CSE 291: Topics in Computer Science and Engineering
Computational Photography, Spring 2020 – Assignment 4

Instructor: Ben Ochoa

Due: Wednesday, April 29, 2020, 11:59 PM

Instructions

- Review the academic integrity and collaboration policies on the course website.
- This assignment must be completed individually.
- This assignment may be completed in the programming language of your choice.
- You may use third party libraries/packages for basic linear algebra, basic image processing, and image file I/O. But, you may not use third party libraries/packages that directly solve the problem. If you are uncertain about using a specific library/package, then please ask the instructional staff whether or not it is allowable.
- You must prepare a report as a pdf file. The report must describe the problems, and your solutions and results. Math must be done in Markdown/LaTeX.
- Additionally, you must create a zip file containing all of your source code, along with an automated build method (e.g., a makefile) and a `readme` file with clear and concise directions on how to build and execute your program.
- The zip file must also contain any output image files.
- You must submit both files (.pdf and .zip) on Gradescope. You must mark each problem on Gradescope in the pdf.
- It is highly recommended that you begin working on this assignment early.

Problems

For this assignment, you may use third party libraries/packages that solve the individual problems below, as long as the library/package does not jointly solve two or more successive problems. You must develop a solution that integrates all of the problems. All frames of the sequence are contained in `calit2bldg.zip`.

1. (15 points) Detect feature points in frame 1 and track (or match) them to feature points in frame 2, establishing a set of putative feature correspondences. Both the detection and tracking methods utilized must perform their respective operations to subpixel coordinates.
2. (5 points) Perform outlier rejection on the set of putative feature correspondences (using the 2D projective transformation model) to establish the set of inlier feature correspondences between frames 1 and 2.

3. (5 points) Estimate (using either a linear or (better) nonlinear method) the 2D projective transformation $\mathtt{H}_{2,1}$ from frame 2 to frame 1, given the set of inlier feature correspondences.

4. (5 points) Detect feature points in frame 2 that are not within a specified window size centered about each inlier feature point tracked (from frame 1) to frame 2. Join the newly detected feature points and tracked inlier feature points, yielding a combined set of feature points in frame 2.

5. (5 points) Beginning with frame $n = 3$ and ending on the last frame in the sequence, track (or match) the combined set of feature points in frame $n - 1$ to feature points in frame n. As above, perform outlier rejection, estimate the 2D projective transformation $\mathtt{H}_{n,n-1}$, detect "new" feature points in frame n, and join them with the tracked inlier feature points.

6. (5 points) For all n, compute the 2D projective transformation $\mathtt{H}_{n,1} = \mathtt{H}_{n-1,1}\mathtt{H}_{n,n-1}$ from frame n to frame 1 (note: $\mathtt{H}_{1,1} = \mathtt{I}$; frame 1 is the reference frame). Using the resulting set of 2D projective transformations, and the frame width and height, calculate the minimum and maximum x and y values bounding all $\mathtt{H}_{n,1}$ transformations.

7. (5 points) Construct a mosaic of the dimensions determined in the previous step by geometrically transforming frame n to the mosaic image under $\mathtt{H}_{n,\text{mosaic}} = \mathtt{H}_{1,\text{mosaic}}\mathtt{H}_{n,1}$, where $\mathtt{H}_{1,\text{mosaic}}$ translates frame 1 to the mosaic such that all geometrically transformed frames are tightly within the bounds of the mosaic. Write the resulting mosaic image to `calit2bldg-mosaic.png` and briefly discuss your results.

8. (0 points) Optionally, determine the reference frame resulting in the mosaic with the least number of pixels and construct this mosaic.