

README.md

High Dynamic Range Imaging

This is a python project from recovering the response curve to reconstruct the irradiance map(HDR) in the real scene.

Author

r08944022 蔡仲閔 / vtsai01@cmlab.csie.ntu.edu.tw

Summary for TA

- to run the code.

`python3 computeHDR.py`

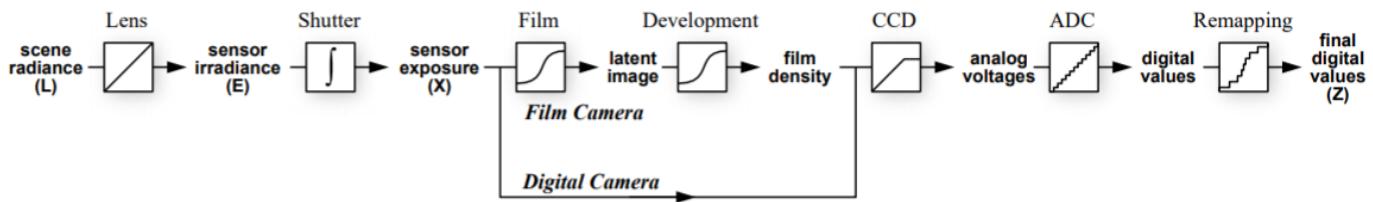
- this project is based on [high-dynamic-range-image](#)
- modified most of the function
- rewrite the `computeRadianceMap()` function in `./hdrtool/hdr.py` and reduce running time down to **3%**.
- implemented the tone mapping function.
- great photography^^

Outline

- Project Description
- Algorithms
- Usages
- Results

Project Description

There are several steps(shown below) between the scene radiance and the image we see. Starting from taking few photographs in different exposures, we use [Paul Debevec's method](#) to [recover the response curve](#) by these photographs. After we got the response curve, we are now able to [reconstruct the irradiance map](#) also known as HDR images. Also, we developed a [global tone mapping](#) algorithm to see the combination of these photographs.



Algorithms

Recovering the Response Curve

The function of response curve is a general term for the many steps shown above. To recover from pixel values record on image to radiance map we need a function g which is:

$$\mathcal{O} = \sum_{i=1}^N \sum_{j=1}^P \{w(Z_{ij}) [g(Z_{ij}) - \ln E_i - \ln \Delta t_j]\}^2 + \sum_{z=Z_{min}+1}^{Z_{max}-1} [w(z)g''(z)]^2$$

- g is the unknown response function
- w is a linear weighting function. g will be less smooth and will fit the data more poorly near extremes ($Z=0$ or $Z=255$).Debevec introduces a weighting function to emphasize the smoothness fitting terms toward the middle of the curve.
- t_j is the exposure time in index j
- E_i is the unknown radiance in different pixel i
- Z_{ij} : is the observed value in pixel i and j exposure time
- i is the pixel location index, j is the exposure index and P is the total number of exposures.

Reconstruct the Irradiance Map

And now we can reconstruct the irradiance map by the formula below.

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

After this step, we are able to have the `output.hdr` file.

Global Tone Mapping

In this part, we implemented a naive global tone mapping algorithm(shown below) to get a image which can match our visual experience.

$$L_d(x, y) = \frac{L_m(x, y)(1 + \frac{L_m(x, y)}{L_w^2(x, y)})}{1 + L_m(x, y)}$$

and now we can have the result.

Usages

There are python file and ipython notebook for you to choose.

Prepare Images and Meta Data

Put your images in a single folder and prepare your meta data file. The meta file should contains filename and exposure time separated with spaces(see `./example/park3.csv`). There are meta file creator in `createMeta.ipynb` if your images contains EXIF data.

Start

here is an example to run the code.

```
python3 computeHDR.py --img-dir [IMG_DIR] --meta_path [META_PATH_IN_CSV]
```

to see more parameters

```
python3 computeHDR.py --help
```

Results

Original image



Exposure 1/4 sec

Exposure 1 sec

Exposure 4 sec

Exposure 15 sec

HDR image



Acknowledgements

- [Digital Visual Effects](#)
- [high-dynamic-range-image](#)
- [HDR-Imaging](#)