Computer Vision
Homework Assignment 6 : Stereo Matching

Autumn 2018

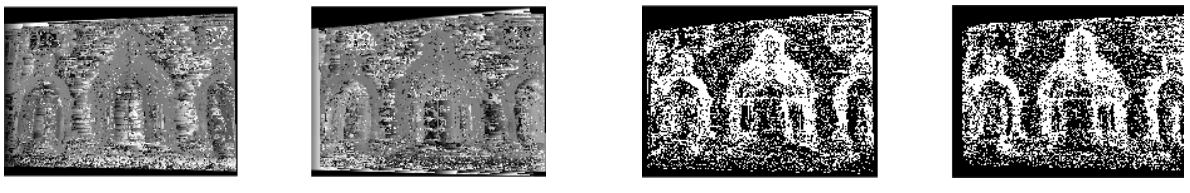
Nicolas Marchal

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

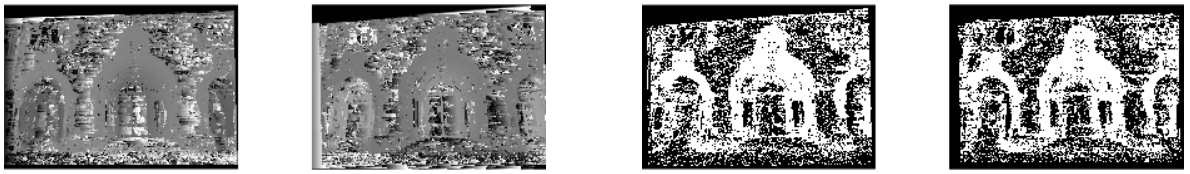
Computer Vision
and Geometry Lab

1 Disparity Computation

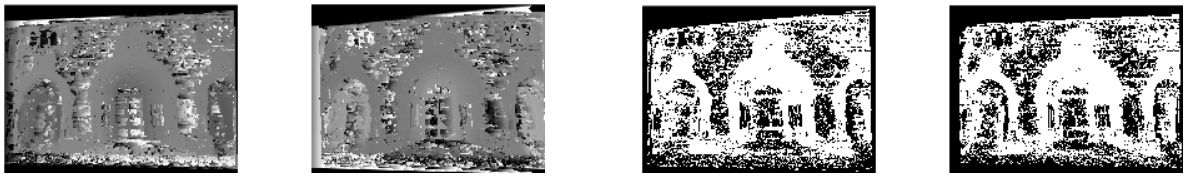
I implemented a winner-takes-all stereo matching using SSD. I used the default disparity range of -40:40 pixels. I evaluated the effect of different window sizes shown in figure 1.



