CS346
Project III
11 May, 2014

Joshua Hertlein
Ethan Wells

Brief Summary

In this project, we took two images that formed a stereo view of a single setting. From these pictures, we computed the disparity of the two views.

Brief Outline of algorithmic approach

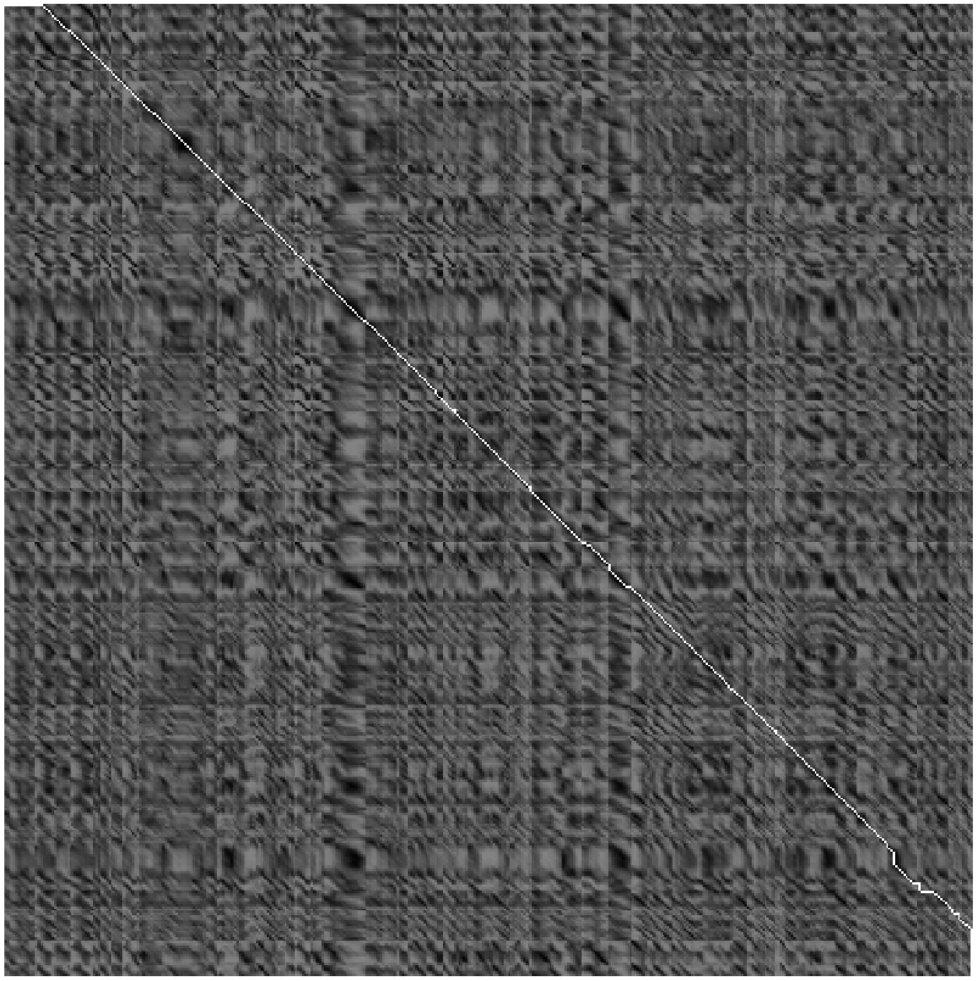
The algorithm used was this:

1. Convert both left and right images into grayscale
2. For each row:
    a. Compute the disparity space image (DSI) using one minus the normalized cross-correlation (1-NCC).
    b. Use a dynamic programming algorithm to compute the min-cost path through the DSI.
    c. Backtrace through the moves the DP algorithm took. If the algorithm went right, the pixel was occluded from the right, and the disparity is set to 0. If the algorithm went diagonally, the disparity is (col_in_DSI - row_in_DSI).
3. Fill in the occluded pixels in the disparity image by taking the nearest valid pixel on the left.
4. Display the disparity image.

