The Exhiliration of Disparity Estimation Experimentation

Disparity estimation is a necessary component of stereo imagery. Disparity defines the difference between two stereo images (for rectified image pairs, this is always a horizontal shift). The disparity can then be used to calculate an estimated depth.
OpenCV provides two standard disparity computation methods: Block-Matching, and Semi-Global Matching.

This report is a summary of results obtained from basic testing of these 2 methods.
The testing will largely involve changing the parameter values (maxDisparity and blockSize) and observing how they affect the disparity map.
I aim to find a correlation between these arguments and the image itself, with the intention of estimating an 'ideal' set of arguments dependent on the image pair contents.

All testing will use:
- the same base source code.
- the same set of image pairs.

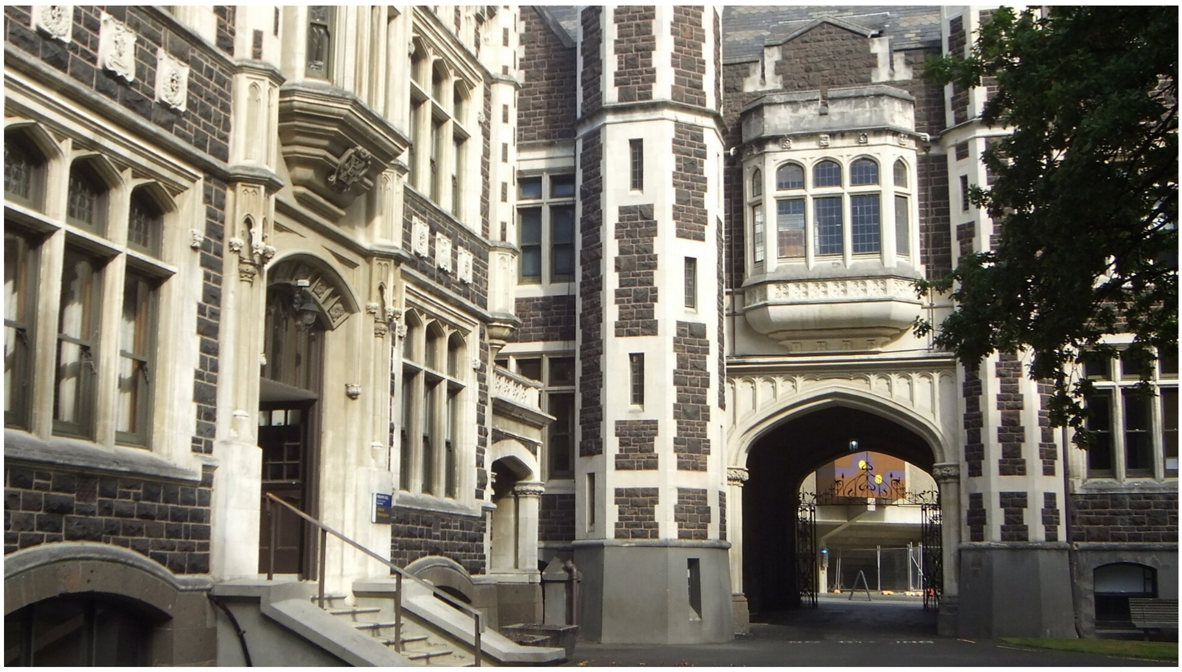
Both of which are provided by blackboard.

Block Matching Experimentation

After creating an experiment and obtaining no meaningful results, I'll instead be testing each image pair independently (just like you told me not to do). I should be able to find a correlation between the image contents and it's 'most accurate' disparity map.
In order to keep my report within a reasonable size, I'll only do a full discussion of 1 image pair and then sum up my analysis of all other disparity maps and their respective images.

