Computer Vision

Feature Extraction

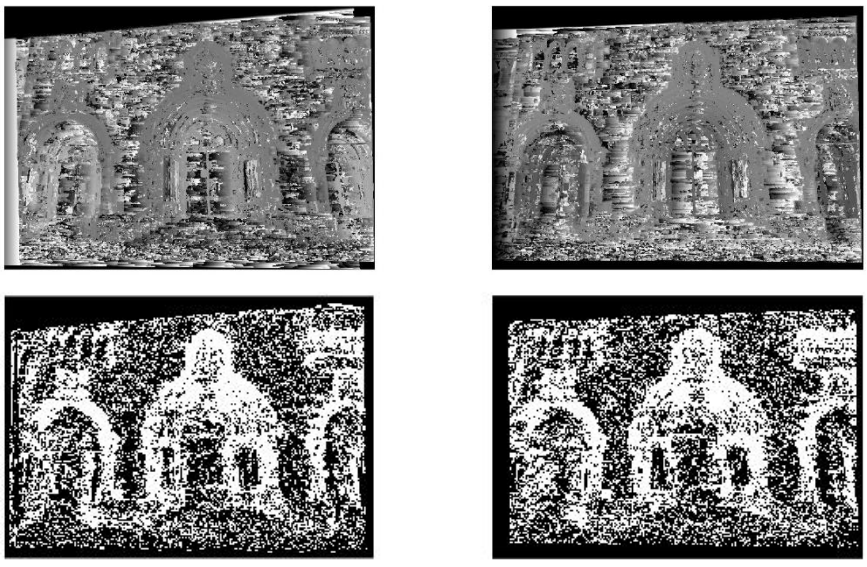
Name: hanxue Liang

Student ID: 16-912-156

In this exercise, I have implemented a winner-takes-all and a graph cut algorithm to compute the disparity map. I also realize the automatically compute the disparity range by using sift matches method. The textured 3D models are constructed using Meshlab.

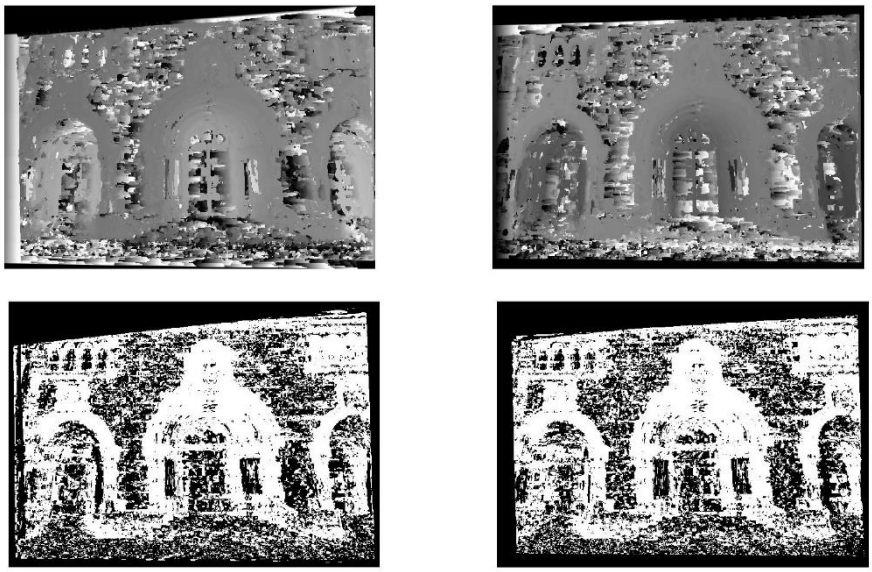
1. Disparity computation

In this part, I implement winner-takes-all stereo using SSD. First I decide the disparity range by manually clicking the points. Then within this range, I use a loop function over all possible disparity value and find the 'winner' disparity value for each pixel. In each iteration, I shift image1 by the respective disparity value, and the calculate SSD error between the shifted image and image2. Then I convolve the result with an average filter. The 'winner' disparity value for a pixel is chosen as the one which provide the smallest SSD error for that pixel. And the disparity maps are shown below. Here, I will compare the disparity map computing from different convolving filter.



