Project 1: High Dynamic Range Imaging

Instructors: Yung-Yu Chuang

李俊賢 *(b03902029)*、劉彥廷 *(b03902028)*

1 Project Description

1. Taking Photograph: We borrow the Nikon D90 camera and tripod from TA, and use the NKRemote to get photos without touching. We can also precisely control shutter speed by the NKRemote in order to construct our HDR image conveniently.
2. Image Alignment By MTB algorithm [3]
3. HDR Image Construction by Debevec's algorithm [1]
4. Tone mapping By Bilateral Filtering We implement the Tone mapping by the Bilateral Filtering as professor mentioned in class. While the Bilateral Filtering [2] implementation is achieved by the `imbilatfilt()` function in MATLAB.

2 How to run our Code

2.1 Prerequisite

- MATLAB R2018a (We need R2018a version for using `imbilatfilt()`)
- Python 3.6
  - imageio
  - skimage
  - numpy

2.2 Alignment

```
python mtb.py --list mtb-input.txt --level 5
```

This will produce files named with `after_*.jpg`

2.3 HDR constructionn

```
python hdr.py --list hdr-input.txt --sample-point 50 --smooth-weight 10 --output output.hdr
```

Every line in `hdr-input.txt` contains two elements, filepath and shutter speed. They are separated with comma.

2.4 Tone mapping

Open MATLAB R2018a and executed tonemapping.m with the target path as the parameter

```
tonemapping('output.hdr', 'output.png');
```

3 Implementation Detail

3.1 Image Alignment

We implement MTB algorithm to align our images. We use first image as standard, and align all the other images with it one by one. Following is the algorithm,

1. We have two images, target image, which is the standard image, and source image, which is an image to be aligned.
2. Create image pyramids for them, and convert to grayscale images in every level
3. Use median value as threshold to convert them into binary map in every level

4. Use target image to generate a exclusive map by setting pixels with values close to median value to 0 and setting others to 1 in every level

5. We start at the highest level, which has the smallest images. Shift x and y of source image in a range of $[-1, 1]$, and do a pixel-wise XOR with target image to get the difference map and then do a pixel-wise AND to eliminate noise in difference map. We find the difference map which has the smallest 1 value, and set the offset as the optimal offset at this level. Then, we add this offset to our accumulated offset. Then, we do the same thing in next level, and multiply the accumulated offest by two before adding the optimal offset of next level. Repeat this process until the lowest level.

6. After we compute offset of other image, we then crop the overlap part of these images (include target image) to get a set of aligned images.

3.2 HDR Image Construction

We implement Debevec's Algorithm, which is recovering response curve g by minimizing a objective function

$$\mathcal{O} = \sum_{i}^{N} \sum_{j}^{P} \left(w(Z_{ij}) \left(g(Z_{ij}) - \ln E_i - \ln \delta_{t_j} \right) \right)^2 + \lambda \sum_{z=z_{min}+1}^{z_{max}-1} \left(w(z) g''(z) \right)$$

We can solve it by finding the least square solution in a linear system.

3.3 Tone mapping

We use Bilateral Filtering as we learned in class. And adjust the contrast factor and the normalized factor to get the beautiful picture. The algorithms are listed below step by step.

1. Get the input image and turn it into intensity by `Intensity = 0.299 * R + 0.587 * G + 0.114 * B`

2. Turn the intensity to log image for computing conveniently `Intensity_log = log10(Intensity)`

3. Apply `imbilatfilt()`, which is provided by MATLAB, to get the Low and High frequency image

   ```
   LowFrequency = imbilatfilt(Intensity_log)
   HighFrequency = Intensity_log - LowFrequency
   ```

4. Then set the Gamma constant, which is used to indicate the contrast level, to be

   ```
   Gamma = (log10(50)/log10(10)) / (max - min)
   ```

   We think we can get the nice picture by factor `log10(50)`

5. Reconstruct the new intensity image by LowFrequency, HighFrequency, and our Gamma

   ```
   newIntensity = LowFrequency * gamma + HighFrequency
   ```

6. Power the new intensity image by 10 to reconstruct the true color

   ```
   newIntensity = power(10, newIntensity)
   ```

7. Reconstruct the new color channel by the old and new intensity

   ```
   img = img .* newIntensity;
   img = img ./ oldIntensity;
   ```

8. Normalize the color

   ```
   img = (img - min(img(:))) / (max(img(:)) - min(img(:)))
   ```

4 What We learn

1. We should remember to normalized our image at the last step of tone mapping, otherwise we' ll get the all white image due to the overflow of the pixel value.

2. It is hard to do proper photography.

Figure 1: Our best result. One can checkout https://www.csie.ntu.edu.tw/~b03902028/vfx/hw1 for other results.

Reference

[1] P. E. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 369–378. ACM Press/Addison-Wesley Publishing Co., 1997.

[2] F. Durand and J. Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. In *ACM transactions on graphics (TOG)*, volume 21, pages 257–266. ACM, 2002.

[3] G. Ward. Fast, robust image registration for compositing high dynamic range photographs from hand-held exposures. *Journal of graphics tools*, 8(2):17–30, 2003.