NTU-VFX-Project1 Report

Team Member

- R10922022 曾筱晴
- B09902028 曾翊綺

Description

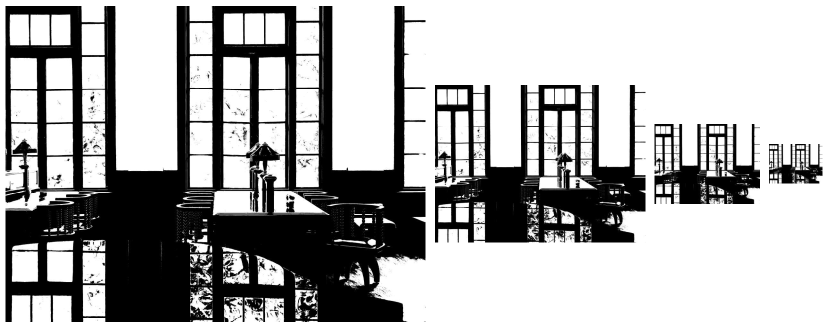
In this project, we have to recover HDR images from original images of a scene under different exposures.

Implementation

- **Image Alignment**
  - MTB Algorithm
- **HDR Reconstruction**
  - Paul Debevec's method
  - Robertson's method
- **Tone Mapping**
  - Global Operator
  - Local Operator
  - Other OpenCV tone mapping (e.g., Drago, Mantiuk)

Step 1. Image Alignment

1. We use the pyramid method with recursion (downsampling by a factor of 2) to implement alignment.



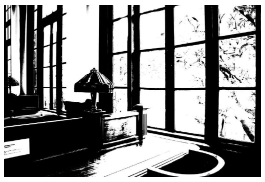
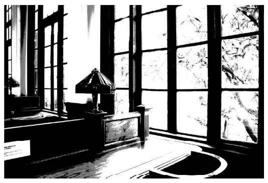
2. Choose the image with **median exposure time** as the base image, and we will then align the other images with the base image.

   > We choose the one with median exposure time because the exclusion bitmaps of the images with extremely low or high exposure can be all 0 or 255, which may cause unexpected result afterwards.

3. For well calculating the difference of two images, we first turn each image into grayscale versions, get their threshold images considering median exposure.

4. Shift the image with offset $(dx, dy) = \{(di, dj) | -1 \leq di, dj \leq 1\}$. For calculating the difference between two images, noted that there might be threshold noise, we introduce exclusion bitmaps here by ignoring pixels (±4) that are near to the threshold.
   Finally, For each layer, select the offset with minimum difference.

   ```
   Taking difference:
   1. XOR two images
   2. AND with their exclusion bitmaps respectively
   ```

Grayscale Image	**Threshold Bitmap**	**Exclusion Bitmap**
		

5. At the next layer, multiply this offset by 2. Repeat the process and get the aligned result of each image.

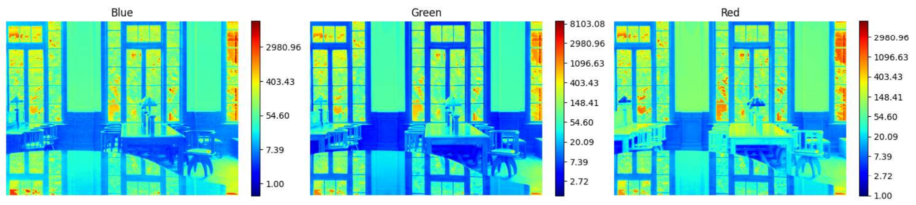
Step 2. HDR Reconstruction

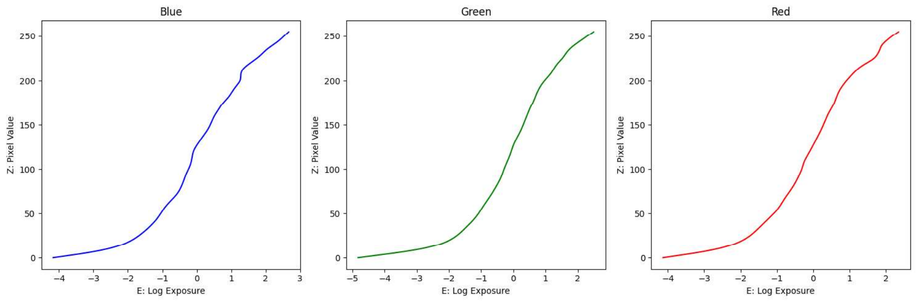
Paul Debevec's method

1. Construct A & b matrix. Solve $Ax = b$ to get least-square solution x, and thus get g curve.

2. Use the below equation to construct radiance map E.

$$lnE_i = \frac{\sum_{j=1}^{P} w(Z_{ij})(g(Z_{ij}) - ln\Delta t_j)}{\sum_{j=1}^{P} w(Z_{ij})}$$