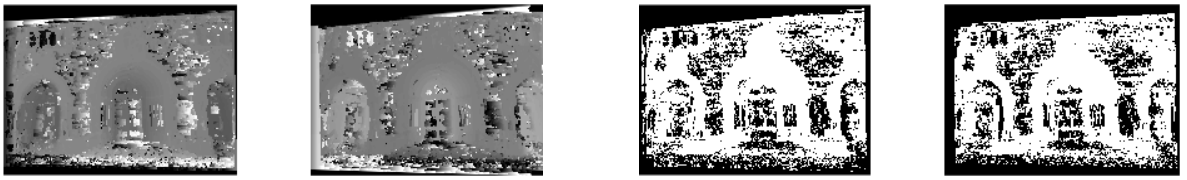
(a) Window 3x3



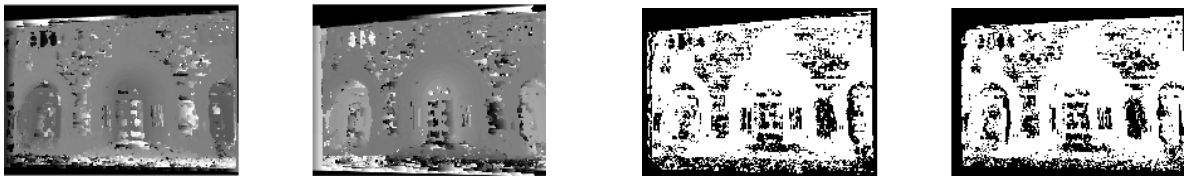
(b) Window 5x5



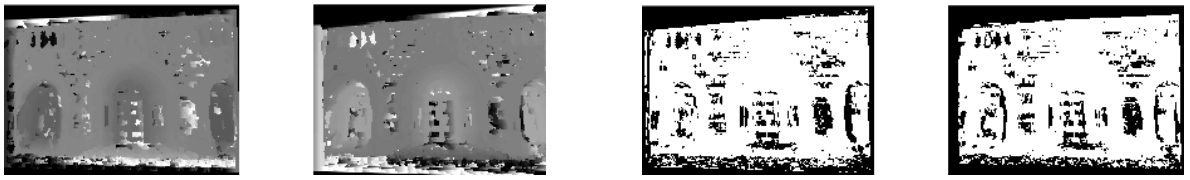
(c) Window 7x7



(d) Window 9x9



(e) Window 11x11



(f) Window 15x15

Figure 1: Effect of window size

Because the window makes an average over a patch, it is intuitive that the larger the window, the less sensitive it will be to noise. As we see on figure 1 taking a larger window does take out a lot of noise, especially on flat surfaces. By looking at the right column of figure 1 we see in white the points that we think have been correctly mapped. The white areas become larger as the window size grows. A large window however could make too much average, and we might miss some small shapes in the image. As we will see in the last part of this assignment, it is necessary to have enough smooth surfaces to be able to reconstruct the textured 3D models.

2 Graph Cut

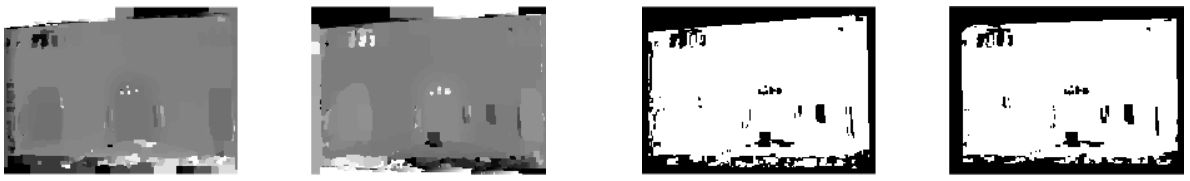
In this part of the assignment, I still evaluate the disparities in the range of -40:40 pixels. When Calculating the SSD, I used a window of 7x7. Using graph cut, I will be able to tune the amount of smoothness I want in my image as I can influence the neighbours to be similar. I reused (and adapted) the sample code and the most important parameters to tune are **Dc** and **Sc** in :

```
gch = GraphCut('open', 1000*Dc, 5*Sc, exp(-Vc*5), exp(-Hc*5));        (1)
```

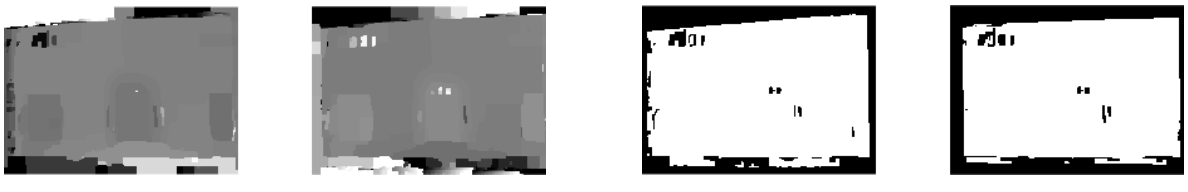
where **Dc** is linked to cost of assigning a label to the pixel of interest (i.e. the larger Dc, the less smooth the image will be). On the other hand **Sc** is linked to the cost of assigning labels to neighbouring pixels (i.e. the larger Sc, the smoother the labels). This method is computationally much more expensive. I have tried different values as shown in figure 2 :

(a) Dc = 1000, Sc = 5 (default)