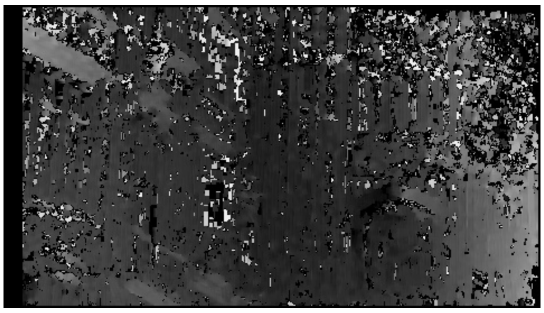
- Archway
maxDisparity from 16 – 80 seem to affect the brightness of the disparity map. Lower maxDisparity makes the map brighter. I'm going to keep the maxDisparity at 16 because from 32 and up it becomes too dark to see any texture.
Choosing a maxDisparity higher than 80 results in lost information in the final disparity map, the left side of the screen is rendered black.

Original Image

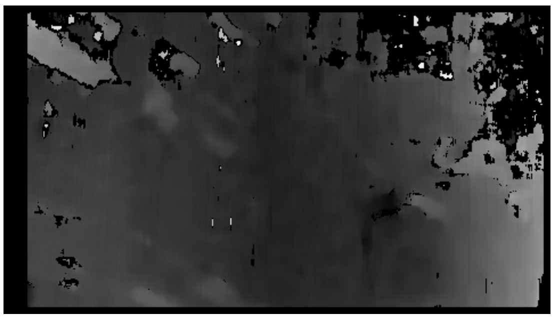


ARCHWAY DISPARITY MAPS (Block Matching)

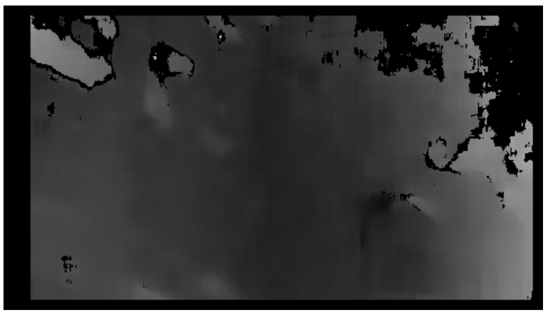
blockSize = 5



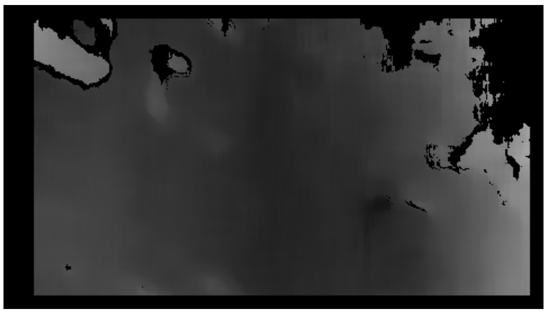
blockSize = 13



blockSize = 19



blockSize = 25



On first impressions, this image pair is not ideally suited for disparity estimation.
The tree is largely rendered as noise with the exception of a few ‘bright’ leaves underneath. The tree should be brighter than the rest of the background as it’s a closer object in the scene. Also, the map hardly resembles the original image.
As you can see, the lower block size contains the most texture from the original image, but also contains the most noise. As block size increases, noise and texture decrease.
All images show the archway but as block size increases, the map starts to blur and become more monotone.
The archway is not represented well in this disparity map. If objects further from the camera are supposed to be darkest then the building through the archway should be darker than the archway. Although, looking at the original image now, the shadow in the archway does look like a dark void.
The only accurate disparity representation I see here (with my very untrained eye) is the far corner is darker than the rest of the scene, as it should be.

This is a “texture heavy” image and I suppose because Block-Matching has no consistency guarantee with neighbouring pixels it has resulted in a lot of lost texture.
I’m expecting other texture heavy images to have similar results.

For the archway, I’d say the ideal blockSize is 21 with maxDisparity 16.
This blockSize is inbetween the ones above where noise becomes minimal without losing too much texture.

All Images are attached in the ‘pics’ folder and are also available through blackboard.

Ideal Parameter Values for each Rectified Stereo Image Pair using Block Matching.

