Gustave Le Gray arguably took the first high dynamic range images in the 1850s. *Brig in the moonlight* (retargeted above) was created via a manual process using seperate negatives for the sky and sea.

Project 5: High Dynamic Range

Due Date: Monday, November 12th, 11:59pm

Brief

- This handout: `/course/cs129/asgn/proj5/handout/`
- Stencil code: `/course/cs129/asgn/proj5/stencil/`
- Data: `/course/cs129/asgn/proj5/stencil/` (or take some pictures!)
- Handin: `cs129_handin proj5`
- Required files: README, code/, html/, html/index.html

Background

Modern cameras are unable to capture the full dynamic range of commonly encountered real-world scenes. In some scenes, even the best possible photograph will be partially under or over-exposed. Researchers and photographers commonly get around this limitation by combining information from multiple exposures of the same scene. You will write software to automatically combine multiple exposures into a single high dynamic range radiance map, and then convert this radiance map to an image suitable for display through tone mapping.

Requirements

There are two major components to this project:

1. Recovering a radiance map from a collection of images
2. Converting this radiance map into a display image

For the first part you will have to recover the inverse of the function mapping exposure to pixel value, g.

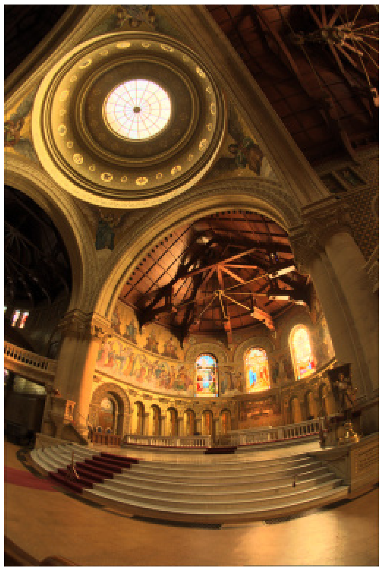
The the second part you will start by implementing your favourite global tone mapping operator, and you then have to improve it to a local tone mapping procedure.

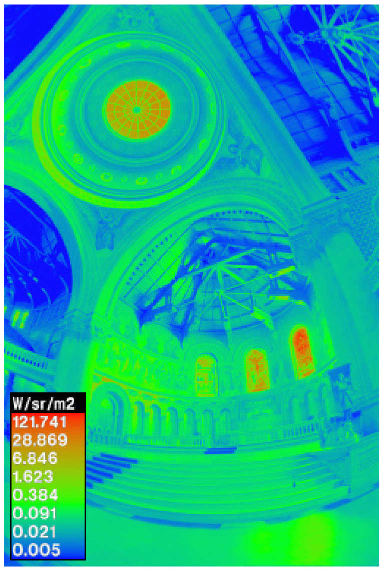
The two relevant papers are **Debevec and Malik 1997** and **Durand 2002**.

Details

Radiance map construction

Images in the data/ directory are labeled with their sequence number followed by their relative exposure (which happens to be milliseconds or microseconds).





We want to build an HDR radiance map from several LDR exposures. It is highly recommended that you read Sections 2.1 and 2.2 in **Debevec and Malik 1997** to help understand this process. Below is a summary.

The observed pixel value Z_{ij} for pixel i in image j is a function of unknown scene radiance and known exposure duration: $Z_{ij} = f(E_i \Delta t_j)$. E_i is the unknown scene *radiance* at pixel i, and scene radiance integrated over some time $E_i \Delta t_j$ is the *exposure* at a given pixel. In general, f might be a somewhat complicated pixel response curve. We will not solve for f, but for $g=\ln(f^{-1})$ which maps from pixel values (from 0 to 255) to the log of exposure values: $g(Z_{ij}) = \ln(E_i) + \ln(t_j)$ (equation 2 in Debevec). Solving for g might seem impossible (and indeed, we only recover g up to a scale factor) because we know neither g or E_i. The key observation is that the scene is static, and while we might not know the absolute value of E_i at each pixel i, we do know that the value remains constant across the image sequence.

Solving for g is the tricky part. After we have g, is it straightforward to map from the observed pixels values and shutter times to radiance by the following equation (rearranged from above): $\ln(E_i) = g(Z_{ij})-\ln(\Delta t_j)$. This is Equation 5 in **Debevec**.

To make the results robust, we want to consider two additional things:

- We expect g to be smooth. Debevec adds a constraint to our linear system which penalizes g according to the magnitude of its second derivative. Since g is discrete (defined only at integer values from g(0) to g(255) we can approximate the second derivative with finite differences, e.g. g''(x) = (g(x-1) - g(x)) - (g(x) - g(x+1)) = g(x-1) + g(x+1) - 2*g(x). We will have one such equation for each integer in the domain of g, except for g(0) and g(255) where the second derivative would be undefined.
- Each exposure only gives us trustworthy information about certain pixels (i.e. the well exposed pixels for that image). For dark pixels the relative contribution of noise is high and for bright pixels the sensor may have been saturated. To make our estimates of E_i more accurate we need to weight the contribution of each pixel according to Equation 6 in Debevec. An example of a weighting function w is a triangle function that peaks at Z=127.5, and is zero at Z=0 and Z=255. This is provided for you in the starter code. This weighting should be used both when solving for g and when using g to build the HDR radiance map for all pixels.