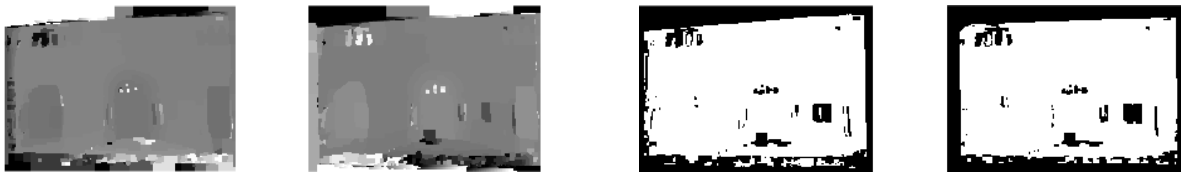
(b) Dc = 3000, Sc = 5



(c) Dc = 5000, Sc = 5



(d) Dc = 1000, Sc = 3



(e) Dc = 1000, Sc = 1

Figure 2: Effect the parameters Dc and Sc

I notice that Dc=1000 and Sc=3 is a good choice and I will keep it for the last part.

The graph cut algorithm offers a much more intuitive tuning and I could easily make a tradeoff between smoothness/continuity of the labels and the precision of my model. As we will see later, the continuity of the labels is desiered to reconstruct the 3D form with texture.

3 Generating a textured 3D model

Having the camera parameters, we can reconstruct the points in the 3D world

3.1 Disparity Computation

For this method, I have shown in figure 3a that taking a too small window will be way too sensitive to noise and the 3d reconstruction is not possible. We can recognize our original image, but it would be very hard for a robot to use this in an unknown environment.

In figure 3b I used a 20x20 window, which gives me much smoother results (note that it is not smooth enough for Mashlab or windows 3d visualization to reconstruct the texture with the original image). In this figure, I can distinguish the doors and the stairs.

(a) Too Small Window (7x7)

(b) Good Window (20x20)

Figure 3: 3D reconstruction with texture - Disparity Computation

3.2 Graph Cut

In figure 4 we see that graph cut got much nicer results for the 3D textured reconstitution of the image. The holes we see on the pictures happen where there is a sharp difference in depth and the algorithm was unsure whether this was due to noise or not. These results could perhaps be used by an autonomous robot to navigate in an unknown environment. The matrix H from the image rectification were also used to get a good reconstruction.

(a) Right View

(b) Left View

Figure 4: 3D reconstruction with texture - Graph Cut

3.3 Comparison

We see that graph cut perform much better to reconstruct the 3D textured model. However, graph cut is computationally more expensive. I have shown that by tuning correctly the parameters in the first simple winner takes all disparity computation, we can improve our results.

4 Bonus - Computing the disparity range

The disparity can be calculated instead of being guessed. We could use SIFT descriptors to match pixels in the image. For all descriptors, we compute the disparity and we will define the disparity range to be from the maximum to the minimum disparity within the SIFT results. We must be careful as this method would be completely biased with a single outlier. We could therefore use RANSAC to filter the outliers from the inliers.

Another method (which I implemented as it was simpler than using SIFT), is to start with a guessed disparity range (-40:40) and plot the histogram of disparities. From fig. 5 we see that the range of -40:40 is a little bit to high, as most of the disparities are centered in -10:10. The very high disparities are most probably due to noise, and it would therefore be good to discard them.

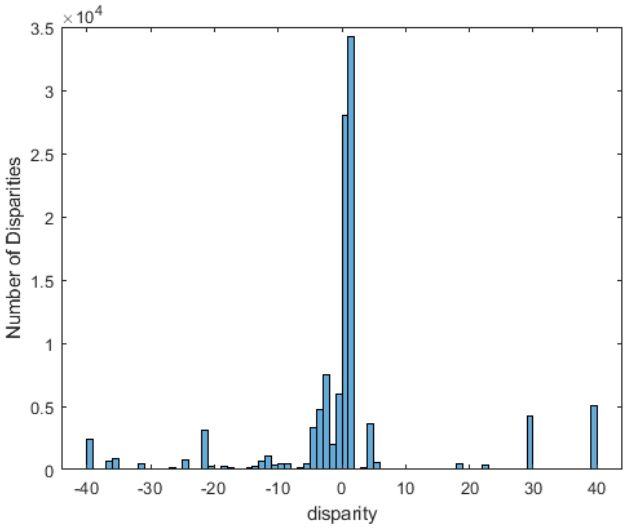


Figure 5: Disparities Histogram