Homework 02

due April 20, 2020

Please submit your assignments through BlackBoard. To get you ready for your final project report, you need to use LaTeX to generate the PDF. Also, include all your code and a README that describes how your code can be run. We prefer running one script that generates all your results. Explicitly add any dependencies you may need to your submission.

1 Fitting Lines through Points [30 pts]

In this section, you will fit a parametric model (2D line) to noisy samples that might include outliers. Two sets of 2D points (*pts1* and *pts2*) are included in two text files: **pts1.txt** and **pts2.txt**. Write code to do the following.

(i). **Least-Squares Line Fitting [10 pts]:** Given the first set of 2D points *pts1*, implement a least squares (LS) solution to the line fitting problem. Remember that an $(n-1)$ dimensional plane in $\mathbb{R}^n$ can be represented as: $\mathbf{a}^\top \mathbf{x} + a_0 = 0$ for every point $\mathbf{x} \in \mathbb{R}^n$ belonging to this plane. Equivalently, you can parameterize the plane by concatenating the variables $(\mathbf{a}, a_0)$ as: $\hat{\mathbf{a}}^\top \hat{\mathbf{x}} = 0$, where $\hat{\mathbf{a}} = [\mathbf{a};\ a_0]$ and $\hat{\mathbf{x}} = [\mathbf{x};\ 1]$. Write down the optimization problem and how you intend to solve it. Plot the points in *pts1* and your LS solution for *pts1* on the same plot in different colors. Print the parameters of your best fitted line.

(ii). **LS Line Fitting with Outliers [5 pts]:** Apply your LS code to the second set of 2D points in *pts2*. Plot the points in *pts2* and your LS solution on the same plot in different colors. Discuss how/why this fit is different from the case of *pts1*.

(iii). **Robust Line Fitting with RANSAC [15 pts]:** As we discussed in class, we can handle outliers by using the RANSAC framework.

- Show that the distance between a point $\mathbf{y}$ to a plane $p(\mathbf{a}, a_0) = \{\mathbf{x} \in \mathbb{R}^n |\ \mathbf{a}^\top \mathbf{x} + a_0 = 0\}$ can be computed in Eq (1).

$$d(\mathbf{y}|p(\mathbf{a}, a_0)) = \frac{|\mathbf{a}^\top \mathbf{y} + a_0|}{\|\mathbf{a}\|_2} \tag{1}$$

- Implement a RANSAC-based line fitting algorithm. Elaborate on the details of this algorithm including the minimum number of samples needed for fitting the model, the rule used to determine whether a point is an inlier (HINT: use Eq (1)), and how each line model is scored. Write pseudo-code for this algorithm in LaTeXby using the *algorithm2e* package. Include it in your writeup. Implement this pseudo-code.
- Apply your code to the points in *pts2*. Clearly state the thresholds and parameters you use. Plot the points in *pts2* and your RANSAC-based solution on the same plot in different colors. Discuss your findings.

2 Image Alignment and Stitching [70 pts]

In this section, you will implement a robust alignment method and use it to stitch photographs together to form a panoramic image. You need to make yourself familiar with the *vlfeat* toolbox. Here, you will use the photographs in the *stitching_images* folder.

(i). **[15 pts]** Use the *vl_sift* function to detect and describe SIFT points. Implement a point matching method that takes two images as input and matches their SIFT points according to Lowe's distance ratio rule that we talked about in class. You can use the builtin function *vl_ubcmatch*. Apply this method to the image pair (im01,im02) and plot their point matches using colored lines. Describe this matching method in your writeup.

(ii). **[45 pts]** Implement the normalized DLT method from class. This method should take as input a set of corresponding 2D points and return a homography. Using this method with RANSAC to remove outliers, find the homography $\mathbf{H}_{2,1}$ that transforms im02 into im01. Repeat the process to compute $\mathbf{H}_{3,2}$ and $\mathbf{H}_{4,3}$. Save and output these homographies. Describe this robust alignment method in your writeup.

(iii). **[10 pts]** Using the computed homographies, transform im02, im03, and im04 into the coordinate system of im01. For example, this can be done in MATLAB using the builtin function *imtransform*. After transforming all these images, stitch them together by simply averaging intensities at pixels that are in overlapped regions. Do this for each color channel separately to generate a colored panoramic image, which you will show in your report. Pixels in the panoramic image that are not covered are set to black.