CS346
Project III
11 May, 2014

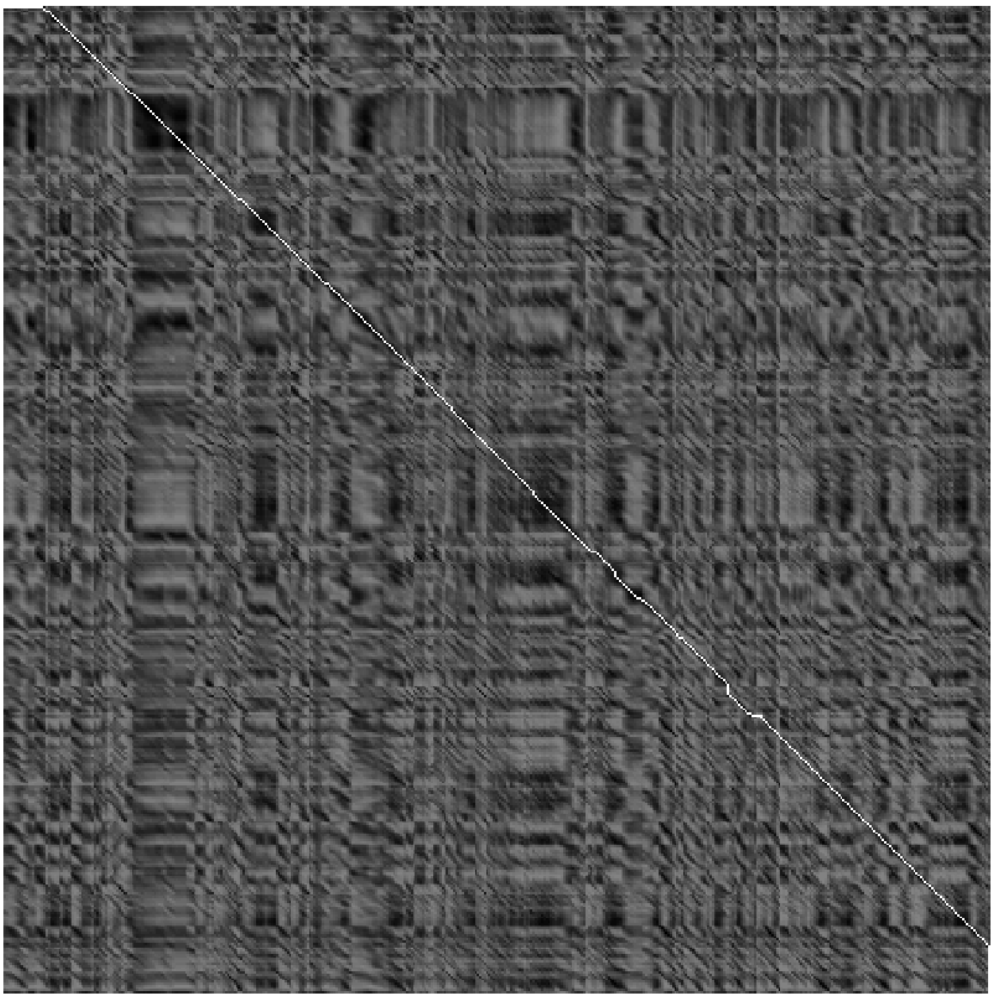
Joshua Hertlein
Ethan Wells