Tone mapping

Getting the radiance map is only half the battle. You want to be able to show off your image clearly. There are a few gobal tone-mapping operators to play with, such as log(L), sqrt(L), and L / (1+L). Regardless of which transform you use, you'll want to stretch the intensity values in the resulting image to fill the [0 255] range for maximum contrast.

Once you see something reasonable using a global tone mapping operation, you'll need to implement a local method for full credit.

You'll be implementing a simplified version of **Durand 2002**. The steps are roughly as follows:

1. Your input is linear RGB values of radiance.
2. Compute the intensity (I) by averaging the color channels.
3. Compute the chrominance channels: (R/I, G/I, B/I)
4. Compute the log intensity: L = log2(I)
5. Filter that with a 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
    1. The offset is such that the maximum intensity of the base is 1. Since the values are in the log domain, o = max(B).
    2. The scale is set so that the output base has dR stops of dynamic range, i.e., s = dR / (max(B) - min(B)). Try values between 2 and 8 for dR, that should cover an interesting range. Values around 4 or 5 should look fine.

8. Reconstruct the log intensity: O = 2^(B' + D)
9. Put back the colors: R',G',B' = O * (R/I, G/I, B/I)
10. Apply gamma compression. Without gamma compression the result will look too dark. Values around 0.5 should look fine (e.g. result.^0.5). You can also apply the simple global intensity scaling to your final output.

Your HDR images have zero values which will cause problems with log. You can fix this problem by replacing all zeros by some factor times the smallest non-zero values.

Implementation Tips

- To create results which are clearly better than any single exposure, make sure your scene actually has a high dynamic range! The data we give you have combinations of very bright (sunlight or bright lights) and very dark (shadowed) elements. If you take your own photos, take photos that have combinations of indoor and outdoor elements, or scenes that are heavily back-lit, possibly with the light source directly visible.
- For your own photos, and for any photos with long exposures, geometric alignment to correct for camera motion will probably improve your results.
- The brute force implementation of a bilateral filter is a little slow (couple of minutes for the test images). You can do any optimizations for extra credit.
- Matlab code for recovering g is available in Debevec and Malik 1997 and the lecture slides and we covered it in class. You can of course look at it and even test it as a reference. However, the code you turn in must be your own.
- The bilateral filter is sensitive to the sigma values for the intensity Gaussian and spatial Gaussian.
- The spatial component of the bilateral filter should, ideally, have a large spatial extent but that will make a naive implementation quite slow.

Extra Credit

- up to 10 points: Implement the bilateral filtering optimization discussed in class, in which Gaussian convolutions are performed in a subsampled 3d space. Discuss the speedup achieved over the baseline method.
- up to 3 points: Try the algorithm on your own photos!
- up to 10 points: Implement any other local tone mapping algorithm. For example a "simplified version" of **this** algorithm.
- Some of the test image stacks are not well aligned (e.g. garden). Try to automatically correct the geometric alignment of the images before running your HDR pipeline.
  - up to 3 points: For implementations similar to Project 1.
  - up to 5 points: For local alignment methods.

For all extra credit, be sure to demonstrate on your web page cases where your extra credit has improved image quality.

Graduate Credit

To get graduate credit for this project you must do at least 10 points of extra credit.

Write up

Describe your algorithm and any decisions you made to write your algorithm a particular way.

It is suggested that you show the following visualizations, for one or multiple test cases:

- Visualize your radiance maps, e.g. `imagesc(log(radiance_map + 1))`.
- Plot the recovered relationship between exposure and pixel values.
- Show the bilateral filtering based decomposition into detail and large scale structure.
- Show tonemapping results for all test cases, ideally compared against some baseline such as linear dynamic range compression or a simple global operator.

Finally, discuss any extra credit you did.

Handing in

This is very important as you will lose points if you do not follow instructions. Every time after the first that you do not follow instructions, you will lose 5 points. The folder you hand in must contain the following:

- README - text file containing anything about the project that you want to tell the TAs
- code/ - directory containing all your code for this assignment
- html/ - directory containing all your html report for this assignment (including images)
- html/index.html - home page for your results

Then run: `cs129_handin proj5`
If it is not in your path, you can run it directly: `/course/cs129/bin/cs129_handin proj5`

Rubric

- +35 pts: Recovering HDR from images at various exposures
- +10 pts: Results with global tone mapping
- +20 pts: Bilateral filter
- +20 pts: Local tone mapping
- +15 pts: Write up
- -5*n pts: Lose 5 points for every time (after the first) you do not follow the instructions for the hand in format

Credits

Project partially based on Noah Snavely's Computer Vision course at Cornell University. Handout written by David Dufresne, Travis Webb, Sam Birch, Emanuel Zgraggen, and James Hays.

Outline of Durand 2002 provided by **Sylvain Paris**.

Some of the test images have been taken from:
http://www.cs.brown.edu/courses/csci1950-g/results/final/dkendall/
http://www.pauldebevec.com/Research/HDR/
http://cybertron.cg.tu-berlin.de/eitz/