Robertson's method

1. Initialize g curve.

$$g(m) = \frac{m}{128}$$

2. Optimize irradiance E based on the equation below.

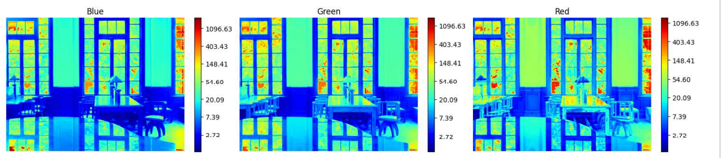
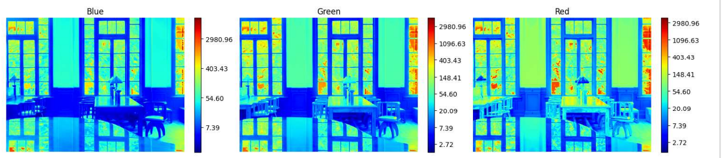
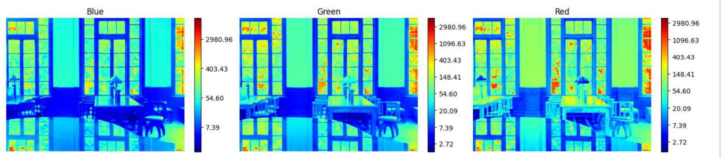
$$E_i = \frac{\sum_j w(Z_{ij}) g(Z_{ij}) \Delta t_j}{\sum_j w(Z_{ij}) \Delta t_j^2}$$

3. Optimize g curve based on the equation below.

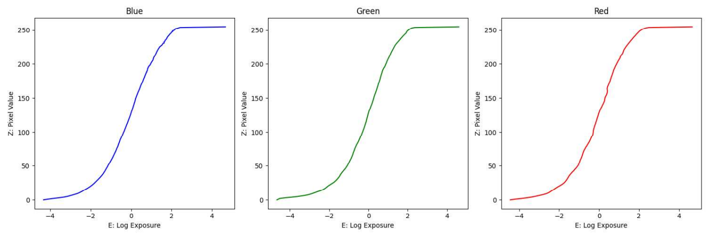
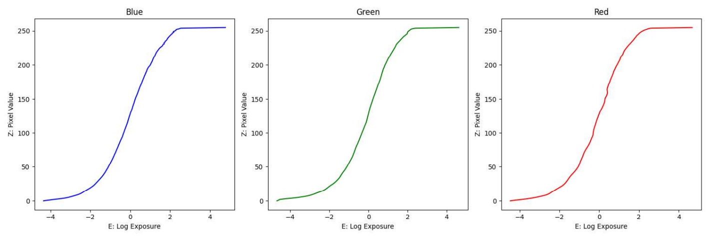
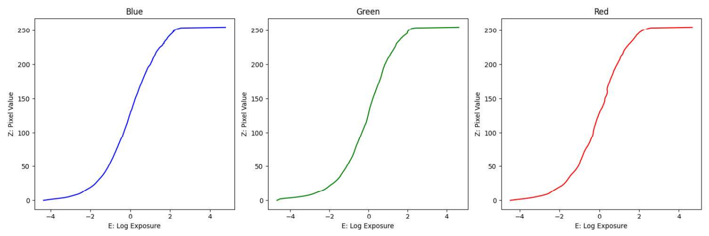
$$g(m) = \frac{1}{|E_m|} \sum_{ij \in E_m} E_i \Delta t_j$$

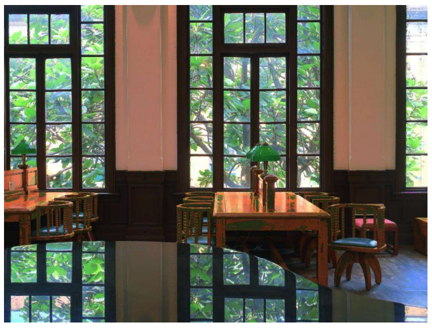
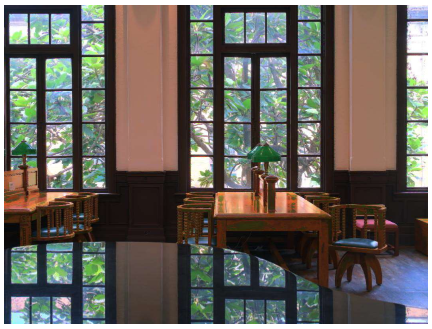
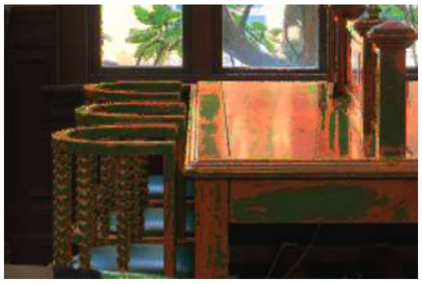
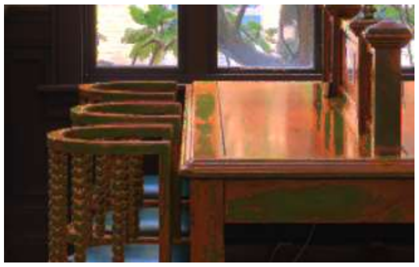
4. Back to Step 2 and repeat.