Pictures of intermediate & final results

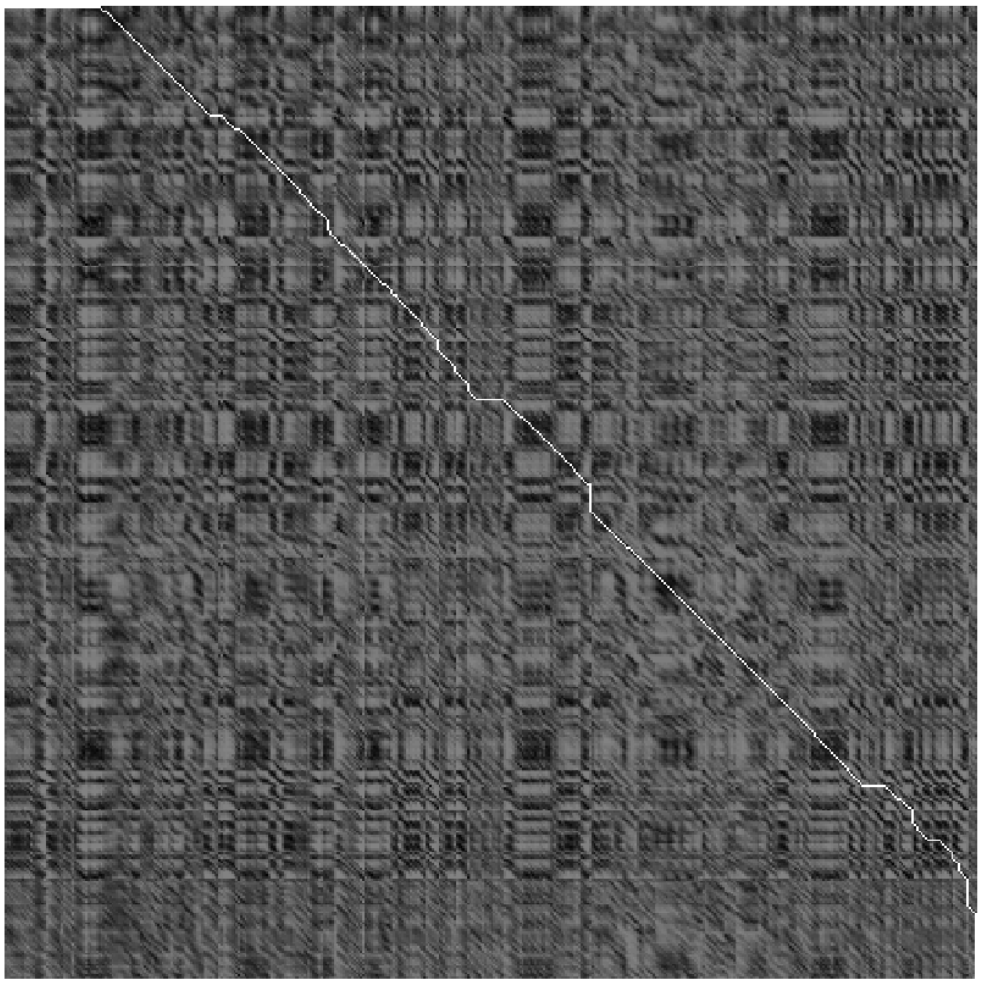
Description	Image
Random DSI + backtrace #1	

