Computational Photography
Homework 4 – High Dynamic Range Image and Tone-mapping

Amiya R Panda
UIN - 727006179

Method

1. Initially, the directory containing the images from which camera caliberation function is to be deducted is given as an input to “ParseFiles()”. This function extracts the exposures per each image from their names and stack into an array and returns it. Also, we get the list of filepaths of each image as the output from this function.
2. Then, the log (base e) is taken for the exposure array and stored in B for further usage.
3. Then after reading one of the images from the “filepaths”, size of the image and from the length of it, the number of images is extracted.
4. Then, I sampled 80 random pixels locations out of the image size range and stored their values for each image in the filepaths in the matrices “Z_r, Z_g, and Z_b” (for red, green and blue channels). This is done as each channel is to be processed separately. The value of the pixels is converted to 1-256 format for further processing.
5. After that, the triangular function (weight “w”) is made but the array form of function is used for further processing to decrease the computational complexity.
6. The camera response functions(“g_r, g_g and g_b”) for different channels is fetched using the “gsolve()” function and plotted.
7. Using the above mentioned strategies, now, the target images and their metadata are extracted but this time for achieving better computational performance, the images are stacked onto a 4-D matrix and passed to the “reconstruct_hdr()” function.
8. The “reconstruct_hdr()” function is the realization of Eq. 6 in the Debevec paper which iterates over each image to calculate the radiance map for the final output. This step is computationally the heaviest and performed for each channel separately. The output of the function is logarithm of E.
9. After extracting the value of E, then global tone-mapping is performed for each output channel and thereafter they are stacked to make the final image matrix.
10. Global tone-mapping is realized by the formula
    Output = (a*input)/(a*input+1)
    And output is observed for different values of “a”.

Observation

There are a lot of scope for improvements in the implementation. Firstly, by using more space, the computation can be reduced which takes on an average of 11 minutes. This can also be achieved by using more vectorized operations rather than looping and iteration.

Tone-mapping

Global Tone Mapping ---The formula used for global tone mapping here is

Output = (a*input)/(a*input+1)

While this filter provides a decent contrast for parts of the image with low luminance , parts of the image with higher luminance will get increasingly lower contrast as the luminance of the filtered image goes to 1.

Another global tone mapping method is gamma compression. In this method, the "gamma" regulates the contrast of the image, a lower value for lower contrast. While a lower constant "gamma" gives a lower contrast and perhaps also a duller image, it increases the exposure of underexposed parts of the image while at the same time.

Local Tone mapping --- A more advanced group of tone mapping algorithms is based on contrast or gradient domain methods, which are 'local'. These operators concentrate on preserving contrast between neighboring regions rather than absolute value. This approach is motivated by the fact that the human perception is most sensitive to contrast in images rather than absolute intensities. Global tone mapping methods usually produce very sharp images, which preserve very well small contrast details, however, this is often done at the cost of flattening an overall image contrast, and may as a side effect produce **halo-like glows around dark objects.**

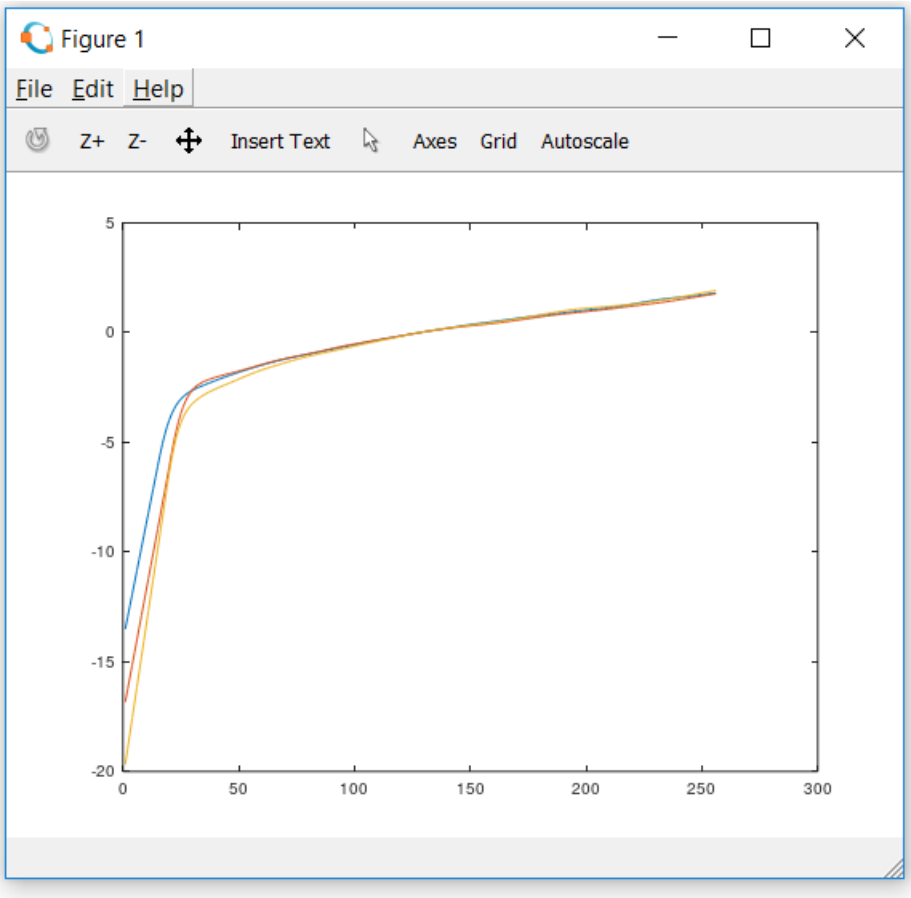
An implementation of such technique can be realized by the following algorithm

