**Report for project 3 Mosaic**

For this project, I searched lots of paper about image Mosaic and read many codes that is written by some experts like VLFeat Library. This library helps me a lot since it’s too complicate for me to finish this from scratch in two weeks. The template on the VLFeat Library helps me a lot to figure out the process of the project.

In a nutshell, I used two templates from internet. One from LVFeat Library and another from initial version of D. Alvaro and J.J. Guerrero, Universidad de Zaragoza (modified by D. Lowe). I will list the functions in detail.

From the VLFeat library (program 1), I learnt how to do the standard DLT and RANSAC. There are three functions for this part, called mosaicTest.mand sift\_mosaic\_no\_normorlizaed\_with\_ransac.m. For the test function, I consider not only two images or several images in sequence, but the images sequences that at least two of them as no overlapping. In other words, the input is random. By choosing the folder for the images, the program will load two images each time, and if the two images have no overlapping it will “mark” the image and skip to the next one until it finds the right image. After the first round is done, if there is a “leftover” image, the program will ask user to choose the folder again, and this time, it will only load image with the“marked” image. By doing this iteration, all the images will be load to the folder and all the correspondence will be established.(bonus)

For the image registration function, it has no normalization but with RANSAC. So for some images combo the function can get a good Mosaic. But sometimes it is still not good for some pictures, or with multiple pictures. The test results are like this:



This one works pretty bad since I use 5 images for input, so the result is bad after done 3 or 4 Mosaic.



If I use two images, however, the result is ok.

The program will show the corresponding H matrix.

For the normalized RANSAC(Program 2). I read lots of paper and programs and implement the “program2” based on some template. I will describe some main functions in detail as follow:(I will skip some help functions)

siftMatch.m

This function reads two images, finds their SIFT features, and

displays lines connecting the matched keypoints.

Sift.m

This function returns IMAGE's SIFT keypoints. I didnt write this function I just used it since the function has encryption so that I cannot read it. But this algorithm is come from D. Lowe. There are some problem for this function since the c program it calls can only handle no more then 9 images. If the number of the input images are larger then 9, it gives me an runtime error.

findHomography.m

This function finds the homography between two planes using bunch of corresponding points.

solveHomo.m

This is the algorithm for getting the corresponding H for homography.m

H33 is uniformed to 1 so that its easier to check.

ransac1.m

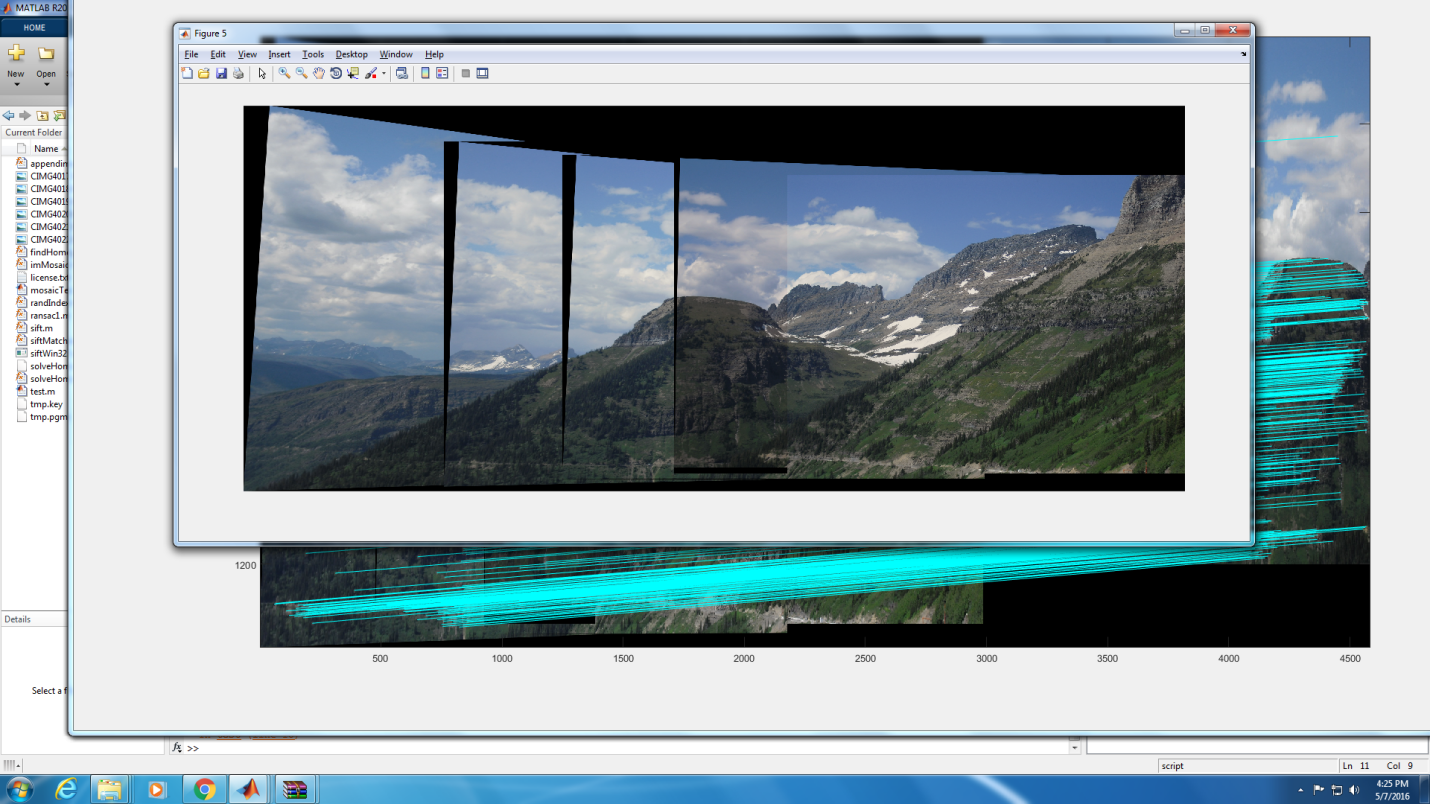
Use RANSAC to find a fit from X to Y.X is M\*n matrix including n points with dim M, Y is N\*n; The fit, f, and the corresponding indices of inliers, are returned. I read some templates for doing RANSAC and modified a lot based on our project. The inliers will be easier to pick. The algorithm is also from D. Lowe.

imMosaic.m

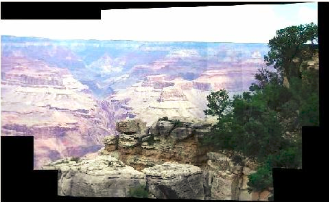
This is a main function that called all the listed function to implement the whole SIFT-RANSAC process. adjColor should be set to 1 to get the best result.

mosaicTest.m

This is a help function. This function loads images and get the result. Recursion is used to load the image dynamically.

I firstly try to call the function several times for each pair image, but I got this:

Finally, I decided to do a recursion for the function to load the files dynamically(9 images). The results are good:



Please run the mosaicTest.m to test the program. I didnt do the program in one shot. What I did was keep calling the function and add image to the sub-mosaic result. Then load the sub-mosaic image to get the whole images

For the bonus, i try some images outside the course.