EE 645 {3D-Computer Vision}

Assignment 1: Panorama Stitching

GitHub Repo: (https://github.com/Harshp1802/3d-CV-Panaroma-Stitching)

- Name: **Harsh Patel**
- Roll No: **18110062**

Tasks:

Take atleast 4 color images per scene in the dataset and perform the following steps.

1. Detect, extract, and match features (inbuilt functions allowed).
2. Estimate homography matrix between two images using RANSAC (inbuilt functions are allowed except those functions which directly estimate the homograpy).
3. Stitch (atleast 4) color images per scene from the dataset using the homography matrix estimated in step (2) to create a panorama (inbuilt functions are allowed except for warping and blending).
4. Stitch the images used as input in step (3) using in-built command for homography estimation and compare it with the panorama obtained in step (3).

Approach Used

1. Detect and extract key features from a given image
    - **SIFT Algorithm**
2. Feature Matching [Simple Brute Force Method]

Example Feature Matching:

3. Using RANSAC (RANdom SAmple Consensus) Algorithm that best fits the Feature matchings of two images
4. Homography Matrix Calculation using the Best Point Correspondences
5. Warping & Blending
    - Pyramid Method for Blending

- Note: Each step is described with the code.

Stitching done for 4 images in all the Datasets

- Notes:
  - **I4** --> Left-most Image | I1 --> Right-most Image
  - The Second Image in all the datasets has been used as the Base Reference image for stitching, i.e, **I3**
  - **H_34** --> Homography matrix of I3 w.r.t I4 [Needs to be inversed before finding the Warp as I4 warp is to be found w.r.t I3
  - **H_23** --> Homography matrix of I2 w.r.t I3
  - **H_12** --> Homography matrix of I1 w.r.t I2
    - Here we calculate H_13 for stitching the right most image by using:
    - `H_13 = H_12.dot(H_23)`
  - Warping is done by finding the above matrices (**H**) and then transforming every image w.r.t I3
  - Inverse Warping method used to avoid creation of holes in the Output Image
  - For Blending, I have used the Laplacian Pyramids method discussed in class. I have decribed its implementation in the code itself.
  - Few references used are:
    - http://pages.cs.wisc.edu/~csverma/CS766_09/ImageMosaic/imagemosaic.html
    - Multi-Band Blending from **Brown, Matthew, and David G. Lowe. "Automatic panoramic image stitching using invariant features." International journal of computer vision 74.1 (2007): 59-73.**
    - [https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_feature_homography/py_feature_homography.html]
    - [https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_pyramids/py_pyramids.html]
- For Q4, the in-built Homography estimation is done using the Cv2 library.

How to run the Code:

Use the Jupyter Notebook for the code

- Required Inputs:
  - **Image Dataset Number** --> (**int**) from 1 to 6
  - **In_built** --> (**Bool**) True or False
    - True, if want to use the in-bulit Homography Estimation
- Tunable Parameters:
  - **No. of Iterations to run the RANSAC : *default --> 800***
    - [For Dataset-3, this value has to be increased to 4000. More details mentioned in its individual README file]
  - **L1 Threshold for no. of inliers calculation : *default --> 5***
  - **No. of Iterations to run the RANSAC : *default --> 4000***
- The code automatically saves the following files in the following directory: `.\results\I{Image Dataset Number}`
- Stitched Images (Without the use of in-built functions):

    - **Panaraoma_1_2_3_4.jpg** [Final Output: 4 images stitched together]
    - stitch_1_2.jpg [Stitching of 2 images]
    - stitch_1_2_3.jpg [Stitching of 2 images]

- Stitched Images (Using in-built Homography Estimation):

    - **Panaraoma_in_built_1_2_3_4.jpg** [Final Output: 4 images stitched together]
    - stitch_in_built_1_2.jpg [Stitching of 2 images]
    - stitch_in_built_1_2_3.jpg [Stitching of 2 images]

Results for each Dataset:

- Individual .md files [Images in these files are automatically updated on running the code]

```
    │   results_I1.md
    │   results_I2.md
    │   results_I3.md
    │   results_I4.md
    │   results_I5.md
    │   results_I6.md
```

- Individual pdf reports (".\results_report_pdf")

```
└───results_report_pdf
        results_I1.pdf
        results_I2.pdf
        results_I3.pdf
        results_I4.pdf
        results_I5.pdf
        results_I6.pdf
```

    - Please ignore the emoji in the pdf files. [Generated due to the conversion from .md to .pdf]