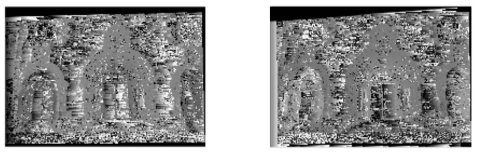
Stereo Matching

Chao Ni

November 15, 2020

1 Winner-take-all stereo matching

In this section, we implemente a winner-take-all stereo matching algorithm to compute the disparity map. For each pixel, it will select a best disparity based on minimum SSD among the given series of disparity candidate. To speed up the process, we use an average convolution to compute the SSD for each disparity. For different selection of the average convolution window, we have different results.


(a) Disparity Map

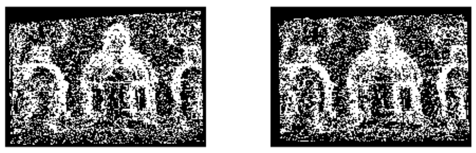
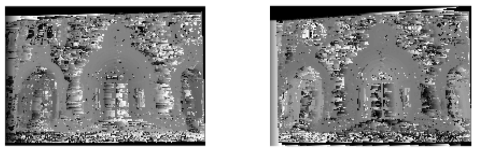

(b) Mask

Figure 1: With winner-take-all stereo matching algorithm and average window size 3, the disparity map is pretty noisy, from mask we can see it is not a good mapping result.


(a) Disparity Map

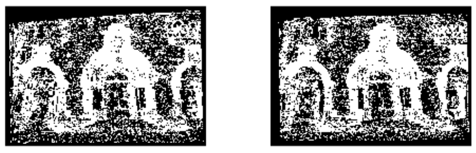
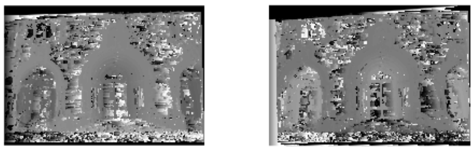

(b) Mask

Figure 2: With winner-take-all stereo matching algorithm and average window size 5, compared to window size 3, now the disparity map is smoother, but still not ideal.


(a) Disparity Map

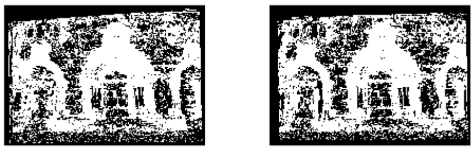
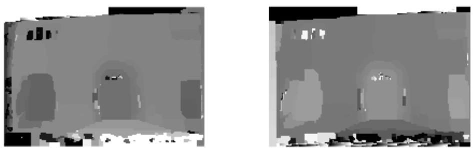

(b) Mask

Figure 3: With winner-take-all stereo matching algorithm and average window size 7, the disparity map is the smoothest one among three.

2 Graph-cut Method

We also formulate the disparity map as a graph-cut problem and solve it numerically. For each pixel, it can be regarded as a node and has a label. We need to label all pixels. To label a pixel with certain disparity, a resulting cost would be generated and consists of two parts. First part is given by the SSD value while the second part is a penalty to having discontinuity in the disparity map. We plot the disparity map with average filter window size 3. Compared with winners-take-all algorithm, Graph-cut is slower, but achieves better performance. It has smoother disparity map because it penalize the discontinuity in disparity map. The out-performance can also been shown by the mask graph, where the majority of the graph is white, which means the disparity map matches most of the time.


(a) Disparity Map

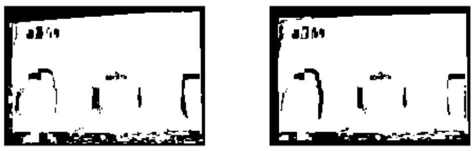
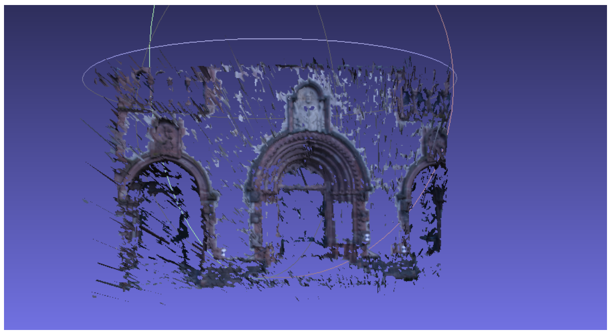

(b) Mask

Figure 4: With Graph-cut method and average window size 3, the disparity map is much better than the winner-take-all algorithm. The disparity map is much smoother because of its penalty to the discontinuity. The mask map is also much better as the dark area is smaller, which shows two maps match better.

2.1 Generating a textured 3D model

Using the results obtained from last two questions, we generate the objects and visualize them.


(a) Winner-take-all: average filter window size 7

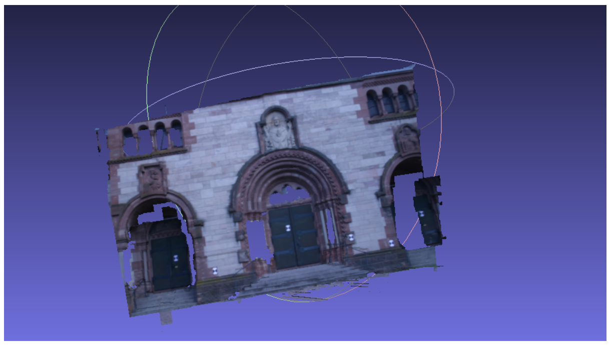
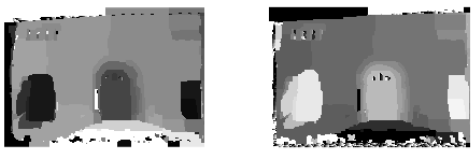

(b) Graph-cut Method: average filter window size 3

Figure 5: We compare the textured 3D model for winners-take-all and Graph-cut method. We use average filter window size 7 for winners-take-all and filter window size 3 for Graph-cut. Even though the first one has bigger average filter, the result of Graph-cut is still much better. It is much smoother, for example, in the wall area. The first picture, however, has lots of noises and we can just reconstruct the door parts.

3 Automate disparity range

In this part, we will come up with a method to automatically compute a candidate disparity range, and hopefully it would potentially be smaller than the fixed one and reduce the computation time for disparity map. We fist extract features using VLfeat for images. Then compute the horizontal differences between feature pairs. To eliminate the effect of outliers, we take the 95 percentage value as the maximum disparity and generate the range based on this. The new range would then be $[-11, 11]$ (we ceiled the percentage value).


(a) Disparity Map

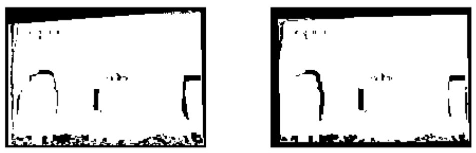

(b) Mask

Figure 6: With Graph-cut method and average window size 3 and disparity range [-11,11].Compared to disparity range [-40,40] which is already shown previously, we have a more contrastive disparity map. If we look at the mask, we can also see the performance is improved, as this mask has more white area.