Akshat Thirani
November 8, 2014
EECS 395

Homework 4

High Dynamic Range Imaging and Tone-mapping

The purpose of this homework was to explore the dynamic range properties of images that were taken using the Tegra tablet.

The first step was capturing a sequence of images that we would later process. A series of images were taken with different exposure times. A simple loop was programmed into the tablet, which increased the exposure in every shot by a factor of 2. I ended up with 4 images.

The stream of images was loaded onto my computer and the analysis and processing was done using Matlab.

Camera Response Curve:

To obtain the camera response curve, the function gsolve was provided to us which gave the outputs as the camera response curve g and and the recovered log radiance lE for each of the pixels that we input to the algorithm.

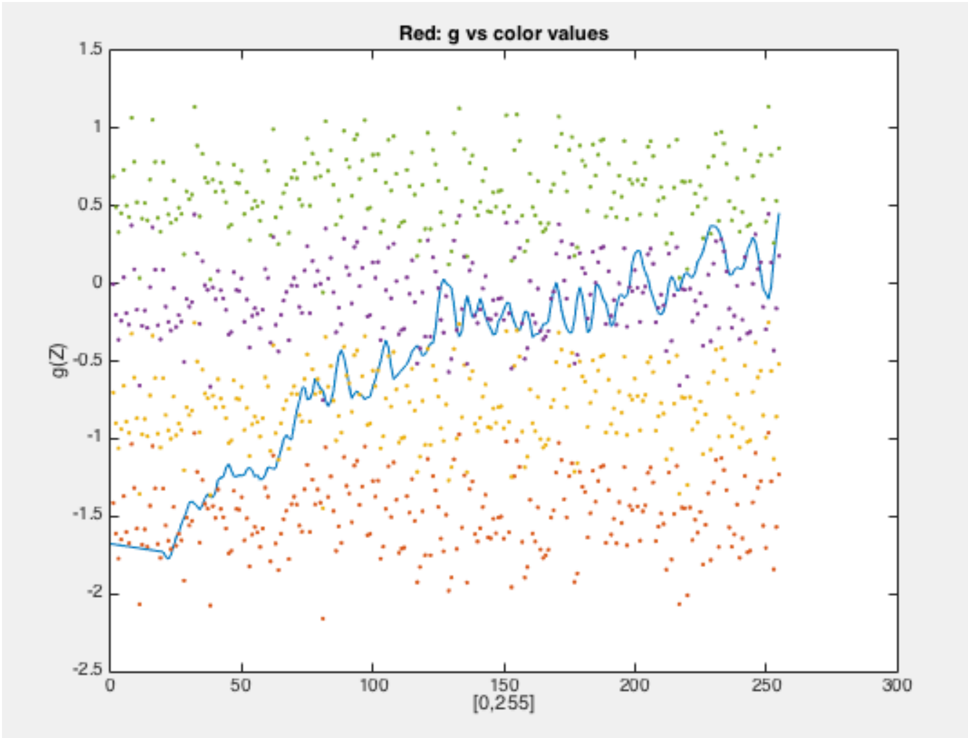
The input to gsolve were a random sample of 1000 pixel values, the log of the exposure times and a constant l which determined the smoothness. This was done separately for the red, blue and green channels.

g was then plotted against the range of pixel values [0, 255]. On the same graph, the log exposure of each pixel used as output for gsolve was mapped after computing its exposure value as: $X(i,j) = lE(i) + \ln(B(j))$, where lE is the log exposure and $B(j)$ is the j exposure lengths.

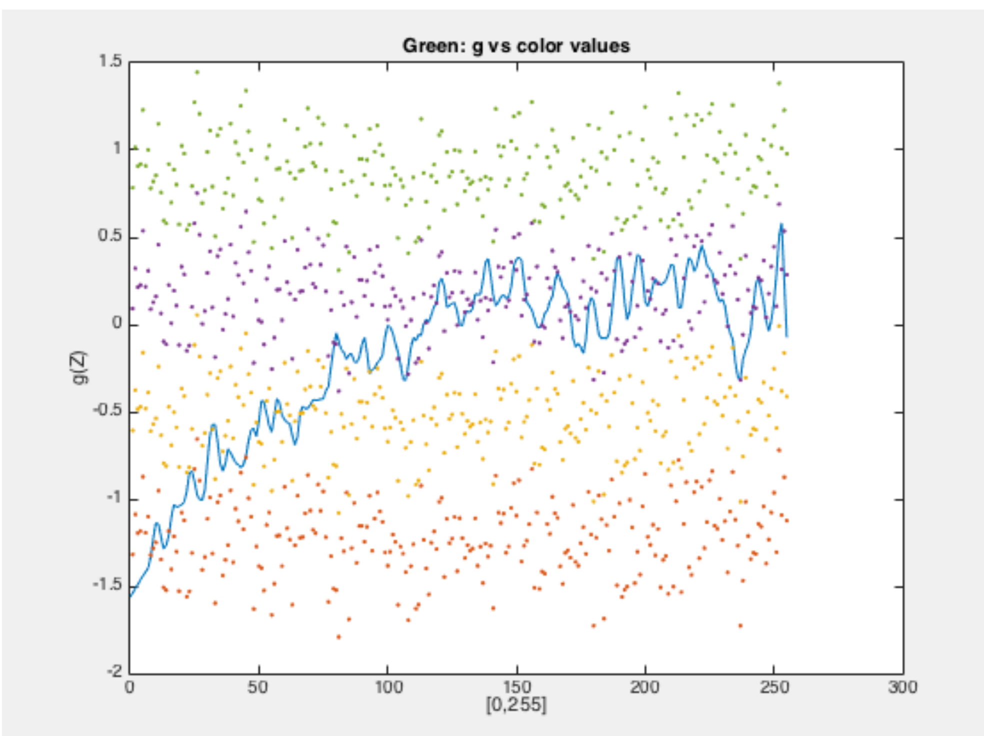
The regularization parameter l affects how noise in the graph. By varying the values of l, I noticed that as the l value increases, the graph becomes smoother.

