2017 Spring Digital Visual Effects

Project #1 — High Dynamic Range Imaging

Team Member: b03901032 Jason Kuo 郭子生, b02901001 S. L. Chung 鍾勝隆

Introduction:

High Dynamic Range (HDR) images have large dynamic ranges which correspond to irradiance value of a scene in physical world. This project implemented a program to assemble an HDR image from a list of images with identical scene but with different exposure times.

Submission Files:

- **Resource**
  ./input_image/: images with different exposure times and an image list
  ./result/: an output HDR image in .hdr format assembled from input images
  ./src/: Matlab script codes
  ./report.pdf
  ./readme.txt
- **Artifact**
  ./artifact.bmp

Usage:

1. Skip this step if you already have files listing above; otherwise, clone from here
2. Put images with different exposure times under directory input_image/
3. Under directory input_image/, create a .txt file named image_list in the following format:
   - line 1: an integer indicating number of images
   - line 2: <image filename> <space> <shutter speed of the image>
     …
   - line N: ...
4. Open main.m under directory src/ with Matlab, there are several settings to adjust according to your preference. To run default settings, simply run the code without changing anything.
   Default Settings:
   - Image Alignment: Not applying.
   - HDR Model: Debevec's method, more details down below.
   - Tone Mapping: Reinhard's method, more details down below.
5. The output HDR image in .hdr format, and the LDR image after tone mapping in .bmp format, are both under directory result/.

Implementation:

1. **Image Alignment**

   Image registration is an important step to be done before actual applying HDR model. It is because HDR models require each pixel being registered along images with the varying exposure times. Although with tripod, pixels may still not being well registered in some cases, and this is the reason why image alignment is needed.

   We implemented a method based on **Ward's Median Threshold Bitmap** (MTB) algorithm[1]. The algorithm first convert 3 channels colorful image into grey scale, then find the median value among pixels of the image. For those pixels with value larger than median, set it to 1; otherwise, set it to 0, so that median threshold bitmap of the image is generated. Next, build an MTB pyramid and starting from the one with lowest resolution. With a pair of MTBs, set one as reference, and shift the other one in nine direction: (-1,-1), (-1,0), (-1,1), (0,-1), (0,0), (0,1), (1,-1), (1,0), (1,1). For each direction, apply bitwise XOR to compute difference between these MTBs. Find the direction with less difference, add the direction vector to total translation vector (which is 0 at the beginning), double the value, and move on to the next layer (do not double the value at the last layer of the pyramid).

Not to mention that in order to improve the result of MTP even more, one can set a threshold to generate exclusion bitmap from the image to avoid noises. The idea of exclusion map is that for those pixels with value close to median, it may become 0 or 1 on bitmap depending on noises, and these may further cause great influences when calculating the difference between bitmaps. Therefore, the exclusion map is generated by setting all those pixels with value close to median within a threshold to 0, and the others to 1. After XOR two MTBs, apply bitwise AND to compute the final difference, so as to stabilize the result from noises.

For more details, please take a look at these functions under directory src/: MTB.m, ComputeBitmaps.m, and BuildPyramid.m. All parameters are well commented in source codes. Also, the original paper is listed in reference section down below.

2. HDR

To assemble HDR images from images with varying exposure times, there are numerous ways to accomplish the task. In this project, we implemented two method which take a list of images as input, and with an HDR image as output.

Paul Debevec's method:

Debevec's method[2] is a classic approach to assemble HDR images. The basic idea of this method is to recover the response curve by information among input image series. Given exposure times and pixel values of these images, as well as some condition such as weight function, regularization constant, and a reference point, Debevec's method are able to recover response curve by solving the linear system with SVD. For more details, please take a look at these function under directory src/: DebevecGsolve.m, DebevecHDR.m. All parameters are well commented in source codes. Also, the original paper is listed in reference section down below.

Robertson's method:

Robertson's method[3] is another popular approach to assemble HDR images. The basic idea of this method is to find the function interval which maps real numbers to 0 - 255. In order to find the interval, Robertson's method has its self-designed condition such as weight function. Simply iterate over the equation until convergence, the HDR image is generated. For more details, please take a look at these function under directory src/: RobertsonHDR.m. Also, the original paper is listed in reference section down below.

