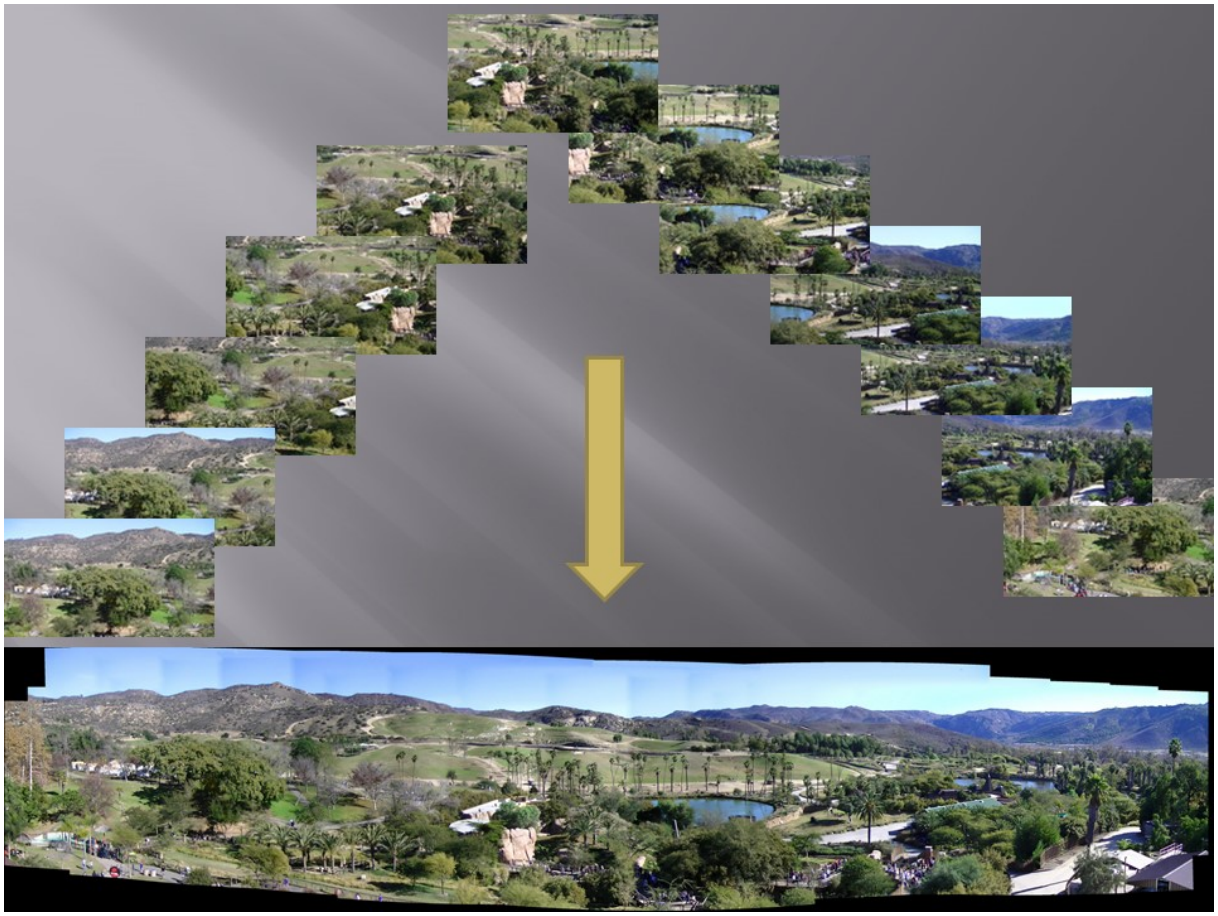
CS/ECE 181
winter 2023
Programming Assignment #1

Due: 11:59pm, Sunday, 1/29



You are to implement a panorama stitching program in this assignment. The input images will be taken by a single camera with no change of any intrinsic camera parameters. For simplicity, the camera trajectory is always a rotation about the person holding the camera, either in a simple panning motion (left to right or right to left) or a "saw-tooth" pattern (panning + up and down tilt motion). This implies that you should always match and stitch adjacent images in an alphabetic or numerical order.

Sample test images can be found in Gauchospace. There are two types of sequences: those depicting a complete, 360^o rotation and those depicting less than a 360^o rotation. Furthermore, you are welcome to use your own images for testing. You program (prog1.py) should take two arguments: the name of the directory that holds the discrete snapshots and the name of the output panorama image. Your program should then stitch all the snapshots in the directory together and output the panorama image, like this:

python prog1.py input_picture_directory output_panorama_image

Please note:

1. To stitch a pair of images, you need to have a sufficient number of reliably matched "anchors" to compute the homography matrix. You are allowed to use a built-in routine (such as SIFT and SURF) provided by OpenCV and Matlab. You can also use your own home-brew anchors (e.g., highly textured patches) and match them by

SSD (sum-of-squared-difference, the smaller the better). To ensure that your home-brew anchors can be robustly matched, please make sure that (1) your camera's pan and tilt motion between adjacent frames is small, (2) you hold the camera upright (no inplane rotation), (3) your scene objects are sufficiently far away and (4) you run a RANSAC routine for outlier filtering.

2. It is not necessary that your program works on 360^o sequences. You will receive full scores if you can stitch about 10 images, covering a horizontal field of view larger than, say 120^o. You can use your own sequences or (partial) sequences from the test sets.

What to code yourself and what not

The following applies to all programming assignments:

The rule of thumbs is that what is "essential" or "core" to a particular assignment should be your responsibilities. Auxiliary functions not central to an assignment can be from existing packages. For example:

- You need libraries to read and write image and video files and also to manipulate image pixels. You can use, say, OPENCV or other image libraries for these.
- You often need to manipulate matrices (e.g., for eigen or SVD computation), and you can use a linear algebra library (eg. Numpy.linalg) for those.
- You might need to perform certain optimization operations (e.g., to perform global optimization in 360^o stitching) that are best left to the experts in numerical analysis.
- You do not need to implement advanced, low-level features such as SIFT and SURF and Harris corner detector — if you choose to use them. But you can also use your home-brew features with SSD matching.

However, if you are to stitch image together or to compute optical flow fields and stereo disparity maps, which are what some programming assignments are about, then you cannot just call a library routine to perform those operations for you. Please also pay attention when programming requirements and hints are discussed in lectures and discussion sessions. If in doubt what to do yourself and what not, please ask on Piazza.

What to turn in electronically and what not

The following applies to all programming assignments. Please turn in electronically:

- Your codes (python, Matlab, C++, etc.), header files, personal libraries, a makefile (if you use C or C++), etc. Basically, files and libraries that are needed to compile and run your programs must be turned in. However, do not turn in large, public- domain libraries (e.g., OpenCV).
- Sample input data files. Do not turn in data files that are found on the web (Gauchospace, Kitti, Middleburry, etc.) Images and videos taken by you to illustrate new and interesting phenomena, algorithm designs, extra capabilities, etc. should be included.
- Statistical analysis results and plots, e.g., timing and accuracy plots, if required as specified in the assignment handouts.
- A short, written report (in PDF format and no more than one page. The page-limit is strictly enforced).

It is important that you run your programs at least once on an ECI or CSIL machine so you know what is available there. Only turn in codes, packages, libraries, etc that are not there.

The report should detail (1) any special way to compile and run your programs, (2) particular data sets that you want the the grader to use, and (3) required project information, such as performance and timing plots.

The program-specific URL to turn in your assignments will be announced later.