**Written Assignment 3**

1. Open “conesNoise.txt”. What combination of match cost functions and smoothing give you the best error score? Why do these work best?

I found that using SSD with Gaussian blur of 3~4 generally yielded better error scores than other combinations (though Bilateral performed admirably too). Without blurring, the noise present in the cones image is overwhelming and makes it very hard to match, but with blur applied (effectively denoising the image somewhat), it becomes much easier to process.

Additionally, while not part of what I would consider smoothing, the segmenting the image and averaging the errors within individual segments proved very useful at the default parameters. This also makes sense, given that segments also have an effect of absorbing noise when they take their averages over individual segments. The segment edges aren’t as pretty or quiet as they are in the normal “cones.txt”, but they’re still surprisingly effective.

2. Open “conesGainOffset.txt”. What combination of match cost functions and smoothing give you the best error score? Why do these work best?

With this, I found that both SSD and SAD were again not very reliable at matching on their own (strangely to me, they performed worse for this file than they did for “conesNoise.txt” above…). Adding Gaussian/Bilateral smoothing helped, but this time, the best combination appeared to be SSD+Segmenting. The higher gains allowed for much clearer segment definition, which yielded a much crisper final result.

3. Open “SouthSister.txt”. What combination of match cost functions and smoothing give you the best results when rendering? Do the same parameters work best for “Garfield.txt”?

I found the best rendering results were produced with SSD+Gaussian, with a sigma of 3. Using NCC, Bilateral, or Segmenting just produced too many too many rough breaks in the photo when using the rendering slider, while Gaussian smoothed things out and while there were still breaks, they were not as sharp or visually displeasing as with the others, so I’d have to pick SSD+Gaussian.

4. Download and read the following papers:

[Large Occlusion Stereo](http://x86.cs.duke.edu/courses/spring06/cps296.1/handouts/Bobick%20Intille%201999.pdf), IJCV 1999

Aaron F. Bobick and Stephen S. Intille

[Stereo Matching Using Belief Propagation](http://research.microsoft.com/pubs/64220/stereo_pami.pdf), PAMI 2003

Jian Sun, Nan-Ning Zheng, Heung-Yeung Shum

[Adaptive Support-Weight Approach for Correspondence Search](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.106.118&rep=rep1&type=pdf), PAMI 2006

Kuk-Jin Yoon, In So Kweon

Write a paragraph (6-8 sentences) on each paper describing the benefits and drawbacks of each approach. Which approach would be best for realtime applications? Which paper is best for image-based rendering? Which produces the lowest disparity errors?

The first paper, Large Occlusion Stereo, described a novel way to represent stereo image data in memory such that it becomes easy to quickly identify large occluding objects. This data structure, called the Disparity Space Image, is generated by repeatedly comparing the left and right versions of similar scan lines by overlapping them and storing their difference. This ends up yielding results that can be analyzed to determine both matching points and occlusion objects using the same resulting data. The authors also describe a way to identify Ground Control Points (high-confidence matches across the two stereo images) to improve the efficiency of analyzing the overall images. Finally, the authors describe ways to interpret edges and occlusion ranges such that they provide significant input into their algorithms, to better analyze images, instead of most previous work, which largely ignored occlusion ranges. The techniques in this paper would likely be useful in minimizing disparity errors, since DSIs are not bounded by a maximum disparity length. It would be less likely to be useful for real-time applications given the processing overhead.

The second paper, Stereo Matching Using Belief Propagation, covered multiple ways to improve upon existing algorithms. Belief Propagation is one of the proposed techniques, and they use it to provide extra context to the matching algorithm in an effort to more quickly find the solution. The belief passed is generally the level of disparity that it is expected the algorithm should be evaluating (given prior knowledge either of surrounding areas, or other factors), and this helps the algorithm reduce its search space. Additionally, the authors propose using segmentation to also speed up stereo matching and evaluation. Segmentation divides the images into chunks that are visually similar, and evaluation algorithms can take advantage of segments by assuming that edges are much more likely to appear at the edge of segments. These techniques are likely to be more useful for real-time rendering, as they’ve focused on ways to speed up algorithm evaluation, and have also noted that in experimentation, have found that their results are very much in line with other state-of-the-art stereo processing algorithms.

The third paper, Adaptive Support-Weight Approach for Correspondence Search, perhaps similarly to the second paper, also offered some options to consider for improving existing algorithms. By analyzing how humans tend to observe the world around them, they offer insights on how to better focus processing power. For instance, when looking at objects, humans pay more attention to edges than plain “middles” of objects. Additionally, other qualities like color, position, and relation to each other (much like segmentation) can be used to assign certain regions within a stereo image a higher weight than others. These techniques would likely apply well to photos, or other images, given that techniques from human vision are used to build the stereo picture overall. However, realtime applications are unlikely to be useful here, given that slow performance is called out.

Finally, write a paragraph on how you might improve upon their results by either using a new idea, or by combining the approaches.

Given that the most recent paper we read was authored before 2006 (and the earliest 1999), most of these thoughts have likely already been considered and investigated. The ideas presented in the second and third papers are likely to have more applicability today, since they described multiple ways to attempt to improve existing recognition and matching algorithms, instead of presenting a wholly new algorithm or data structure (like the first paper did). I think it’d be interesting to see how possible it is to combine all of these ideas though. Given the advantages of representing stereoscopic images as Disparity Space Images, what added benefit can we claim from applying human vision recognition patterns to features that the DSIs seek to highlight? Additionally, what other things can we do to speed up matching? Cameras can focus on scenes differently than human eyes can; how useful would it be to attempt to get a rough depth map from images quickly at first, and use that depth map to attempt to prioritize a more complete evaluation of other objects that are in the picture, perhaps that are closer to the camera (suggesting that they are of larger interest)?