ENPM 673 - Project 3 Report
Date: 18/04/22

-Jeffin Johny Kachappilly
UID : 118293929

Output link : https://drive.google.com/drive/folders/1xCjD5UrdSqfUJbjEjwAt1OQplVN4-4qa?usp=sharing

Step 1 : Calibration

Feature detection:

- Orb_detect was used to determine the features and to get good feature points, Lowe's ratio test was deployed. This prevented the need to use Ransac as only inliers were considered by choosing approximate ratio.
- Then the keypoints were converted to pixel coordinates and stored as a list.

Fundamental Matrix:

- The first 8 points from the list are passed to fundamental matrix function and is determined using SVD.
- The rank was checked and also the condtion: xt*F*x=0, was checked to verify whether the determined one is correct or not

Essential Matrix:

- The pre-multiplication with K transpose and post multiplication with K to the fundamental matrix, generated the Essential matrix.
- After applying SVD to E matrix, the different set of possible camera configurations were determined.
- **Pseudo code for correct camera pose given in (curule.py) inside def correct_pose().** As the function showed error while outputting.

CHALLENGES : (Resolved)

- The fundamental matrix outputed by inbuilt function, different than the one generated by my function. Issue solved by:
  - Checked whether the self-made function satisfied, the condition mentioned above.
- For (octogon), the rank of fundamental matrix was 3 for, various keypoints. Solved by:
  - Enforcing rank = 2 ,by making the S from SVD of F to zero and multiplying all the matrices to form new F matrix.

Step 2 : Rectification

- The inbuilt function cv2. computeCorrespondEpilines was used to determine epilines. Found epilines corresponding to points in left image (first image) and drew its lines on right image and viceversa.
- For drawing lines a function was built which incorporated attributes from class ‘zip’ and ‘map’.
- The homography matrices for both images were determined using inbuilt function and wraprespective was done to make the epipolar lines horizontal.
- **H matrices printed in terminal, while running code**

CHALLENGES : (Not resolved)

- The epipolar lines for ocotogon wasn’t coming horizontal. Reasons:
    - This maybe due to the noise in the image or slight change in the value of fundamental matrix obtained.

Step 3 : Correspondance

Disparity :

- The sliding window technique was developed to find feature correspondence.
- A window was created for left image, which ran along the entire image. For the right image, a search region was employed and a window of same dimension as that of the left was traversed.
- Then the right window with least SSD was selected and disparity found(by considering the difference between pixel locations(first pixel coordinate ofa window) of width from each image.
- The window size and stride, were tuned for all the cases , to make output smooth as possible.
- Then the keypoints were converted to pixel coordinates and stored as a list.
- The disparity values which ranged from negative to positive were all made positive and scaled from 0-255.
- Then the grayscale image was developed and heatmap obtained by passing grayscale image to inbuilt OpenCV function.

CHALLENGES : (Resolved)

- The disparity map was showing inverted colors(such as nearby region black and faraway region white) Issue solved by:
    - Swaping the windows between left and right images.
- Initial map determined features even from end of image, when search region was along entire width . Issue solved by:
    - Limiting search region
- The map was almost white . Issue solved by:

- By initialising the blank image as np.float and while displaying converting back to np.unint

Step 4 :

Depth :

- Used the simple depth conversion formula (baseline*focal)/Disparity
- And outputted gray and heatmap...
- TO avoid infinity, when disparity becomes zero, 1 was added.