6.815/6.865 Digital & Computational Photography

Problem Set 7: Panoramic Imaging

Due Friday, April 8 7:00pm

Panoramic Imaging

In this problem, we'll guide you through the process of creating your own panoramic image. There are a variety of software products out there, and even MATLAB itself provides a few helpful functions. Needless to say, you should avoid using these here (because you wouldn't learn anything if you could just call a provided function).

Problem 1 (6.815/6.865)

Take at least four pictures that you'd like to stitch into a panorama. If you don't have a camera, please contact the course staff for a loaner. When you shoot your photos, it'll be helpful to keep the following things in mind:

- Make sure the images overlap by a significant amount (say, 40%). Small overlapping regions make it very difficult to accurately align the images.
- Shoot all images from same viewpoint, preferably on a tripod. If you start shifting around, then things become a lot more difficult because of parallax.
- For simplicity, you might want to choose scenes with distinctive features. Otherwise, it might be difficult to find correspondences.
- Try to keep the same exposure for all images. If you can, meter and shoot in manual mode. Alternatively, you can use the exposure lock button that comes with most digital cameras.
- Avoid lenses that introduce too much distortion. Wide zooms (even the very expensive ones) generally introduce some amount of barrel distortion, so avoid going too wide. Either that, or postprocess your images with Photoshop's lens correction filter.
- Stick with static scenes. If you shoot a panorama of a busy intersection, it probably won't give you very attractive results.

In your writeup: Show your images and briefly describe how you shot them. Something like "these photos were shot with my Nikon D200 with a 28mm lens at f/8 and 1/60s at ISO 200. I corrected them for barrel distortion in Photoshop."

Problem 2 (6.815/6.865)

Mark correspondences between your images. Say you took four pictures from left to right:

$$\begin{bmatrix} A & B & C & D \end{bmatrix}$$

Then you might identify common features between $A \leftrightarrow B$, $B \leftrightarrow C$, and $C \leftrightarrow D$ (probably 8–12 features per image pair). Or maybe you took four pictures in a square:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

Here, you might use $A \leftrightarrow B$, $A \leftrightarrow C$, and $A \leftrightarrow D$. In general, you should create a tree-like structure of correspondences (no cycles).

In your writeup: For each pair of images that you computed correspondences for (if there are n images, you should have $n - 1$), show the pair of images side-by-side along with the shared feature points. You can draw lines connecting the feature points if you want, but it's sufficient to show dots.

Problem 3 (6.815/6.865)

Using the correspondences, compute homographies between images. Set this up as a system of linear equations (as described in lecture) and use the MATLAB backslash operator.

Now, you can transform pairs of images so that they align with each other. Select one image (probably one close to the middle) as the reference. The homographies that you computed for all images connected to this reference can be used to align those images to the reference. For images that are not directly connected to the reference, you can chain homographies together using matrix multiplies.

For this problem, you should compute the locations of "corners" of each image in the coordinate system of the reference. The minimum and maximum coordinate will determine the frame of the panorama.

In your writeup: Create an image that shows the "layout" of the final panorama: the placement of each individual image should be represented as a (warped) rectangle. You can draw polygons using the MATLAB `line` command. Briefly describe how you got your results.

Problem 4 (6.815/6.865)

Now it's time to assemble the final panorama. In the previous problem you computed the locations of the "corners" of each image in the coordinate system of the panorama. Use this data to warp each image to the proper location (it may help to use the MATLAB `interp2` function, as in the last problem set).

Finally, combine the images together to get your final panorama. This is not as simple as pasting all the images together because you have to account for the boundaries of the warped images. There are a number of ways to do this. Take a look at the `inpolygon` function.

In your writeup: Show your final panorama image, as well as summarizing how you assembled the panorama in Matlab.

Submission

Submit a ZIP file named after your Athena login. This should contain your writeup (PDF), your images, and any MATLAB code that you wrote. All submissions are due by April 8 at 7pm.