CS346
Project III
11 May, 2014

Joshua Hertlein
Ethan Wells

Random DSI + backtrace #2	

Joshua Hertlein
Ethan Wells

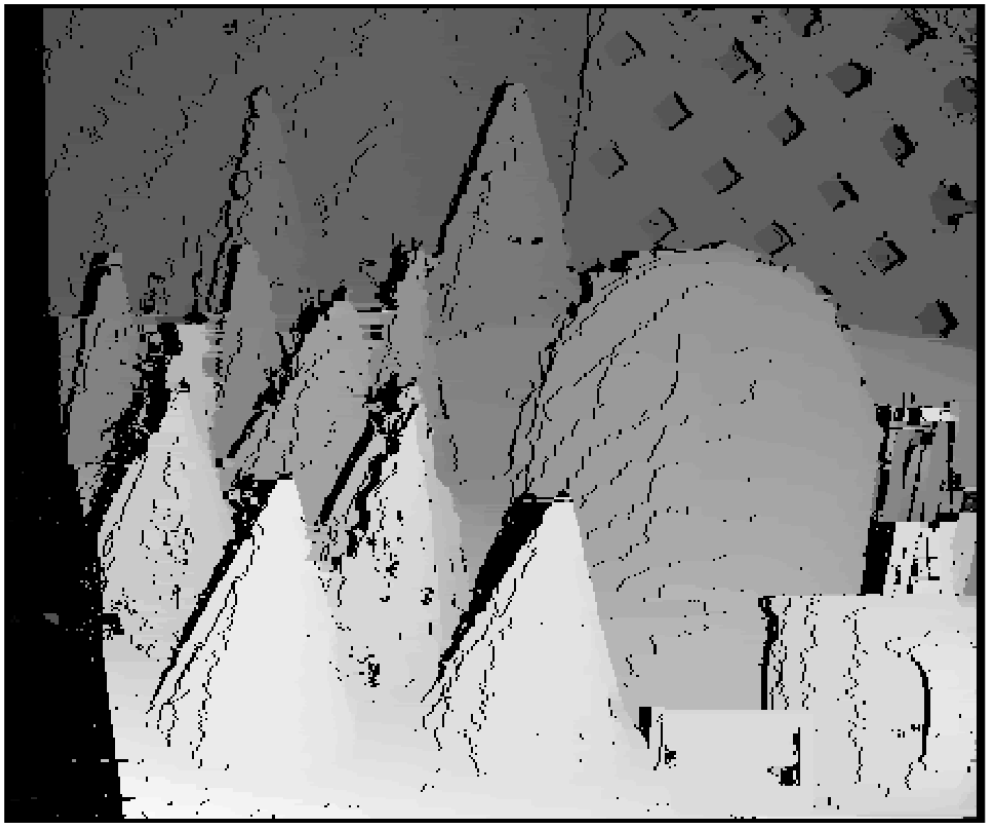
Random DSI + backtrace #3	

CS346
Project III
11 May, 2014

Joshua Hertlein
Ethan Wells

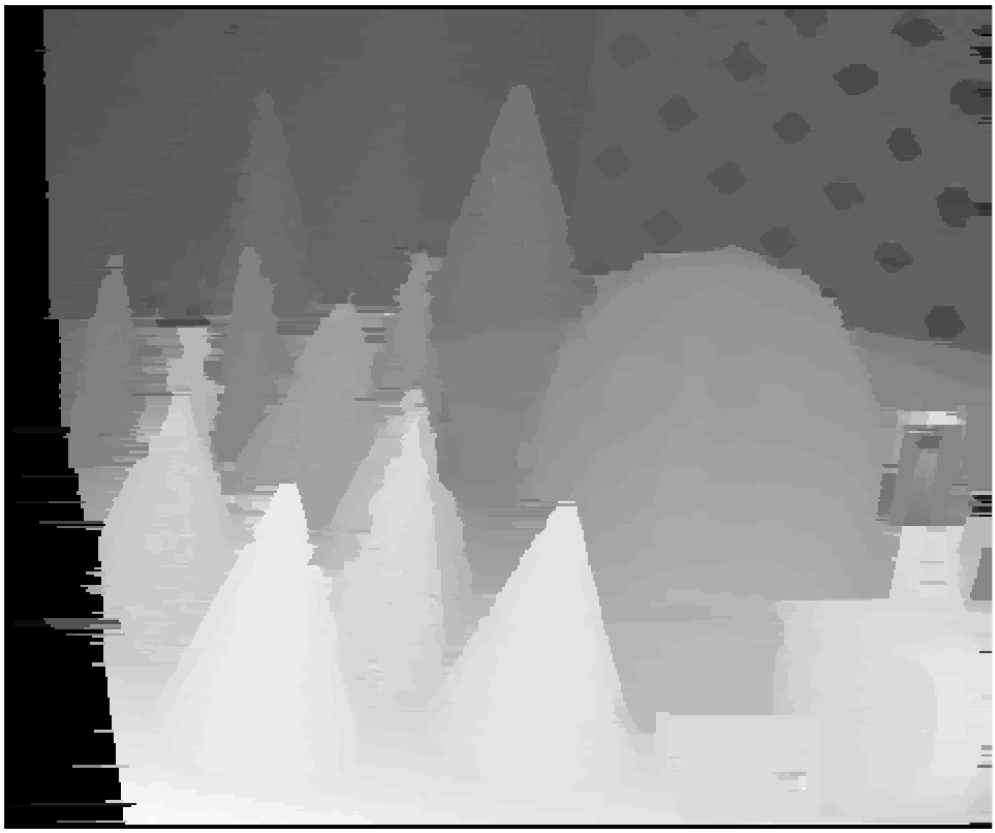
Reconstructed image (w/ occluded areas set to black)	

Joshua Hertlein
Ethan Wells

Disparity image (w/ occluded areas set to black)	

CS346
Project III
11 May, 2014

Joshua Hertlein
Ethan Wells

Disparity image (w/ occluded areas filled in)	

Design Decisions

SSD vs 1-NCC: We used 1-NCC because it uses normalization, which makes it easier to scale the results into a range of [1,255] to display the DSI. It is also faster than SSD in Matlab, since NCC can be calculated as a product of unit vectors, as opposed to using multiple nested for loops.

