Yue Pan
yuepan@student.ethz.ch

Assignment 6

Computer Vision
November 15, 2019

Task 1: Disparity computation

For this task, please refer to *stereoDisparity.m*.

Firstly, we should do the rectification to make the epipolar lines horizontal so that the correspondences would have the same row index y. Then the disparity can be simply calcualted as the difference of correspondence on x value (column index). The rectification result is shown in Fig.1(b), in which those epipolar lines have become horizontal lines.

(a) original image pair

(b) rectified image pair and the rectified horizontal epipolar lines

Figure 1: Processed dataset before and after rectification

Then a winner-takes-all stereo using SSD (or SAD) is adopted to calculate the disparity map. After a certain disparity range is determined, for each disparity, this method calculate the gray value difference (absolute or square distance) between the corresponding pixels. Then an $K \times K$ average filter is applied to the difference image. This average filter acts as the sum of distance operation of each pixels' $K \times K$ neighborhood, which represents the similarity of two patches. For each pixel, I then remember the minimum the squared difference

(SSD) or absoulte distance (SAD) as well as the corresponding disparity. By traversing the disparity range, we finally determine the disparity map.

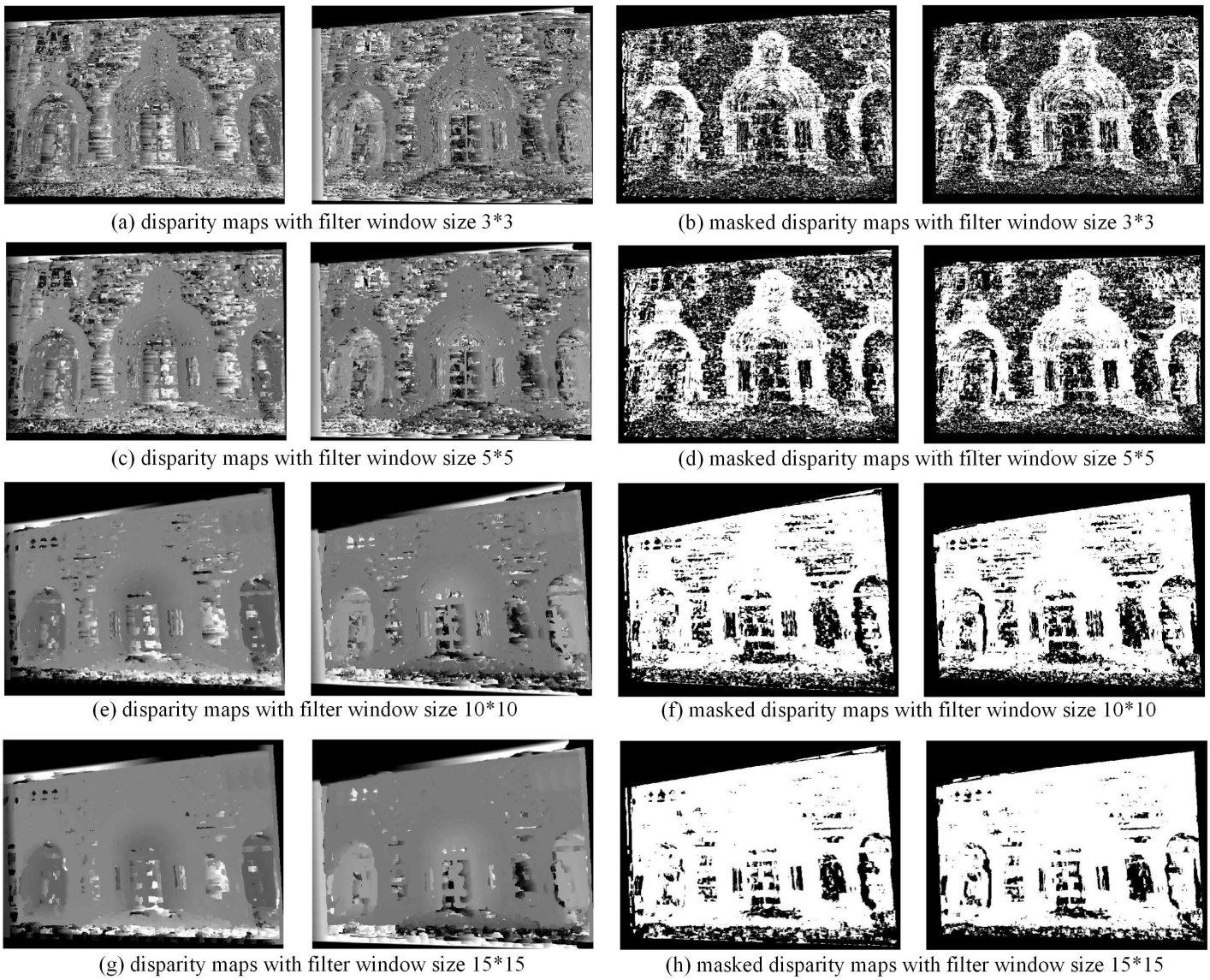
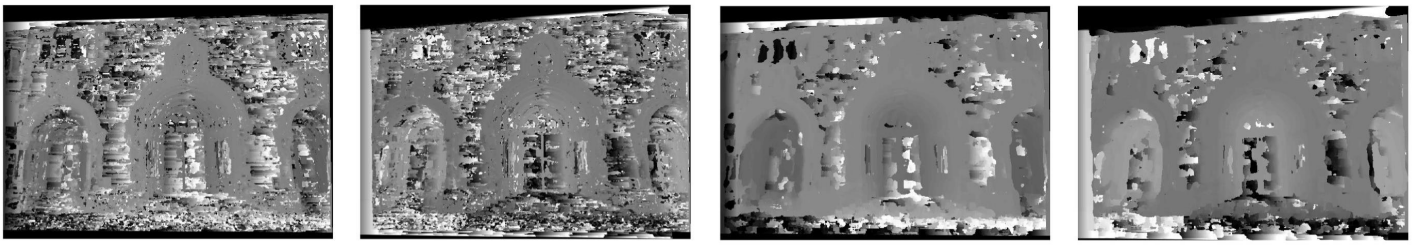


