Matplotlib are among the libraries that the code imports.

Read Image and Canny Edge Detection: It uses OpenCV's imread function to read an image called "table.png," storing the result in the variable img. Subsequently, the image undergoes Canny edge detection with the cv.Canny function, which identifies edges within the image. The outcome is kept in the canny variable.
Plotting Pictures: Using a for loop and the plt.subplot and plt.imshow functions, it generates a Matplotlib figure and plots the original image and the Canny edge-detected image side by side.

Hough Line Transform: Using the CV, it applies the Hough Line transform to the image that has the Canny edge detected.HoughLines are operational. The parameters that CV received. The Canny edge-detected picture (canny) is represented by HoughLines.

My seniors in ARK Kumar Anant helped me a lot in this.

Task B part.

cv2.imread(): Reads an image file.

left_image_path and right_image_path: Paths to the left and right stereo images.

cv2.IMREAD_GRAYSCALE: Loads images in grayscale.

cv2.StereoBM_create(): Creates a StereoBM object for stereo matching.

numDisparities: Sets the number of disparities to compute.

blockSize: Determines the size of the window for the blocksize

stereo.compute(): Computes the disparity map.

left_img and right_img: Grayscale images for stereo matching.

disparity.min() and disparity.max(): Identifies the minimum and maximum values in the disparity map.

np.uint8(): Converts the disparity map to an 8-bit unsigned integer.

Normalizes the disparity map to a range of 0 to 255 for better visualization.

cv2.bitwise_not(): Inverts the disparity map to generate a depth map.

return: Outputs the depth map obtained from the stereo images.

VII. FUTUREWORK

LThis was my first time doing something related to computer vision I think I cannot answer this without more exposure. it was a good learning experience.