The graphs are as follows:

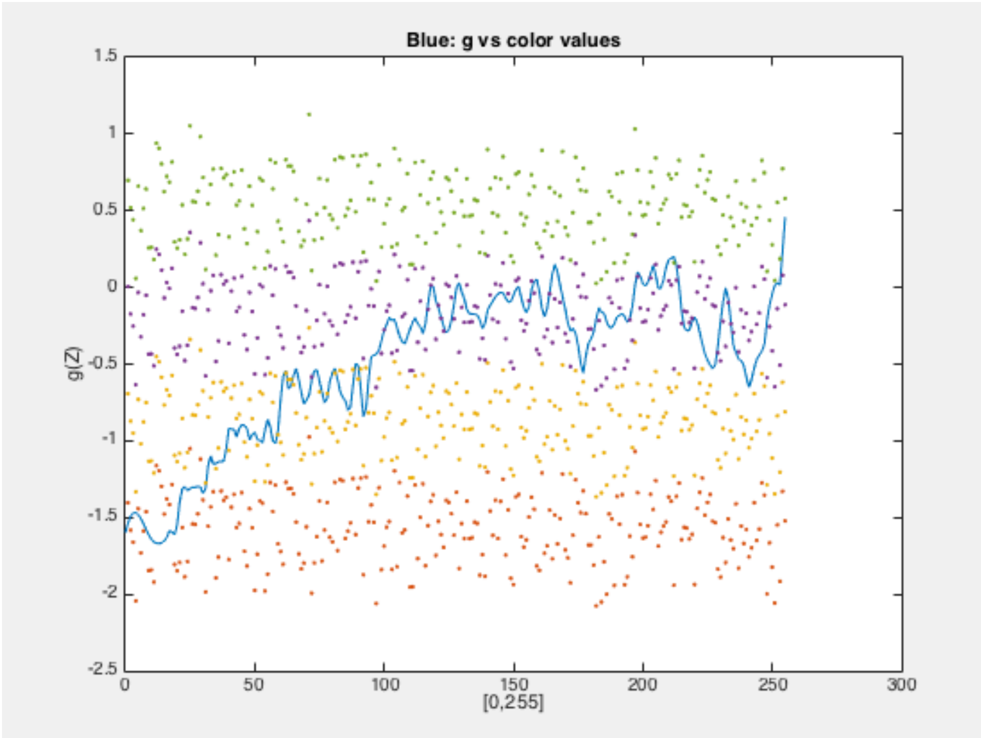
Camera Response Curve (Red):



Camera Response Curve (Green):



Camera Response Curve (Blue):



HDR radiance map of the scene:

The radiance map for each color channel was recovered using the equation

$$\ln(E[i]) = \frac{1}{P}\sum_{j=1}^{P}(g(Z[i,j]) - \ln(B[j]))$$

where p is the number of images captured.