Figure 2: Disparity map generated using winner-take-all method based on SSD

The results of this method based on SSD for different average filter window size are shown in Fig.2. Some of the results based on SAD are also shown in Fig.3. However, it is found that there's almost no difference between the disparity maps generated by them so that I would only consider the case of SSD for the following discussion.

It is shown that with a small average filter window size, the disparity map is less smooth with plenty of depth discontinuities. When we gradually increase the window size, the disparity map becomes much more

smooth. However, since the smooth cost (penalty) is not considered in the winner-take-all method, we can still find some depth discontinuities even when the filter window size is 15×15.



(a) disparity maps with filter window size 5*5 (b) disparity maps with filter window size 15*15

Figure 3: Disparity map generated using winner-take-all method based on SAD

Task 2: Graph-cut

For this task, please refer to *diffsGC.m* and *gcDisparity.m*.

Graph-cut is a global solution to this problem. This algorithm would minimize the energy function which is the sum of data term (SSD of gary value after shifting the image by certain disparity) and smooth term (penalty of different disparity within neighborhood) for the whole image. This is based on the following intuitions: nearby points mostly have similar 3D depth (or disparity) and corresponding pixels should be similar.

Optimizing such kind of global energy function is a NP Complete problem. Methods firstly introduced in (Yuri Boykov et al.,2001) is applied here, as shown in Fig.4. The disparity map generated by this method for different gaussian filter size is shown in Fig.5.