Occlusion filling: We used the nearest valid pixel on the left to fill in the occlusion. It had to be a valid pixel, because if the pixel was occluded then there was no disparity value for it.

We performed the stereo reconstruction on the left image, because it was always thought of as the “primary” image while we were traversing the DSI for the dynamic programming algorithm.

Experimental Observations (patch size)

Changing the patch size to be larger than 10 pixels didn’t seem to change the results much, but making the patch size too small greatly decreased the accuracy of the DP matching algorithm.

CS346
Project III
11 May, 2014

Joshua Hertlein
Ethan Wells

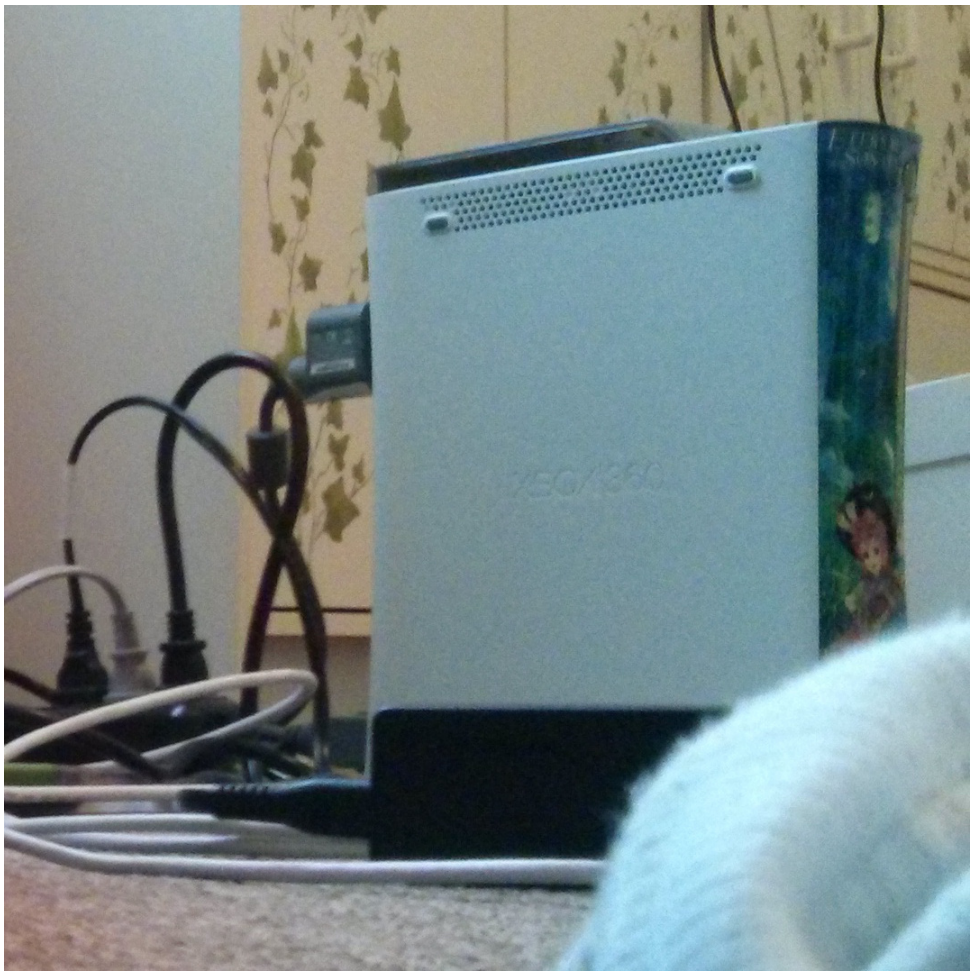
Member Contributions
We both contributed equally

Extra Credit

For the extra credit, I took two pictures of the mess of cords surrounding the game consoles in my apartment.