WTA result with 3*3 window

WTA result with 5*5 window



WTA result with 7*7 window

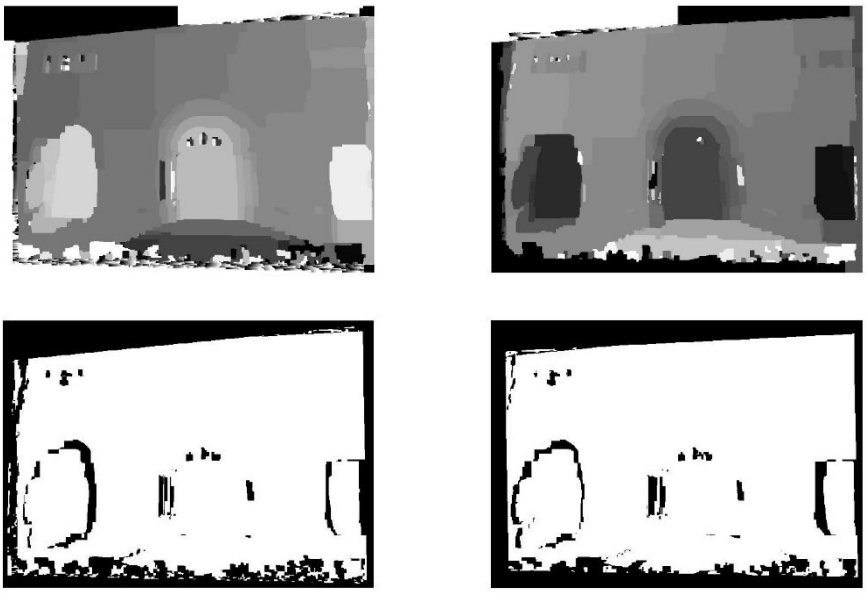
Figure1. Disparity map using WTA method

It can be found that WTA method could provide a good disparity map. When we improve the size of the average filter, a bigger part of neighboring pixels is forced to take the same disparity value. We could clearly see the smoothness effect of growing window size and consequently, the result becomes more accurate for the overall image, although we lost some fine details.

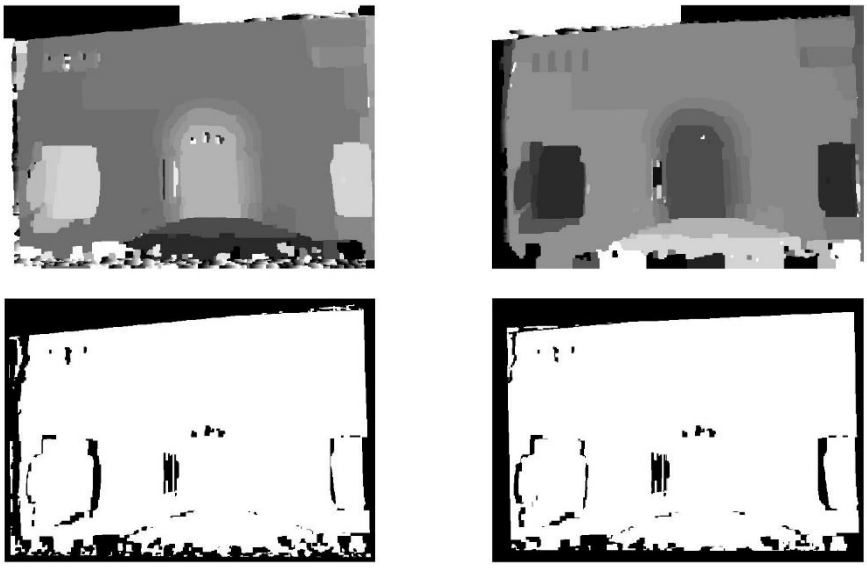
2. Graph-cut

In this part, computing disparity is formulated as a graph labeling problem. The goal is to find a labeling f which map pixel to disparity. The objective function is defined to

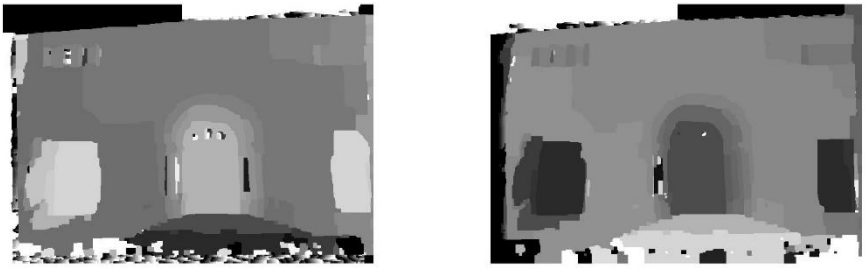
penalizing the neighboring pixels having different disparity values. I put my code for calculating SSD error of the first part in the file diffGC.m and also compute the disparity values with different size of average filter. And the maps are shown below:



Graphcut result with 3*3 window



Graphcut result with 5*5 window



Graphcut result with 7*7 window

Figure2. Disparity map using Graphcut method

For graphcut algorithm, we could find that when we change the size of the average filter, the results don't change a lot. But compared with WTA method, there is a good improvement and the smoothness effect is obvious.

3. Textured 3D model

In this part, I used the result of dense stereo algorithm and the camera parameters, generate a textured 3D model.