1. Input is the RGB values of radiance
2. Compute the intensity (I).
3. Compute the chrominance channels: R/I, G/I, B/I
4. Compute the log of the intensity: L = log2(I)
5. Apply bilateral filter: B = bf(L)
6. Compute the detail layer: D = L-B
7. Apply an offset and a scale to the base: B' = (B-o)*s
8. Reconstruct the log intensity: O = exp(B'+D)
9. Put the colors back: R',G',B' = O*(R/I, G/I, B/I)
10. Apply a gamma compression

Results

For 0_Calib_Chapel and 1_Bicycles

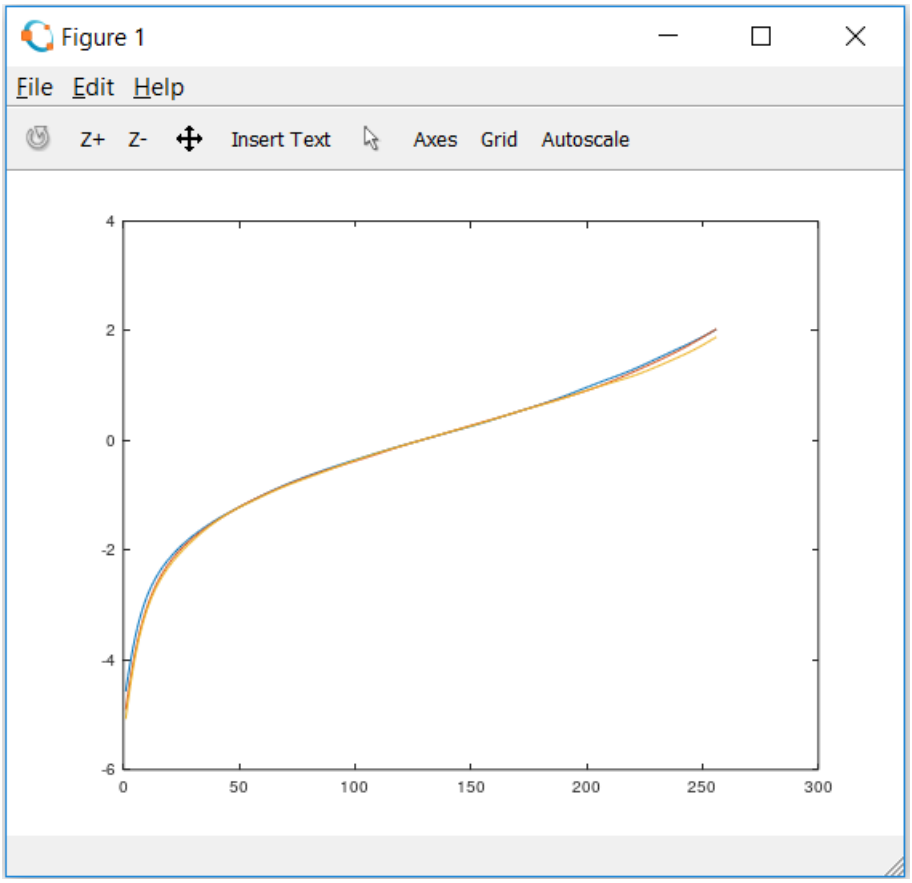
Plot g_r, g_g, g_b



Radiance Map

For 1_Calib_Office and 1_Statue

Plot g_r, g_g, g_b



Radiance Map

Tone mapping

For a = 0.01

For a = 0.01

For a = 0.1

For a = 0.1

For a = 1

For a = 1