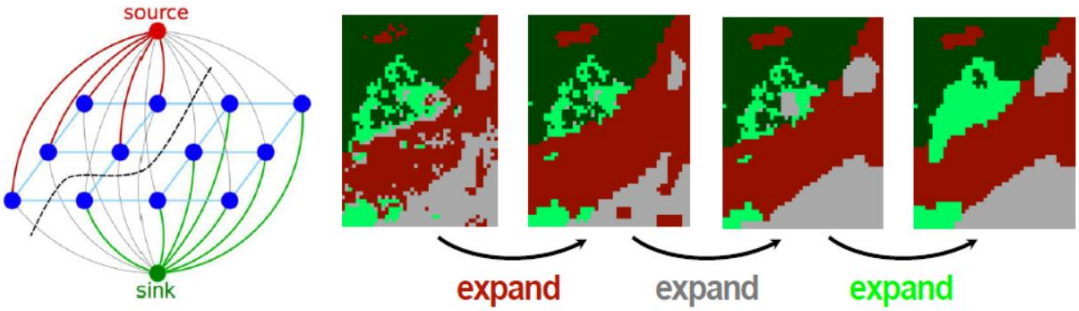


Figure 4: Graph-cut and alpha-expansion (Yuri Boykov et al., Fast Approximate Energy Minimization via Graph Cuts, IEEE TPAMI 2001)

It is shown that the results got by Graph-cut are much better than those obtained by Winner-take-all method in task 1. Even when the filter window is relatively small (3), the disparity map is smooth enough.