Image Pair	maxDisparity	blockSize
Archway	16	21
Bark	?	?
Bell	64	23
Bookshelf	80	9
Cat	80	5
Desktop	64	5
Flowers	64	13
Hallway	32	21
Poster	?	?
Rock	32	29

All disparity maps resulted in a lot of noise around 'edges' in the scene.

For the bark images I could only ever get noise as a resulting disparity map. I imagine the "texture-heavy", planar, repeating pattern makes it difficult to compute disparity.

The desktop and the cat were difficult to determine blockSizes for. The lower blocksizes were most accurate in terms of depth and texture detection, but resulted in noisy disparity maps.

The depth of the flowers was portrayed well by the contrasting greys. But again the only way to portray texture is to lower block size which results in noisy, 'non-smooth' maps.

The hallway had similar results to the archway. MaxDisparity only seemed to affect the brightness, as opposed to the noise. The major difference is that depth is far more pronounced, which is expected considering the greater change in depth throughout the hallway.

For the poster, once again, a planar scene is difficult to produce a disparity map for. It seems there is a small amount of increased depth to the left of the image but I couldn't find any parameter values which portrayed this in their disparity map.
A low blockSize did make the words slightly visible so it's possible this would be the best disparity map to calculate depth.

The rock was also difficult to compute an accurate map for. The top corner of the rock was always displayed as noise. Again, a texture heavy planar object is included in the scene. This likely has an effect on disparity computation.

Block-Matching Conclusions

Based on this testing, object images further from the camera seem to require a lower maxDisparity value and a higher blockSize value.
Planar scenes are difficult to compute disparity maps for. This is understandable because disparity is inversely related to depth, If there's no depth in the scene there will be little, or no, disparity.
Texture heavy scenes seem to require a lower blockSize in order to better represent the textures in their disparity maps, but this results in more noise.
maxDisparity was much easier to find the optimal value for, there was often one very clear optimal value. Likely due to the drastic change in consecutive values (maxDisparity must be a multiple of 16).

Semi-Global Matching Experimentation

Firstly, the optimal parameter values were unchanged. All disparity maps, no matter the parameter values, were extremely similar to their block matching counterparts.
The main observations of note include;
- Less noise around 'edges' in the image scene.
- Greater contrast between the furthest objects in the scene and the closest.
- Better representation of depth and 3-dimensions, largely due to the improved contrast.

These observations are consistent with the fact that semi-global matching generalizes a range of pixel disparities, whereas block-matching evaluates each pixels disparity independently. This generalization results in less noise and smoother maps.

The mode parameter largely affected the noise, with MODE_SGBM_3WAY = 2 resulting in the least noise every time.
Mode parameter makes no difference for a low block size value but can have a large effect on high block size values (blockSize > 21). Of course there's no point increasing blockSize if the disparity map doesn't properly represent the image scene and the highest block Size I'm using is 29. Any higher makes the disparity maps extremely inaccurate and warped. Still, maps with higher blockSizes are improved by this small amount of reduced noise.

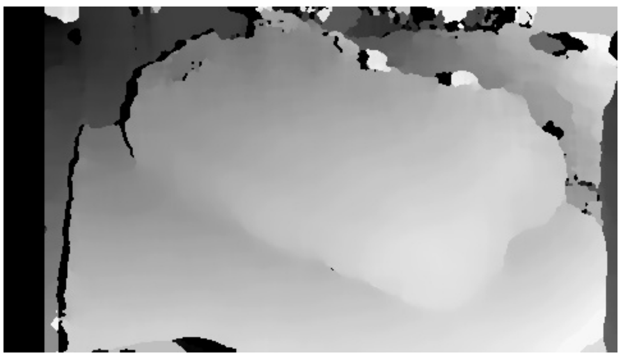
The P1 and P2 parameters have a drastic effect on the smoothness of the map, by setting these to 'ideal' values they could create less noisy disparity map. This is especially useful with all the images pairs that required a lower block size to represent their textures but resulted in a lot of noise.

