Computer Vision (CSL7360)

Programming Assignment-1 (100 marks)

Instructions:

1. Do not copy from other students. Any case of plagiarism will result in zero marks.
2. This is an individual assignment.
3. You can refer to codes online (e.g., Github, Kaggle), but do not copy-paste. The resource must be cited in the report if referred.
4. Strictly follow the submission guidelines.
5. Allowed languages: python

Submission Guidelines :

- Submit .py python files and .ipynb (with proper output blocks) for all the questions.
- Strictly submit a single report (.pdf) for all the questions. No .doc, .docx file will be accepted. Your report should include all the analysis and detailed observations.
- If you are using Colab, then attach your Colab link in the report (preferred). Make sure the submitted files do not have restricted access to it.
- Submit a single zip file containing all Python files and reports.
- The name of the zip file should be roll_no.zip, and Python files should have the name, e.g roll_no_qu1.py or roll_no_qu2.py, etc. The report should have the name roll_no.pdf. Include the .ipynb file as well.

If the naming convention is not followed, we will award zero marks.

---

1. **Harris Corner detection:** Your task is to implement the Harris Corner detection technique. You have to do the following: [50 Marks]
   - Implement the algorithm **from scratch**. You should not use any in-built functions.
   - Adjust parameters such as the window size and threshold value to optimize corner detection performance.
   - Make the code modularise to take images from the folder as an input.
   - Compare the performance of your algorithm with the corner detection libraries from OpenCV.

- The input images are present in the following location: https://drive.google.com/drive/folders/1la4hwF_n4g7T25d2gyCF1ob3HJOir3Th?usp=sharing
- Compare the visual results for both approaches, with and without the library.

2. Consider two stereo images I1 ("bikeL.png") and I2 ("bikeR.png") of a static scene captured from a stereo camera with the given intrinsic matrices of both the cameras in the file ("bike.txt"). Implement the stereo 3D reconstruction algorithm to find the disparity map, depth map (depth map at each pixel), and 3D point cloud representation of the underlying scene. [30 Marks]

3. Consider two images I1 ("000000.png") and I2 ("000023.png") of a static scene captured from a single camera with the given Fundamental matrix F ("FM.txt"). Write down a code to find the epipolar lines and draw the epipolar lines in both the images. Now, consider uniformly spaced 10 pixels on the epipolar line in the first image and find the corresponding pixels on the second epipolar line. Now, consider uniformly spaced 10 pixels on the epipolar line in the second image and find the corresponding pixels on the epipolar line in the first image. You should search only on the epipolar lines. [20 Marks]

ALL THE IMAGES ARE PRESENT IN THE BELOW LOCATION:
https://drive.google.com/drive/folders/1la4hwF_n4g7T25d2gyCF1ob3HJOir3Th?usp=sharing

ADDITIONAL INSTRUCTIONS:

1. **Make sure that you submit a running code. We will schedule a viva session later for the code demonstration.**
2. If you are submitting a collab file, make sure to provide relevant access before sharing it.
3. STRICTLY FOLLOW THE SUBMISSION GUIDELINES.
4. **In Q1, when you are implementing from scratch, you can only use the opencv libraries for basic operations such as reading, saving the images, or performing operations such as cv2.cvtColor.**
5. **You can use inbuilt functions to find features and feature descriptors in Q2 and Q3. However, you should not use inbuilt functions for finding correspondences and fundamental/essential matrix.**

DelayPenalty: Upto 1 day: 10%, upto 2 days 20%, upto 3 days 50%, beyond 3 days 100%