3. Tone Mapping

Tone mapping is a method to convert HDR images back into LDR image, but keeps both details in bright and dark regions of a scene. We implemented two methods in Reinhard's SIGGRAPH 2002 paper[4], which is global operator and local operator respectively.

Global Version:

Tone mapping takes the luminance map and rescales it by the average logarithmic luminance. Afterwards, user can define the scale to desired brightness level by adjusting the key value 'a'. With the following equation, we can convert the HDR image into a LDR image.

$$L_d(x,y) = \frac{L(x,y)\left(1+\frac{L(x,y)}{L_{\text{white}}^2}\right)}{1+L(x,y)}$$

Local Version:

Tone mapping mimic how the photo were printed in the past — dodging and burning. The luminance of a dark pixel in a relatively bright region will decrease the display luminance, thereby increasing the contrast at that pixel. We can find that different from the global version, the contrast in local version of tone mapping is a bit stronger.

Finally according to Fattal's SIGGRAPH 2002 paper[5], we re-apply the color of image back and are able to adjust the saturation of the picture with input argument of the function. For more details, please take a look at these function under directory src/: tonemapping_global.m, tonemapping_local.m. Inside these functions, s is for the saturation, C is for the color channel, Lin is the HDR luminance map, and Lout is the LDR luminance map. Also, the original paper is listed in reference section down below.

Experiments:

1. Input images

Figure 1. memorial input image series

Figure 2. sea input image series

(f 3.5, iso 400, 18mm, shutter speeds: 1/6400, 1/3200, 1/1600, 1/800, 1/400, 1/200, 1/100, 1/50)

Figure 3. plant input image series

(f 10, iso 400, 32mm, shutter speed: 1/20, 1/40, 1/80, 1/160, 1/320, 1/640, 1/1250)