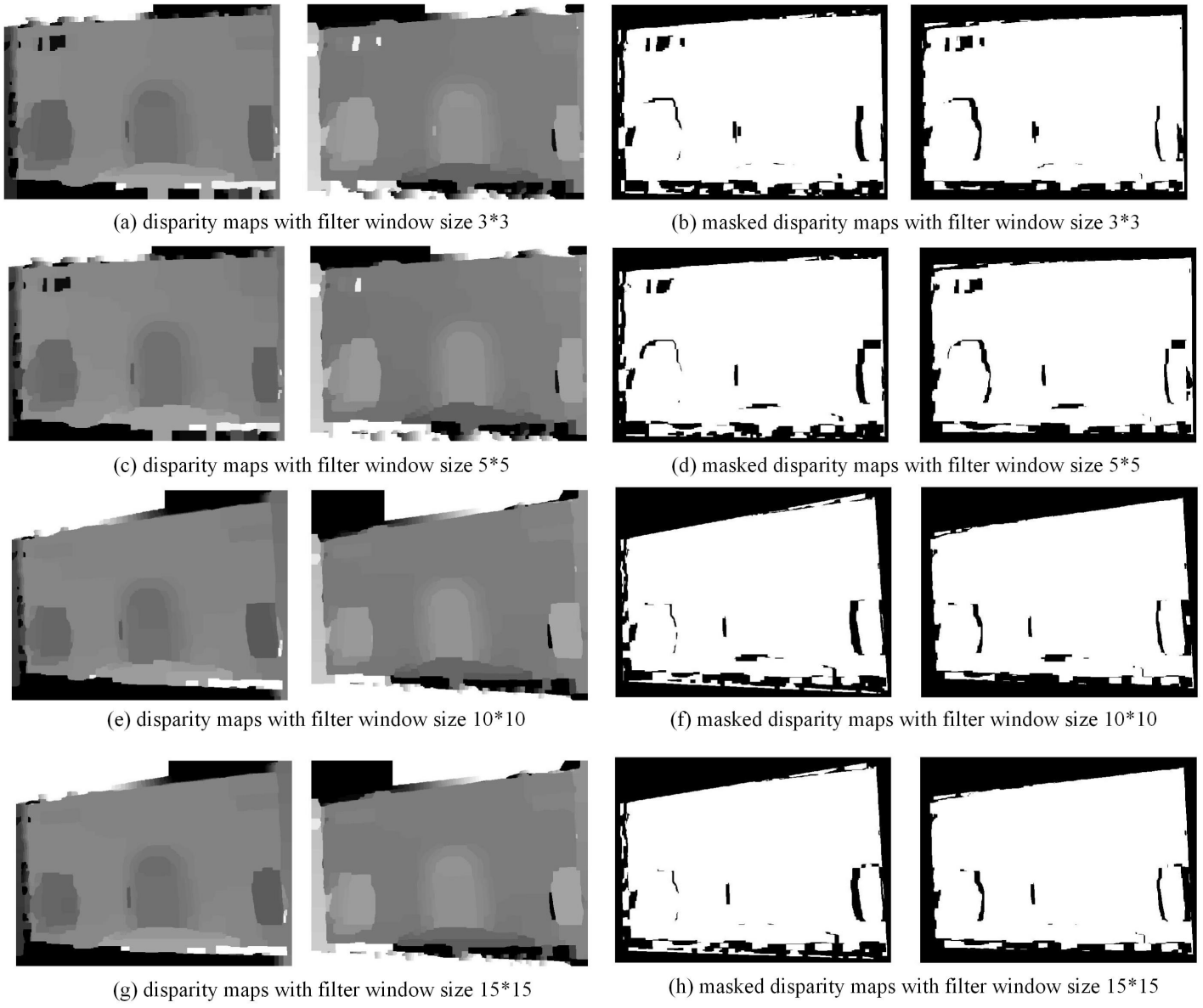
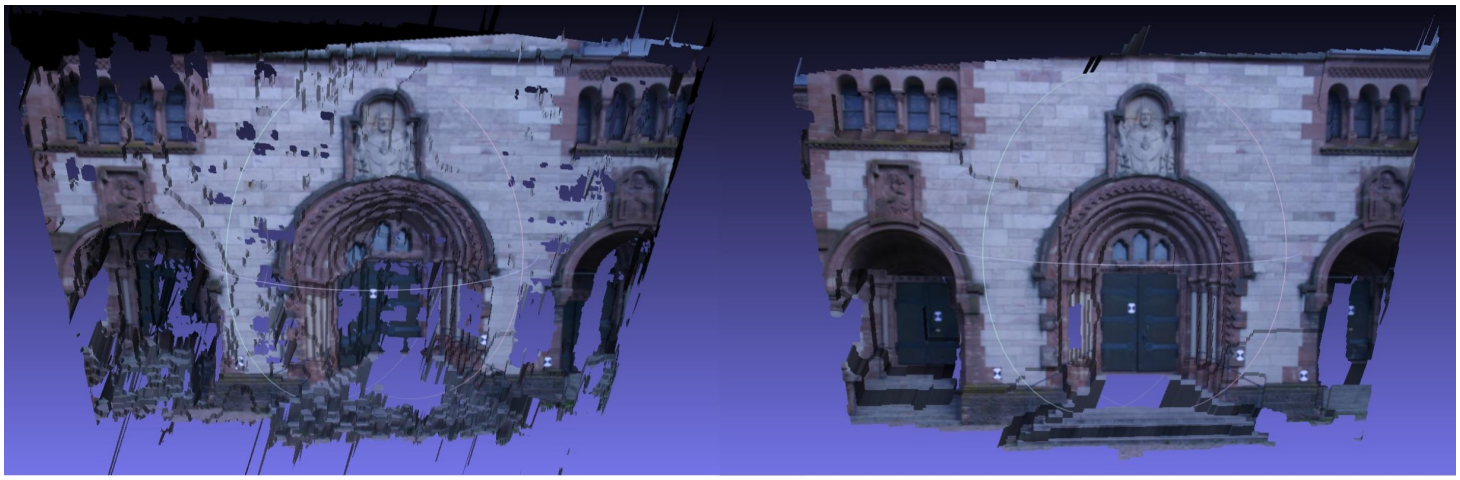


Figure 5: Disparity map generated using Graph-cut method

By increasing the filter window size, it seems that the disparity map just change a bit and the discontinuities in the disparity map almost disappear.