Using these values to set P1 and P2 resulted in by far the smoothest disparity maps.
P1 = 8 * blockSize * blockSize
P2 = 32 * blockSize * blockSize
(Formula taken from:https://docs.opencv.org/4.x/d2/d85/classcv_1_1StereoSGBM.html#details)

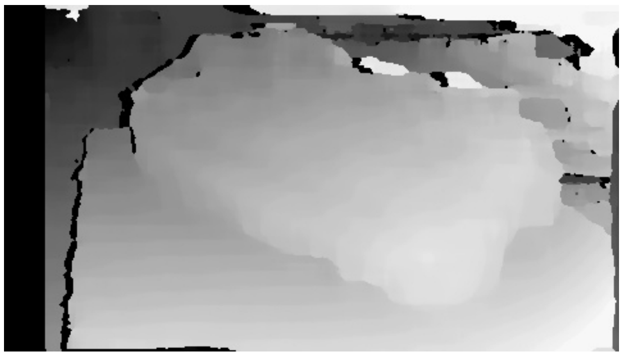
Smoothing did cause issues with any pairs that previously had a high blockSize value (> 10). Before I was increasing blockSize to remove noise without losing texture. These P1 and P2 values allow the blockSize to be much lower than usual because they perform the smoothing that I was trying to obtain from the higher blockSizes. This also means less texture is sacrificed when attempting to smooth.

For example, the rock image pair previously had an ideal blockSize of 29. This made the most accurate disparity map. By changing P1 and P2 (using the formulae above) the previous blockSize of 29 resulted in an unrecognizable map. However, lowering the blockSize to 13 fixed this and even created a more texturous, and more accurate disparity map with very little noise.

Before setting P1 and P2
blockSize = 29



After setting P1 and P2
blockSize = 13



Conclusions

Supposedly block matching is less memory intensive than Semi-global matching. So, there's at least one reason to use it. In terms of visual accuracy, it made little difference and disparity maps were often noiser than the semi-global matching maps.
This is because block-matching analyzes each pixel independently and therefore each pixels disparity has no consistency guarantee with neighbouring pixels.

For each method, the maxDisparity value was unchanged and seems to be dependent on the distance of the objects in the scene. Further testing is required before any definitive proof is obtained for this theory.

Semi-Global matching is by far the superior solution, in my opinion.
Depth was better represented by the increase in contrast. I could actually see the original image in the semi-global maps, whereas with block-matching I usually had to use the assignment specification to decipher the grey blob I was looking at.
This improvement is obvious even before customizing the additional method parameters.

I didn't notice any obvious difference in changing the mode argument, this may have made a bigger difference for disparity maps that required a much higher blockSize value. Mode had no effect for lower block-sizes and my set of test images had very similar blockSize values, for the most part.

The P1 and P2 arguments, of the semi-global matching method, made the most drastic change to the disparity maps. Before using these arguments, I found I was attempting to balance noise with texture. Increasing blockSize reduced the texture of the image scene's map, but also reduced the noise.
By setting P1 and P2, this balancing act was made far easier, if not irrelevant.
- I chose low blockSizes to accurately represent the contents of the original scene, but this resulted in more noise. Setting P1 and P2 reduced this noise drastically without sacrificing accuracy.
-Maps with a high blockSize were smoother but sacrificed a lot of texture from the original images. Setting P1 and P2 allowed a lower blockSize and therefore increased texture in the final disparity map, without increasing noise.

Even though this basic testing favours semi-global matching, it is a very subjective visual test.
I have not taken into account:
- time-taken to compute disparity
- memory required for each method
- exact amount of noise in the final disparity maps

These values are very likely to have a drastic change between the 2 methods and dependent on the overall application of the stereo imagery, may make the difference in which method you should use.

In terms of visual accuracy, Semi-global matching is recommended.

Zac Seales
6687905
seaza886