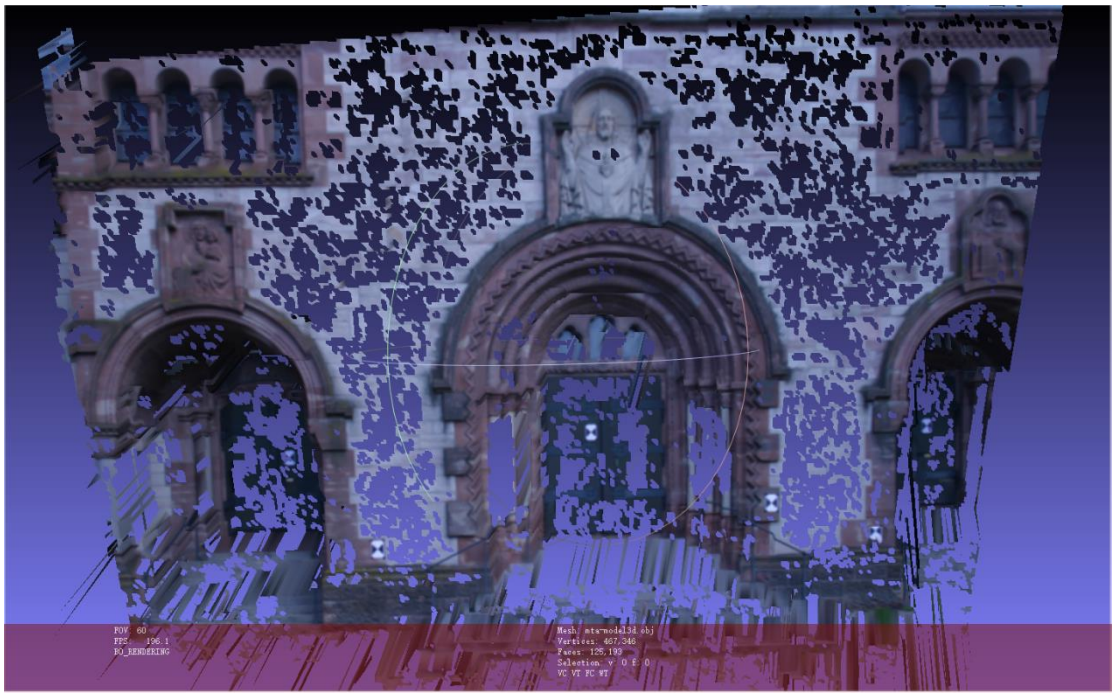


Figure3: 3D model of left image using WTA algorithm

Figure 4: 3D model of left image using Graphcut method

I can find that constructed 3D model using Graphcut method has a better performance than that using WTA algorithm. But for certain part of the object, especially the upper part, constructed 3d model has a deeper point than real model.

4. Bonus

For this part, I firstly sift features from two images and I find the matching points between two images. After that, I use Ransac algorithm to find the inliers matching points. From the inlier points, I get difference of x-coordinates of these matching points in the images. In this way, I get an estimation of the range of the disparity values.

I find that the randomness of ransac algorithm influences the results and the ranges vary quite a lot. Here I choose relatively good result, when the range is specified to be between -7 to 7. It can be found that using automatically computed range of the disparity values can give even better result that manually defining disparity range.

Figure5: Automatically finding disparity range with graphcut algorithm

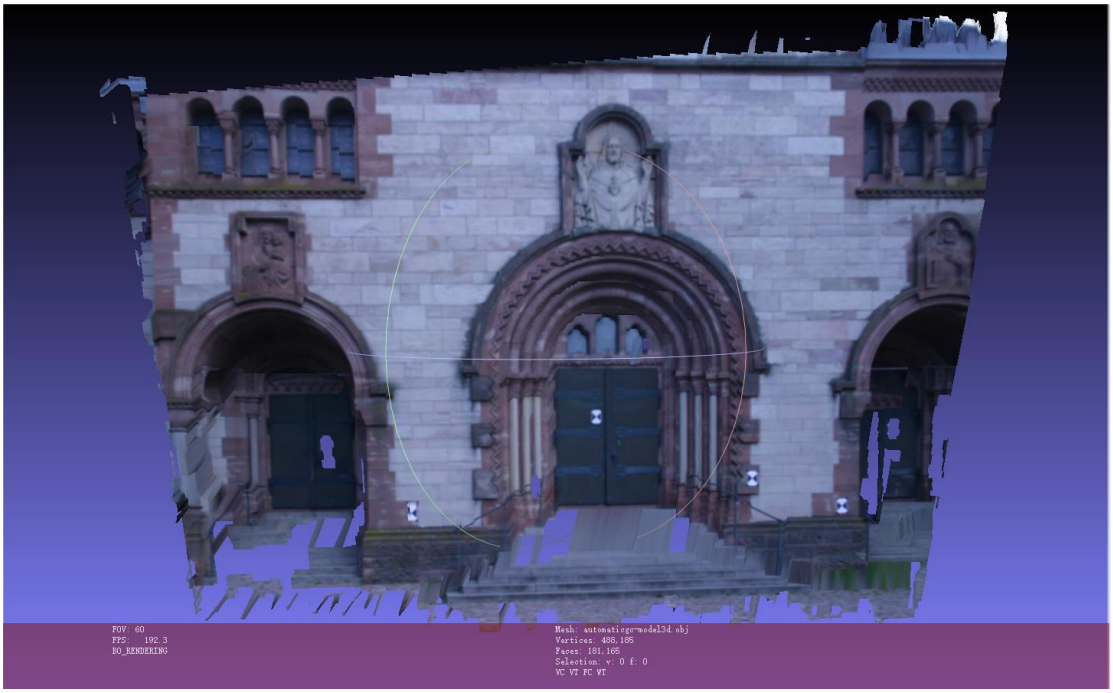


Figure6: 3D model using automatic method with graphcut algorithm