It's clear that the global method like graph-cut can generate more smooth and accurate disparity map than local methods like winner-take-all. However, the drawback of graph-cut is that it's quite time consuming. Even with the help of a C++ library, it still takes 5 more times than the winner-take-all method realized purely by Matlab.



(a) 3D model generated from disparity map made by winner-take-all method (window size 15*15)

(b) 3D model generated from disparity map made by graph-cut method (window size 15*15)

Figure 6: Textured 3D Models Viewed by MeshLab

Given the disparity map, the dense correspondences are constructed. Then we can use triangulation to all these correspondence to generate the dense point cloud, which can be further reconstructed as 3D mesh or models. The results of 3D models generated by disparity map of both of the winner-take-all and the graph-cut methods are shown via the OpenSource software MeshLab as Fig.6.

It is shown that the 3D model obtained with graph-cut are much better than the model generated by winner-take-all method. Even with the large window size 15×15, model generated by winner-take-all has quite a lot holes and some unexpected protruding part of facade. Graph-cut's model is smooth and good enough to recover the actual 3D scene.

Task 3: Automate disparity range

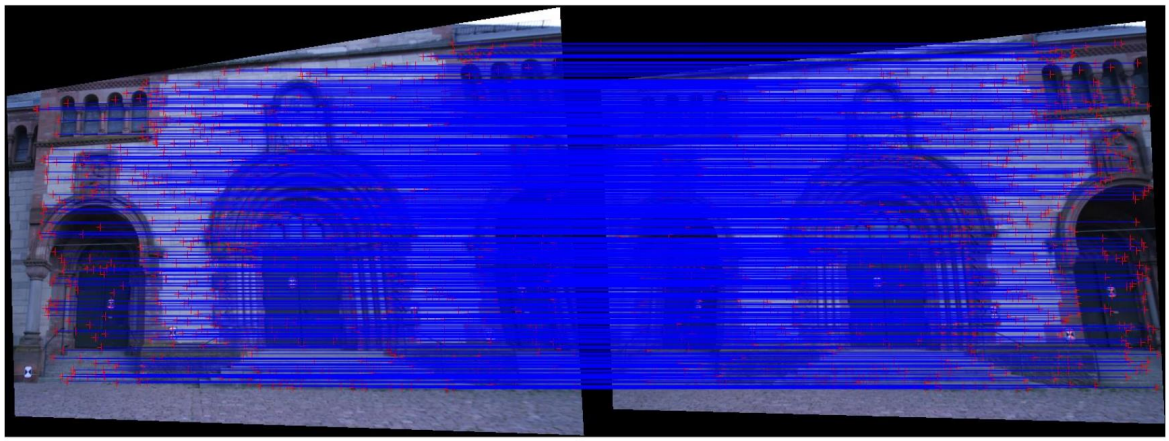
Firstly, a disparity range determination by manually selecting target points is adopted. By using the function *getClickedPoints.m*, I selected more than 10 pairs of target points or context feature points on each image. Then the max and min disparity of these correspondences are calculated as 12 and -11, which are shown in Fig.7. I also add a buffer value 3 to the disparity range in order to cover some texture-less pixels so the final disparity range is set as -15 to 15.

Then, I also try to fully automate this process. SIFT feature points are detected and matched with each other. In order to guarantee there's no outlier matching, three filtering criteria are applied. The first is that a feature similarity ratio test is conducted (ratio threshold set as 2) to filter the mismatch caused by repetitive features. The second one is that a RANSAC (distance threshold set as 3, all-inlier probability set as 0.99) is adopted to further filter some outlier matchings. The thrid one is that those correspondences whose y value are larger than a threshold (set as 1) would be filtered due to the rectification of images. After these three type of filtering, the number of matching decrease from 1850 to 1002, as shown in Fig.8.