The following Matlab code was used to recover the radiance map:

```
for p = 1:256
    for k=1:4
        index = find(I(:, :, 1, k) == p);
        radiance = g_r(p) + log(eB(k));
        radiance = radiance/4;
        outIndex_r(index) = outIndex_r(index) + real(exp((radiance)));

        index = find(I(:, :, 2, k) == p);
        radiance = g_g(p) + log(eB(k));
        radiance = radiance/4;
        outIndex_g(index) = outIndex_g(index) + real(exp((radiance)));

        index = find(I(:, :, 3, k) == p);
        radiance = g_b(p) + log(eB(k));
        radiance = radiance/4;
        outIndex_b(index) = outIndex_b(index) + real(exp((radiance)));
```

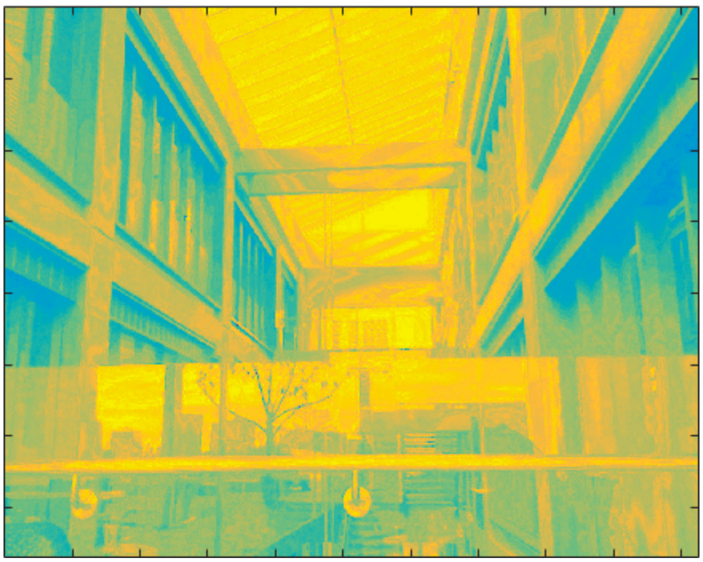
```
    end
end

outIndex = cat(3, cat(3, outIndex_r, outIndex_g), outIndex_b);
```

Dynamic Range:

Using the following code, I was able to get the value of the dynamic range of the scene as 3.3354.

```
max(exp(outIndex_r(:)))-min(exp(outIndex_r(:))) /mean2(exp(outIndex_r))
```

The plot of the radiance image recovered from the AEB sequence is as follows:



Tone Mapping to Display HDR Images:

The next step was to apply global tone-mapping algorithm to the radiance image to visualize the scene.

First, I scaled the brightness of each pixel uniformly so that all of the pixels fell in the range [0,1].

$$E_{norm}[i] = \frac{E[i]-E_{min}}{E_{max}-E_{min}}$$

where Emax and Emin are the maximum and minimum pixel values taken across all color channels. The following code did the computation:

```
for i=1:rows
    for j=1:cols
        normE_r(i,j) = (outIndex_r(i,j)-minE)/(maxE-minE);
```

```
        normE_g(i,j) = (outIndex_g(i,j)-minE)/(maxE-minE);
        normE_b(i,j) = (outIndex_b(i,j)-minE)/(maxE-minE);
    end
end
```

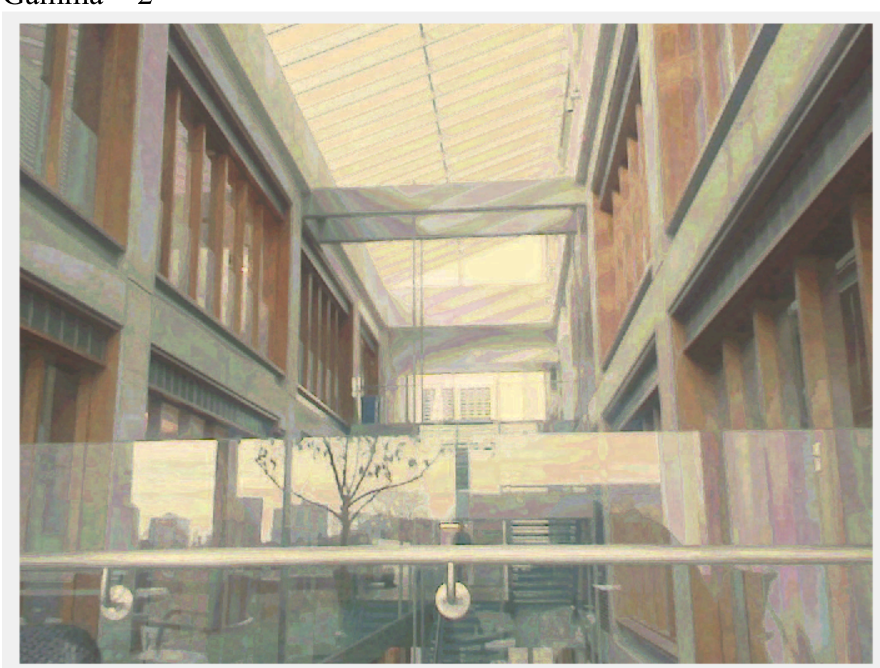
Next, a gmma curve was applied to the images by raising the irradiance of each pixel by gamma.

