Computer Vision
ETH Zürich
Fall 2020

Lab Assignment 6 - Stereo Matching

Mokhles Bouzaien

1 Disparity Computation

The first step was to extract features from both images, determine a descriptor for each feature and then match them. The 2 images used as well as the epipolar lines are shown in figure 1. Then, the 2 images were rectified by aligning the epipolar lines which resulted in images in figure 2.

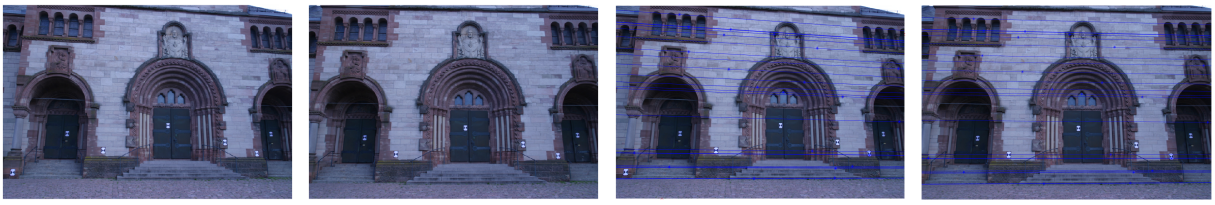

Figure 1: The main images and the epipolar lines.

Figure 2: Rectified images.

The next step is to get the disparity map. To do so, an average box filter was applied to the difference between the image and the shifted one for each value of disparity. The best disparity is kept for each pixel. In this part, I tried many filter sizes, i.e., 3×3, 5×5, 7×7 and 15×15. The results are shown in figure 3.

By increasing the filter size, we get smoother images since the average is applied on wider ranges (figure 3). This can be useful to avoid representing noise but we can also loose details by choosing a big kernel size.

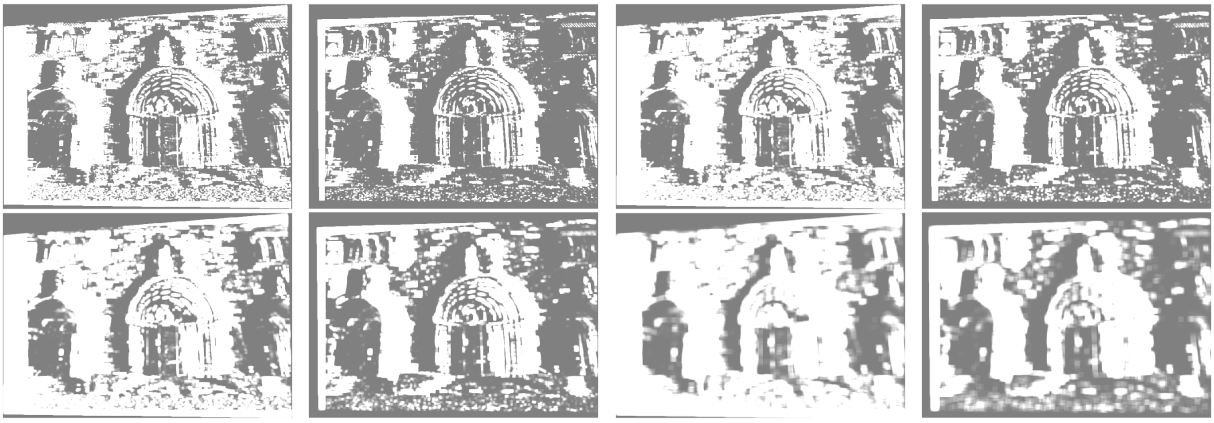

Figure 3: Average filter sizes: 3×3, 5×5, 7×7 and 15×15.

2 Graph-cut

In this part, graph-cut is used to compute disparity. The main idea is to assign a disparity label to each pixel and then minimize a cost function based on SSD value. As we can see in figure 4, results are better than the first solution because even with a relatively small average box there are no noisy depth discontinuities.

Figure 4: Graph-cut disparity results and corresponding masks.

In this part, the generated .obj-files are used to represent the textured 3D models using MeshLab. We can clearly see in figure 5 that the winner-takes-all result is noisy and there are many depth discontinuities. The graph-cut generated mesh is a bit noisy near the edge (since no correspondences exist in this region) but give a better overall result.

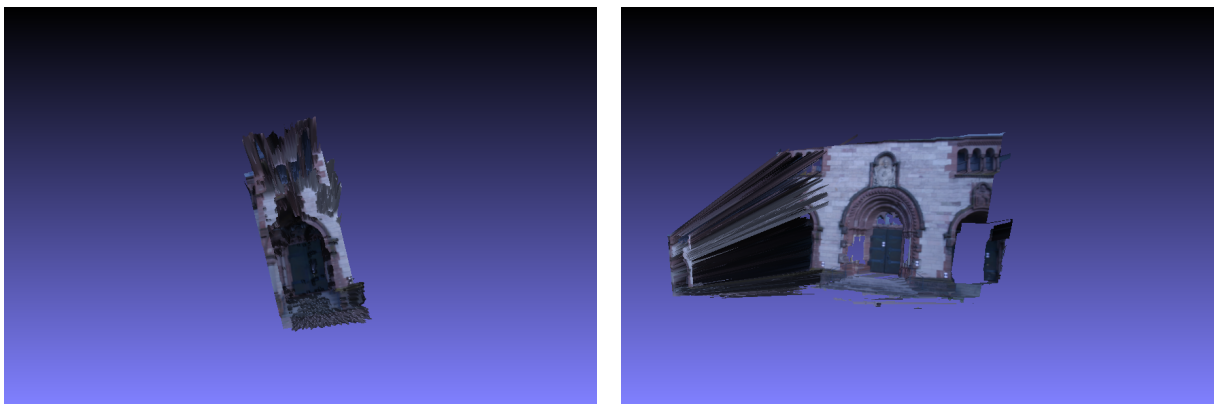

Figure 5: Screenshots of the textured 3D models.