Introduction to Computer Vision

Assignment #3

Due: Dec.-20 (Fri.) (before 11:59pm)

Instruction

a. Submit your source codes, report, and descriptor 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 5 files.

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

Part #1. Fundamental Matrix [35 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 ( ... )
```

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
```

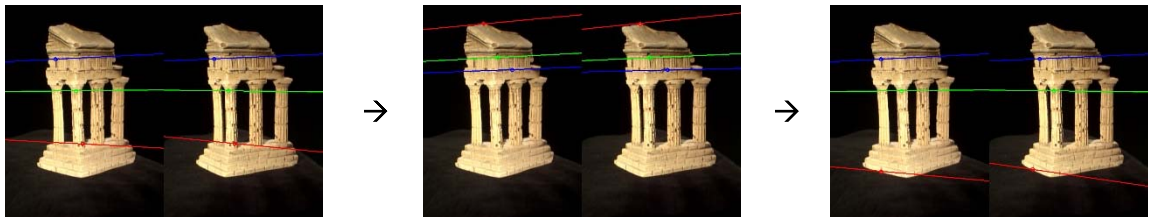
f) Your script should produce the results of (e) for the following three pairs of images: ('temple1.png' / 'temple2.png') , ('house1.jpg' / 'house2.jpg'), and ('library1.jpg' / 'library2.jpg')

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

Part #2. Image Retrieval [65 pts]

The requirements of Part #2 will be evaluated based on your descriptor file, '***A3_StudentID.des'***

2-1. Dataset: a subset of UKBench

a) 1000 images and their pre-extracted SIFT features are given.

b) Each four consecutive images (e.g. ukbench00000--00003.jpg, ukbench00004--00007.jpg, ...) are considered as relevant items, and thus they are ground truth retrieval results.

c) Each SIFT feature corresponding to an image (sift100000--100999) has the following binary file format:

$$\begin{bmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,128} \\ v_{2,1} & v_{2,2} & & v_{2,128} \\ \vdots & & \ddots & \vdots \\ v_{n,1} & v_{n,2} & \cdots & v_{n,128} \end{bmatrix}$$

The image has n SIFT features and each SIFT descriptor is 128-dimensional vector. Note that, each value $v_{i,j}$ is stored as 'unsigned char' format (1 byte representing a range from 0 to 255).

2-2. Task: computing image descriptors for the retrieval task

a) Your task is to compute image descriptors that reflect similarities among images by L_1 or L_2 distances. You should store the descriptors as the following binary format file:

$$\boxed{\begin{array}{l} N\ D \\ d_{1,1}\ d_{1,2}\ \dots d_{1,D} \\ d_{2,1}\ d_{2,2}\ \dots d_{2,D} \\ \cdots \\ d_{N,1}\ d_{N,2}\ \dots d_{N,D} \end{array}}$$

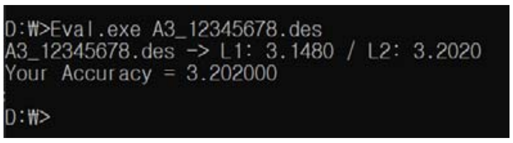
, where N and D are the number of images (=1000) and the dimensionality of each descriptor, respectively. And $d_{i,j}$ is j^{th} element of i^{th} descriptor. The required data types of variables are summarized as follows:

Variable	Type
N	Signed 32 bits integer (***int*** in C/C++)
D	Signed 32 bits integer (***int*** in C/C++)
$d_{i,j}$	32 bits floating point (***float*** in C/C++)

Note that, the <u>dimensionality of your descriptor cannot be larger than 1,024</u> (i.e. $D \leq 1024$). In other words, the size of your descriptor file '***A3_StudentID.des'*** can be up to 4,096,008 bytes ($= 2 \times 4 + 1000 \times 1024$ bytes).

b) You need to submit a code (either of python or C/C++) to produce the descriptors. Name your code as '***A3_compute_descriptors.py***' or '***A3_compute_descriptors.cpp***'. Your script (in case of python) or executable (in case of C/C++) should run when they are located at some directory containing 'image' and 'sift' directories as sub-directories. In other words, you should specify the data path as a relative directory path in your implementation.

c) Once you produced the descriptor file, you can evaluate your descriptors by using provided executable 'Eval.exe'. The program will evaluate your descriptor according to both L_1 and L_2 distances and take the higher one. It only works in Windows OS. You have to specify your descriptor file name as the following example:



```
D:₩>Eval.exe A3_12345678.des
A3_12345678.des -> L1: 3.1480 / L2: 3.2020
Your Accuracy = 3.202000

D:₩>
```

Note that, the accuracy is ranging from 0 to 4.

d) You are required to submit a report '***A3_report.pdf***' that describes your approach to compute the image descriptors. It should be longer than 1 page.

e) It is strongly prohibited to produce image descriptors by abusing the characteristics of the dataset. One typical example is setting $d_1 = d_2 = d_3 = d_4 = [1,0,0,\dots,0]$, $d_5 = d_6 = d_7 = d_8 = [0,1,0,\dots,0]$.