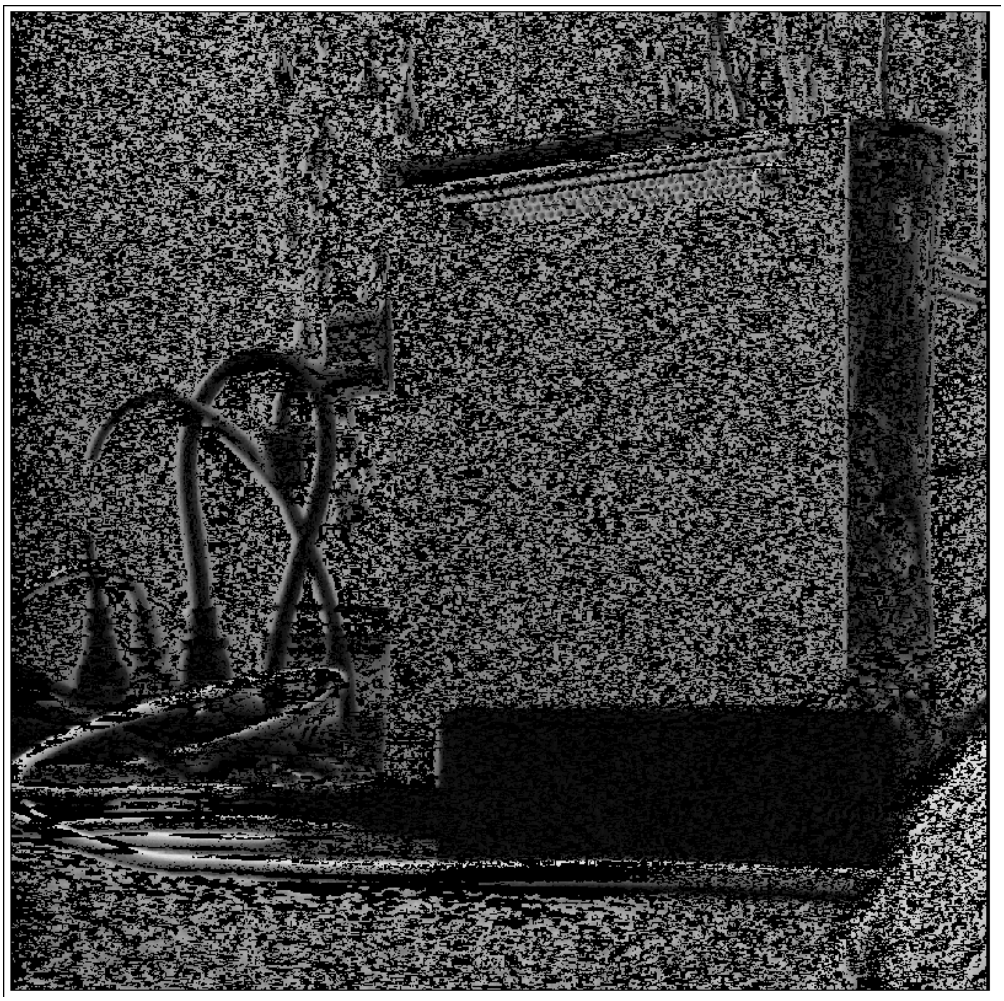
Description	Image
Left	

Joshua Hertlein
Ethan Wells

Right	

CS346
Project III
11 May, 2014

Joshua Hertlein
Ethan Wells

Reconstructed	

CS346
Project III
11 May, 2014

Joshua Hertlein
Ethan Wells

Disparity, no filling	

Joshua Hertlein
Ethan Wells

Disparity, filled	

These results didn't come out very well, for a few reasons:

- My camera takes pictures at an extremely high resolution
  - this makes the program take a very long time to execute (10+ hours)
  - scaling or cropping down to size both introduce/amplify noise
- The images are JPG images
  - both cropping and scaling require it to be re-compressed, which adds artifacts to the image
- The scene had large areas of uniform color (side of XBox, the carpet, etc)
  - This would make it hard to correctly match corresponding patches

So, there was so much noise in the image, it was unable to get a good result. As a possible solution, we could've used a filter that averages/smooths the image- but this would also slow down an already slow process, as well as making it even harder to match corresponding patches.