2. Image Alignment

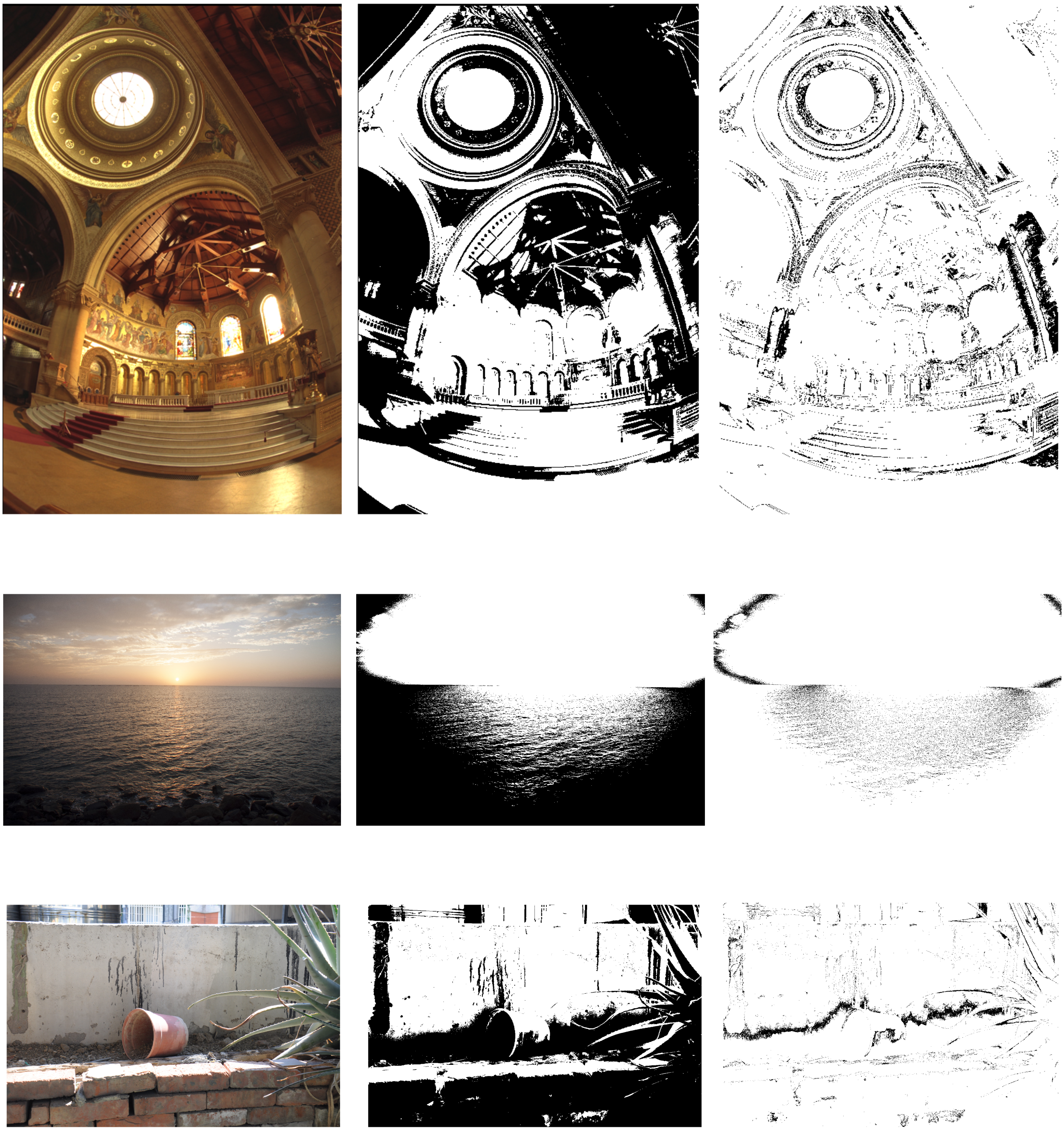

Figure 4. From left to right: original image, median threshold bitmap, exclusion bitmap. The purpose of median threshold bitmap is for alignment, and the purpose of exclusion bitmap is for stability from noise.

