CS 435 Homework 1

Sarah Kushner

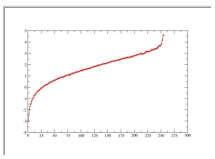
April 24, 2016

1. **combine_exposures.py**

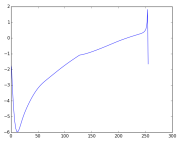
   Despite my lack of numpy knowledge and being confused by some typos in the lecture slides, I implemented code to compute the response functions and to combine the exposures using them.

   The main problem I ran into for this part was making sure the right values were put into the right places in the A matrix and b vector. What tripped me up the most about this was the extra equation $g(128) = 0$. To me, this meant setting $A[NP+ncolors-2] = 0$, but the Matlab code sets it to 1. The slides, which also set it to 0, furthered my confusion.

   My response functions plotted look like this. They are not what I expected them to look like based on the slides and the paper. They do, however, look almost exactly like the ones on this page. See below:

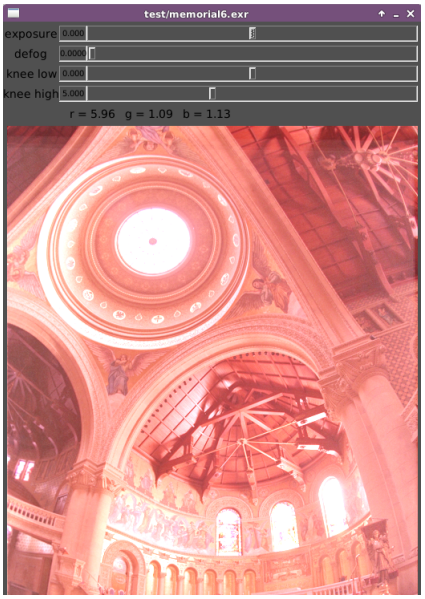


(a) The mentioned site's response function.

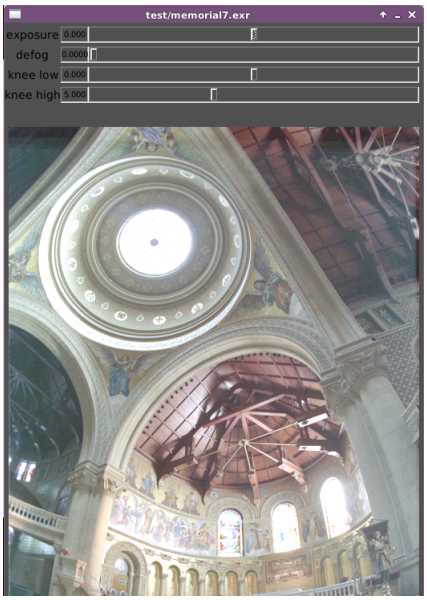


(b) My response function.

   But the resulting HDR exr images looked fine for both the garden and the main building photo sets. For the memorial photo set, my exr was tinted red:



At first, this led me to believe that my response functions were not completely correct. But after running `combine_exposures.py` on the memorial images with the sampling parameter higher than the default 4096 (I used double that, 8192), it looked much better:

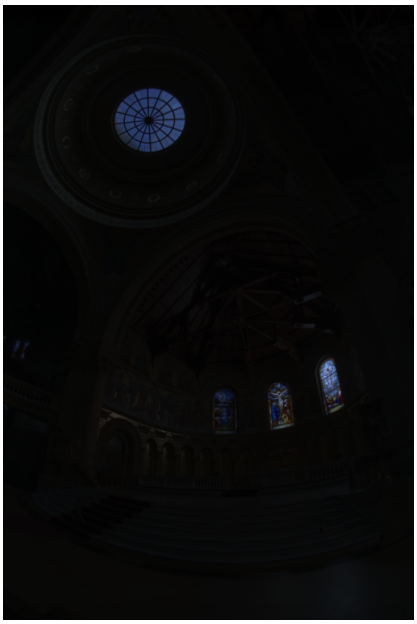


2. **tonemap.py**

   In this part of the assignment, I wrote code to compute a bilateral filter and then use that to tone map my HDR image. Then I rescaled each color channel that it could be displayed as a LDR image.

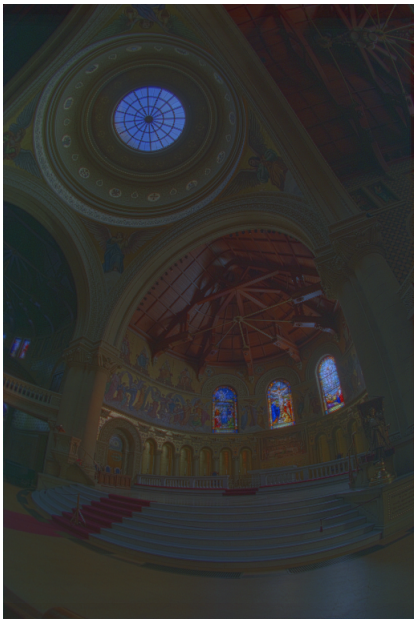
   My first (minor) problem I came across was how, once I had three separate images for each channel, to put them all back into one image. I learned a lot about numpy doing the second part of this assignment because of this problem. I learned specifically about `numpy.dstack()` to combine three channels into one image. I also found out about this cool syntax trick. Since these `ndarrays` are 3 dimensional, I can access just one color channel by typing `image[..., channel]`. Pretty cool.

   My main problem during this part of the assignment was that my images initally came out too dark:



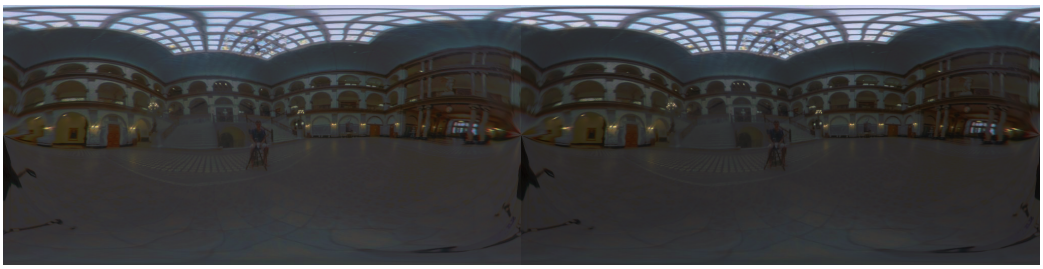
   This turned out to be because in my bilateral filtering function, where my kernel was a 3×3 window, I was accessing the same 9 pixel window for every pixel location. Once I fixed that, it was a little better:

Then I played around with the dR value until I got one I liked. Here, dR is 4:



This is the garden photo with $dR = 5$:

And finally, the main building with $dR = 5$ and 6:



(a) $dR = 5$.

(b) $dr = 6$.