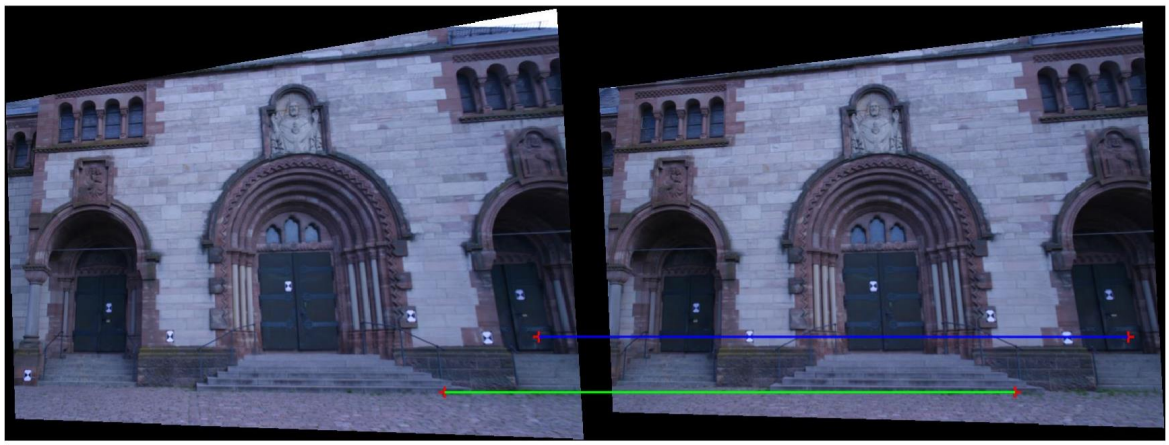
Figure 7: The manually selected target correspondences with the minimum and maximum disparity

Then the min and max disparity of these 1002 correspondences are calculated as -14.7 and 12.1. I use the ceiling value of the one with the larger absolute value then the final disparity range is set as -15 to 15, which is the same as manually set range.

The disparity map generate with winner-take-all and graph-cut methods with such automatic determined disparity range is shown in Fig.9. The smaller range of disparity not only speed up the processing but also increase the contrast of the disparity map.

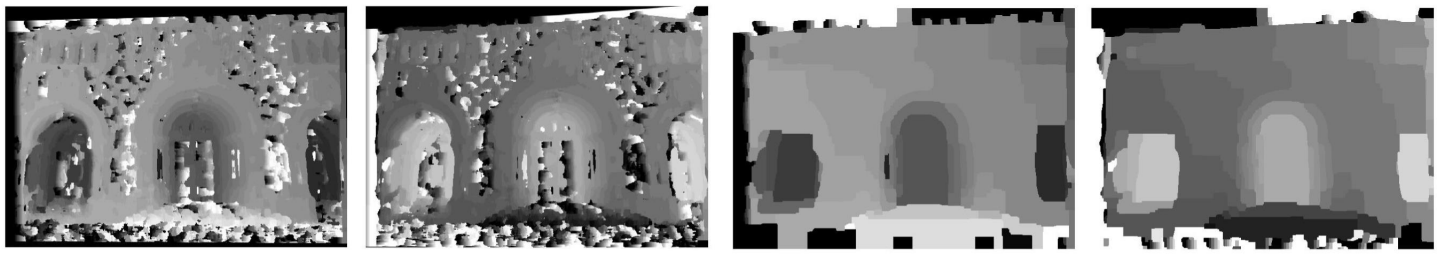


(a) Automatic selected and filtered correspondences



(b) Correspondence with the maximum and minimum disparity

Figure 8: Automatic determined disparity range



(a) disparity maps generated by winner-take-all method

(b) disparity maps generated by graph-cut method

Figure 9: Disparity map generated using the automatic determined disparity range [-15, 15] with filter window size 15×15