Introduction to Computer Vision

Assignment #3

Due: Dec.-8 (Thur.) (before 11:59pm)

Instruction

a. Submit your source codes in a single compressed file "CV_A3_*StudentID*.zip" to iCampus.

b. Python 3.7 or higher / OpenCV 3.4 or higher will be used to execute your submitted codes.

c. You can submit at most 3 files.

d. Any work that you turn in should be your own.

Part #1. Fundamental Matrix [100 pts]

The requirements of Part #1 will be evaluated by running '***A3_Fmat.py***' file.

1-1. Fundamental matrix computation

a) Load two images 'temple1.png' and 'temple2.png'. The feature correspondences between two images are also provided in 'temple_matches.txt' file. You can use `np.loadtxt()` function to load a text file:

```
M = np.loadtxt( 'temple_matches.txt' )
```

Each row of the text file gives one correspondence by 4 values (x_1, y_1, x_2, y_2) describing that a feature point (x_1, y_1) of the first image is matched to (x_2, y_2) of the second image. You can utilize those correspondences to compute the fundamental matrix.

b) Implement the Eight-point algorithm to compute the fundanmental matrix:

```
function F = compute_F_raw ( M )
```

Refer the lecture material (page #63 – #70 of 'CV_08_Two-View_Geometry.pdf').

c) Implement the Eight-point algorithm with a normalization:

```
function F = compute_F_norm ( M )
```

One simple normalization scheme indepedent from the feature locations is recommended:

- Translation to move the image center to origin $(0,0)$
- Scaling to fit the image into an unit square $[\,(-1,-1),(+1,+1)\,]$

You need to remember the normalization and un-normalization should be reflected to your fundanmental matrix.

d) Implement your own algorithm to compute the fundanmental matrix:

```
function F = compute_F_mine ( ... )
```

- It should return the result within 3 seconds.

e) Print the average reprojection errors of three functions implemented in (b), (c), and (d) to the console. In order to compute the error, you are required to use the given function in 'compute_avg_reproj_error.py' file. Refer the following example:

```
Average Reprojection Errors (temple1.png and temple2.png)
   Raw = 4.658691021702988
   Norm = 3.4652248099806697
   Mine = 1.1315629509043314
```

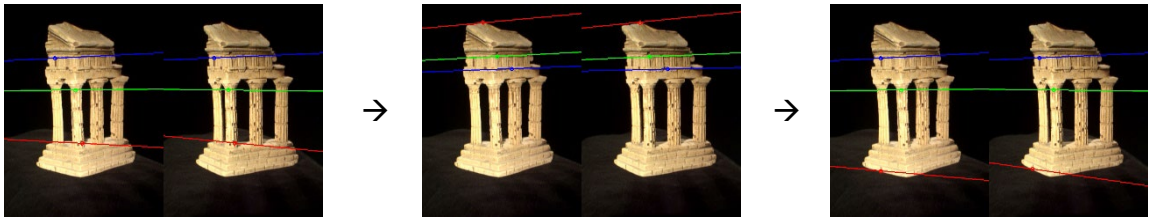
f) Your script should produce the results of (e) for the following three pairs of images:
('temple1.png' / 'temple2.png') , ('house1.jpg' / 'house2.jpg'), and ('library1.jpg' / 'library2.jpg')

g) Tip: You can compare the OpenCV built-in function '`cv2.findFundamentalMat`' to validate your solution. However, do not spend too much time to outperform it.

1-2. Visualization of epipolar lines

a) Implement a script that performs the followings:

- Randomly select 3 correspondances: $(p_1 \leftrightarrow q_1)$, $(p_2 \leftrightarrow q_2)$, and $(p_3 \leftrightarrow q_3)$
- Compute 6 epipolar lines $l_1, l_2, l_3, m_1, m_2, m_3$ corresponding to $p_1, p_2, p_3, q_1, q_2, q_3$.
- Visualize p_1, q_1, l_1, m_1 as red, p_2, q_2, l_2, m_2 as green, and p_3, q_3, l_3, m_3 as blue.
- If any key except 'q' is pressed, then you should repeat the above process.
- If 'q' is pressed, then you need to close the window and finish the visualization.



b) Your script should run (a) for the following three pairs of images:
('temple1.png' / 'temple2.png') , ('house1.jpg' / 'house2.jpg'), and ('library1.jpg' / 'library2.jpg')

c) Note that, you need to use the fundamental matrix computed by the function implemented in 1-1-d).