3. HDR

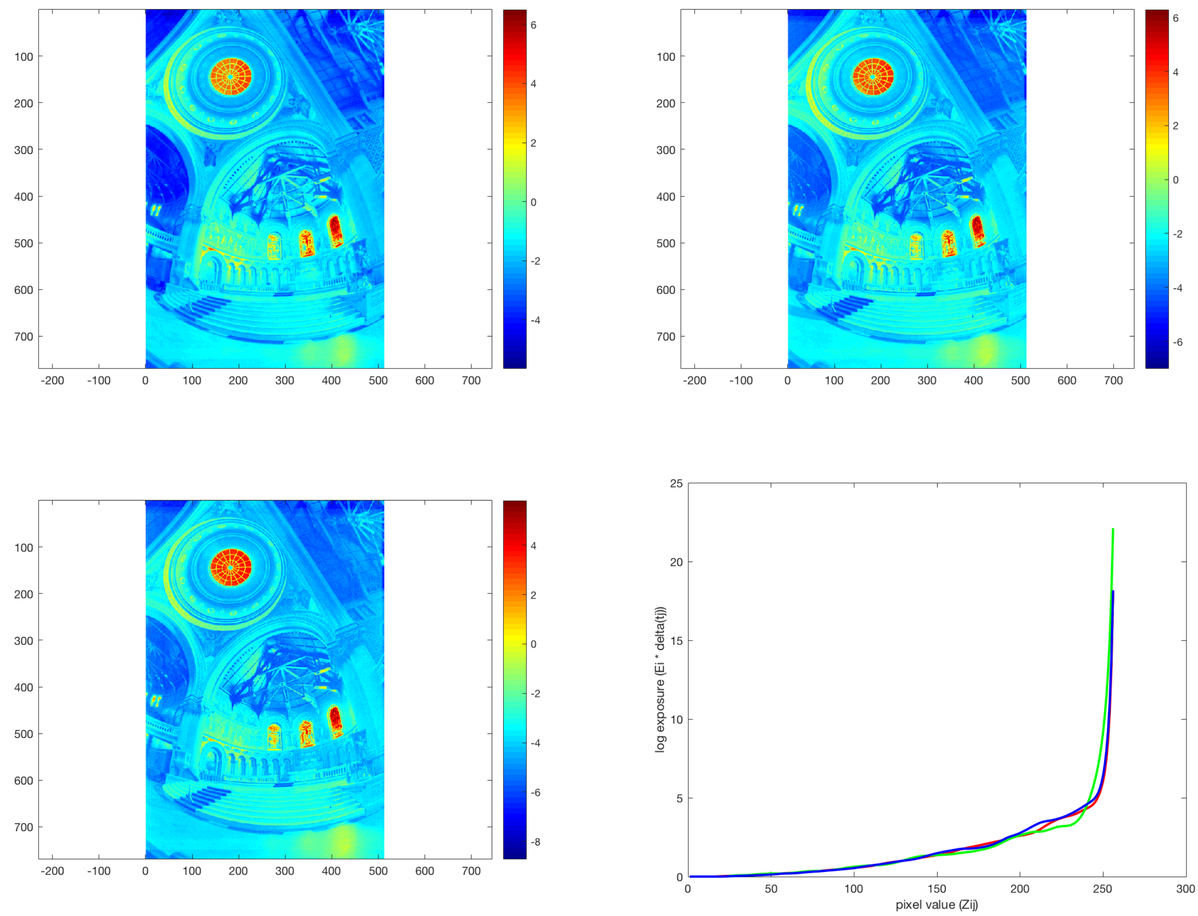

Figure 5. Irradiance map of RGB channels respectively and function $g(Z_{ij})$ based on Debevec's Method with lambda set to 10.