$$E_{gamma}[i] = E_{norm}^{\gamma}[i]$$

```
for i=1:rows
    for j=1:cols
        gamma_r(i,j) = (normE_r(i,j))^g;
        gamma_g(i,j) = (normE_g(i,j))^g;
        gamma_b(i,j) = (normE_b(i,j))^g;
    end
end
```

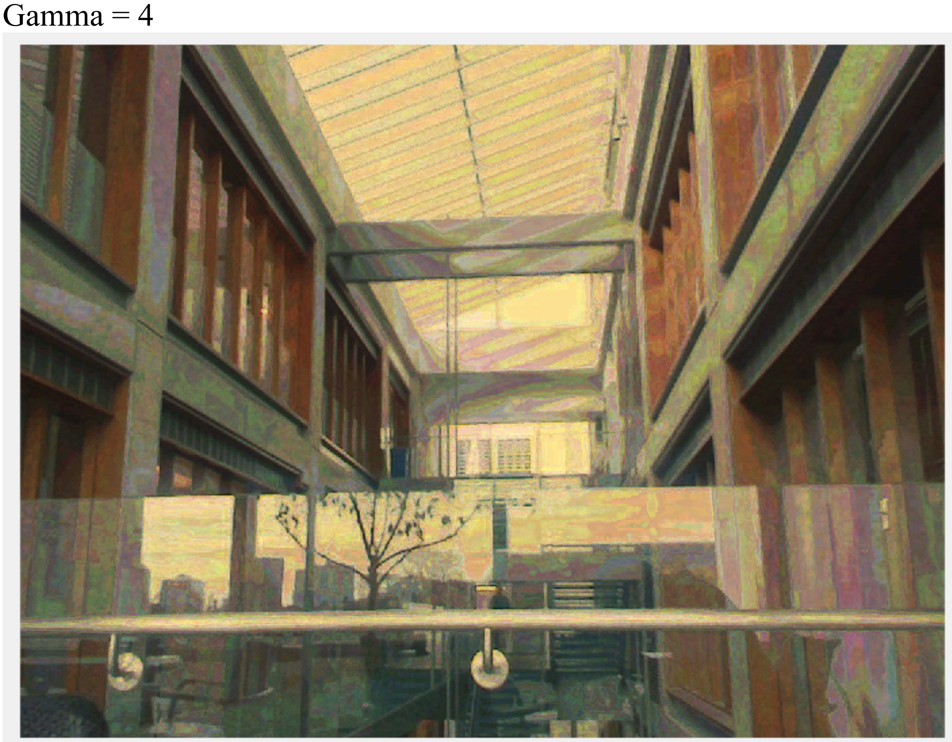
The following are results for different values of Gamma:

Gamma = 2



Gamma = 3

Gamma = 4



Gamma = 5

Among different values of Gamma, I liked Gamma = 4.

After converting the radiance image from color to grayscale, tone mapping was implemented using steps of computation. First the following equation was used to calculate the average log exposure:

$$L_{avg} = \exp\left(\frac{1}{N}\sum_i \ln\left(L[i]\right)\right)$$

```
lAvg=0;

for i=1:rows
    for j=1:cols
        lAvg = lAvg + log(l(i,j));
    end
end

lAvg = lAvg/(rows*cols);

lAvg = exp(lAvg);
```

Next, the image was scaled according to

$$T[i] = {}^{a}/_{L_{avg}}\, L[i]$$

```
a = .5;

for i=1:rows
    for j=1:cols
        t(i,j) = (a/lAvg)*(l(i,j));
    end
end
```

Next the Reinhard tone-mapping operator was applied according to the equation

$$L_{tone}[i] = \frac{T[i]\left(1+{}^{T[i]}/_{T_{max}^2}\right)}{1+T[i]}$$

```
for i=1:rows
    for j=1:cols
        lTone(i,j) = (t(i,j)*(1+(t(i,j)/(tmax^2))))/(t(i,j)+1);
    end
end
```

And finally, I descaled the image by the operation

$$M[i] = {}^{L_{tone}[i]}/_{L[i]}$$

```
M = zeros(1280,1024);

for i=1:rows
    for j=1:cols
        M(i,j) = lTone(i,j)/l(i,j);
    end
end
```

Using this we will scale eacj of the color channels in the radiance image according to:

$$R_{new}[i] = M[i] \cdot R[i],$$
$$G_{new}[i] = M[i] \cdot G[i],$$
$$B_{new}[i] = M[i] \cdot B[i]$$

```
for i=1:rows
    for j=1:cols
        Rnew(i,j) = M(i,j) * normE(i,j,1);
        Gnew(i,j) = M(i,j) * normE(i,j,2);
        Bnew(i,j) = M(i,j) * normE(i,j,3);
    end
end

RGBImage = cat(3, cat(3, Rnew, Gnew), Bnew);

imshow(RGBImage)
```

Finally, by fusing the three channels, we will get an image with dynamic ranging of the exposure.

The following image was obtained: