CAP 4410: Computer Vision Programming Assignments 2: Estimating Transformations

Due Date: Friday, October 8, 11:59 pm

Submission Type: Upload onto Canvas

Problem Statement: In this assignment, you will write Python 3 code to estimate the transformation between two images, I_1 and I_2. The steps involved are:

1. Detect image point features (F_1, F_2) in each of the images:
   - F_1 = detect_features (I_1, parameters)
   - F_2 = detect_features (I_2, parameters)

2. Match features detected in the images to create a list of possible matches (M) of the image features
   - M = match_features (F_1, F_2)

3. Write a function to compute the least square estimate of the transformation parameters (P) given a small set of matching points (m)
   - P = least_square_estimate (m, type_of_transformation)
   - The type of transformations could be (i) affine, (ii) 2D rotation and translation, and (iii) 2D perspective (or homography)

4. Estimate transformation parameters given the matches using RANSAC (Random Sample Consensus). The algorithm works by returning the best estimate among many random samples of matches (m) - the *minimal* subset of correspondences needed - and selecting the best parameter estimate based on the error on the rest. See Section 8.1.4 of your textbook or the following for more information https://en.wikipedia.org/wiki/Random_sample_consensus

To test your code, you can generate some transformed images using your code from Programming Assignment (Project) #1 and see if you can estimate the parameters back.

We will provide you a set of images to run your code and to report your estimates in your report.

Submission Requirements:

Please upload a ZIP file containing the following files:

1. All code files are according to the REAME.md that comes with the skeleton code.
2. A 2-Page technical report containing the following sections:
   - Pseudo-code for the least-squares estimating algorithm, and
   - Pseudo-code for the RANSAC algorithm, along with a specification of any assumptions about input and output.
   - See https://link.springer.com/content/pdf/bbm%3A978-1-4471-5173-9%2F1.pdf for pseudo code conventions to use.

- Show examples of estimation of parameters for each of the three transformations for 2 images from the dataset provided.

3. Your code will be tested on different test inputs and graded based on the progress of your approach on these test inputs. A demo session could be required if there is problems with the code you submit.

Grading:

Each assignment will be graded out of 100:

1. Code & Correctness (out of 65): quality of coding, readability, understandability (comments, variable names, etc.), and will compare function outputs with different images and parameters and test against instructor's code outputs.

2. Submission Requirements Met (out of 10): Can you follow the submission guidelines? Did you turn in your deliverables as instructed with proper naming conventions and packaging requirements as the instructor asked? Did you use the function prototypes correctly so that it works the way it was supposed to work? Basically, can you follow directions. At a real job, you have strict requirements and must adhere to them. Failure to do so can be disastrous.

3. Report (out of 25): To get top grades on the reports, you must have professional IEEE or CVPR formatted reports which are dual column, include good example images, mathematical formulas, etc. You can write them in any word processor but for the best results, check out Overleaf.com which uses LaTeX (lay-tech) but you do not need to use it. Your report must be in the zip submission as a PDF, report.pdf. Make sure to spend time on this, it is worth half your grade and should be taken seriously.

Solutions to your **programming assignments** must be self-sufficient and **not dependent on other computer vision code** other than image read and write, the SIFT feature detection, and feature matching functions from OpenCV package. You may use packages for display graphics or mathematics packages, such as for linear algebra (numpy or scikit).

All reuse of code must be clearly acknowledged in the source code, any README files, and also in the report. Failure to do so will be considered plagiarism.