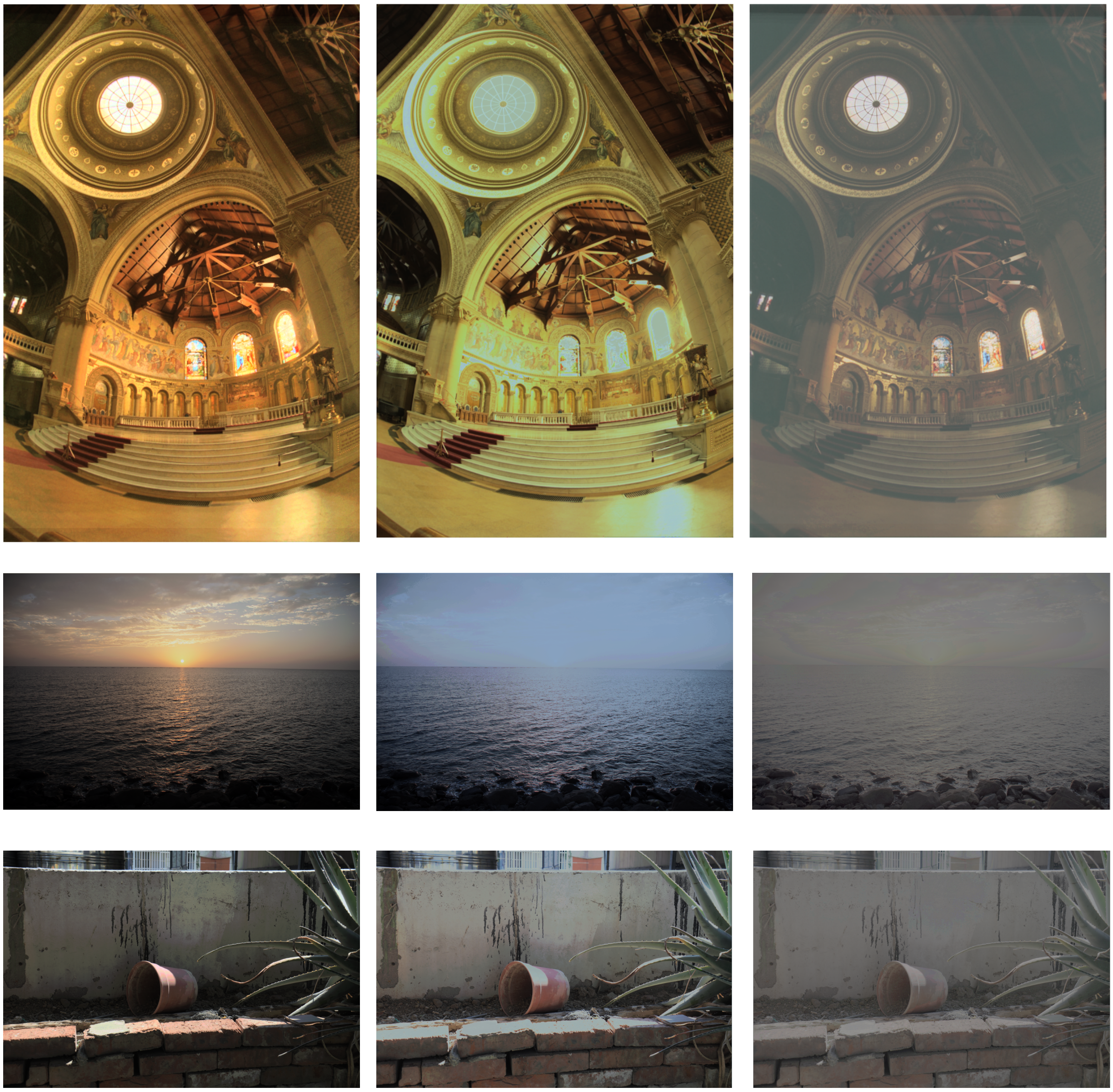

Figure 6. From left to right: Paul Debevec's method, Robertson's method, Matlab's default method. In Paul Debevec's method, constant lambda is set to 5. In Robertson's method, iteration number is set to 5. All three HDR images were converted to LDR image with Reinhard's global version tone mapping with key value 'a' set to 0.72 as well as Fattal's saturation parameter set to 0.5.

Figure 7. All by Debevec's Method but with varying constant lambda. From left to right: lambda is set to 1, 5, 10, 20. It seems that as lambda is set to 10, the image has better color and sharper.

4. Tone Mapping

Figure 8. Reinhard's global version tone mapping with varying key value 'a'. From left to right: 'a' is set to 0.5, 0.72, 1.0, 1.5, 2.0. All by Debevec's Method with lambda set to 10, and by Reinhard's global version tone mapping with saturation set to 0.5.

Figure 9. Reinhard's global version tone mapping with varying saturation value. From left to right: saturation value is set to 0.34, 0.4, 0.5, 0.6, 0.65. All by Debevec's Method with lambda set to 10, and by Reinhard's global version tone mapping with key value 'a' set to 1.0.

Figure 10. Three method for tone mapping. From left to right: Reinhard's global version, Reinhard's local version, Matlab's default version. It is obvious that the global version works well overall; the local version focus much more on details; and the Matlab default version is robust but colorless.

Figure 11. Submission Artifact.

Assembled from sea image series. By applying Debevec's method with lambda set to 10, and by applying Reinhard's global version tone mapping with key value set to 2.5 and saturation set to 0.6.

Reference:

[1] Greg Ward, Fast Robust Image Registration for Compositing High Dynamic Range Photographs from Hand-Held Exposures, jgt 2003.

[2] Paul E. Debevec, Jitendra Malik, Recovering High Dynamic Range Radiance Maps from Photographs, SIGGRAPH 1997.

[3] Mark Robertson, Sean Borman, Robert Stevenson, Estimation-Theoretic Approach to Dynamic Range Enhancement using Multiple Exposures, Journal of Electronic Imaging 2003.

[4] Erik Reinhard, Michael Stark, Peter Shirley, Jim Ferwerda, Photographics Tone Reproduction for Digital Images, SIGGRAPH 2002.

[5] Raanan Fattal, Dani Lischinski, Michael Werman, Gradient Domain High Dynamic Range Compression, SIGGRAPH 2002.