The changes from the first iteration to the 3-th iteration are more obvious than third iteration to fifth iteration. After the third iteration, although the radiance map was only slightly different, we can see the results is getting better and better.

1-th iteration	
3-th iteration	
5-th iteration	

The change of response curve is not obvious.

1-th iteration	
3-th iteration	
5-th iteration	

1-th iteration	5-th iteration
	
	

Step 3. Tone Mapping

Global Operator

1. Calculate HDR's average luminance in log domain, taking exponential to get $\bar{L}_w$.
2. Normailize L_w to get L_m (median). Here, a is user-defined, we set it to 0.5.

   $L_m(x,y) = \frac{a}{\bar{L}_w} L_w(x,y)$
3. Calculate L_d, and multiply it by 255 to get final output. (set $L_{white} = 200$)

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

Local Operator

1. Calculate HDR's average luminance in log domain, taking exponential to get $\bar{L}_w$.
2. Normailize L_w to get L_m (median). Here, a is user-defined, we set it to 0.5.

   $L_m(x,y) = \frac{a}{\bar{L}_w} L_w(x,y)$

3. Try each s with odd numbers, get L_s^{blur} with the convolution of L_m and G_s.

   $$L_s^{blur}(x,y) = L_m(x,y) \otimes G_s(x,y)$$

4. Calculate each $V_s(x,y)$ with different s.

   $$V_s(x,y) = \frac{L_s^{blur}(x,y) - L_{s+1}^{blur}(x,y)}{2^\phi a/s^2 + L_s^{blur}(x,y)}$$

5. Find the s_{max} of all (x,y), and thus get $L_{s_{max}}^{blur}(x,y)$.

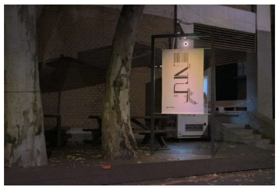
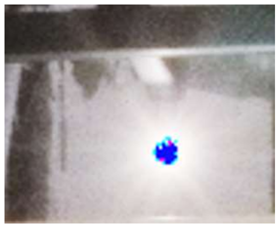
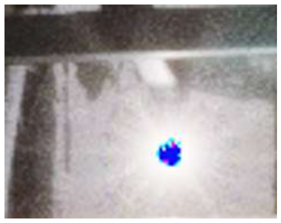
   s_{max} : $|V_{s_{max}}(x,y)| < \varepsilon$, here we have $\varepsilon = 0.01$

6. Calculate L_d based on the equation below, and multiply it by 255 to get final output.

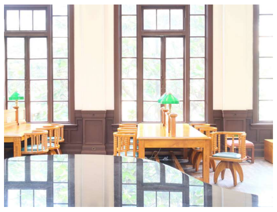
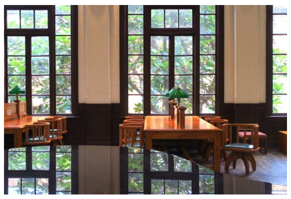
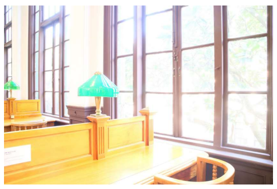
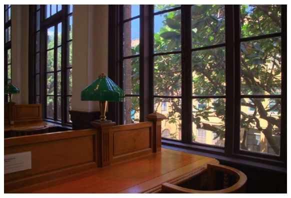
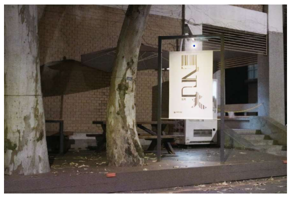
   $$L_d(x,y) = \frac{L_m(x,y)}{1 + L_{s_{max}}^{blur}(x,y)}$$

Other OpenCV tone mapping

e.g., Drago, Mantiuk

Drago	Global	Local
		
		

Some Results

Original Image1	Original Image2	Best Result
		
		
		

What we have learned

We've got a great sense of achievement through this project. During the work, we learned how to optimize our program and speed up running time (especially for the iteration of Robertson's algorithm).
Besides, there are some parameters that must be adjusted manually based on different photo shooting environment; therefore, being familiar with the environment we choosed is important as well.
The results have shown that there doesn't exist a best algorithm that are suitable for all series of images. We need to try through several methods so that we can get the most good-looking picture for each series.