Due: 19 Feb 2024 3:30 PM Total points: **100**

In this project you will use python to combine a stack of images captured with different exposures into a single high-dynamic-range image.

Your submission should include both your code and a pdf report describing your implementation and results. Details are included on the last page of this document.

Dependencies

You will need python modules to read the images and their meta data. It's easiest to install them with anaconda:

```
conda install -c conda-forge exifread opencv matplotlib
```

You will use the program dcraw to process the raw images you are provided. Dcraw for windows can be downloaded here: `https://www.easyhdr.com/download/dcraw/9.28/dcraw.exe`. Windows users: Either run "dcraw" from the command line where the .exe files is or copy the .exe file to C:\Windows to add it to your path. Linux installation should be easier. Dcraw may be depreciated. Libraw is an alternative.

1 Processing the Raw Images

10 points total

The provided exposure stack of images included in the assignment are all raw files which must be demosaiced and white balanced (white balancing could be done later instead) before use.

First use the dcraw program to extract the 13th exposure's auto-whitebalancing information using the -i -v arguments. We chose the 13th exposure as portions of this image were well-exposed and should provide reasonable white-balancing values. This provides a set of four multipliers which will scale the red, green1, blue, and green2 channels.

Next, convert all the raw NEF files into *linear* 16-bit TIFF images using the white balancing multipliers you just found (I suggest using a multiplier of 1 for both green channels). Be sure to use the same white balance settings for every image. You can process all images at once with a single command.

See dcraw documentation here: `https://manpages.ubuntu.com/manpages/bionic/man1/dcraw.1.html`.

2 Merge the LDR images into an HDR image

35 points total

You will now form a single HDR image by computing a weighted combination of all the LDR images. To do so, you will scale each image by the reciprocal of its exposure time (so they all have the same relative brightness) and then use the weighting function to average together only pixels that were properly exposed; don't average clipped nor underexposed and noisy pixels.

For every pixel (i, j) the HDR image is

$$I_{i,j,HDR} = \frac{\sum_k w(I^k_{i,j,LDR}) I^k_{i,j,LDR}/t^k}{\sum_k w(I^k_{i,j,LDR})},$$

where t^k denotes the exposure associated with the k^{th} image, *not* t to the power k.
You can extract exposure information using the provided "get_exposure" command[1].

Weighting. A number of different weighting schemes can be applied to merge the images.

$$w_{uniform}(z) = \begin{cases} 1, & \text{if } Z_{min} \leq z \leq Z_{max}, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

$$w_{tent}(z) = \begin{cases} \min(z, 1-z), & \text{if } Z_{min} \leq z \leq Z_{max}, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

$$w_{Gaussian}(z) = \begin{cases} \exp(-4\frac{(z-.5)^2}{.5^2}), & \text{if } Z_{min} \leq z \leq Z_{max}, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

$$w_{photon}(z, t^k) = \begin{cases} t^k, & \text{if } Z_{min} \leq z \leq Z_{max}, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Z_{min} and Z_{max} are tunable parameters which control the clipping thresholds. We recommend $Z_{min} = .05$ and $Z_{max} = .95$. To avoid NaNs, you can set Z_{min} to 0 when processing the longest exposure image.

Implement and apply all four weighting schemes to create four HDR images.

3 Tone-map the HDR image into an LDR image

35 points total

You now have four HDR images that needs to be tone-mapped for display. We will tone-map using the operator proposed by Reinhard et al. [2][2]:

[1]

[2] There is a typo in the equations found in the paper. The equations below are correct.

$$I_{i,j,TM} = \frac{\tilde{I}_{i,j,HDR}(1 + \frac{\tilde{I}_{i,j,HDR}}{\tilde{I}^2_{white}})}{1 + \tilde{I}_{i,j,HDR}}, \tag{5}$$

where

$$I_{m,HDR} = \exp\Big(\frac{1}{N}\sum_{i,j}\log(I_{i,j,HDR} + \epsilon)\Big), \tag{6}$$

$$\tilde{I}_{i,j,HDR} = \frac{K}{I_{m,HDR}} I_{i,j,HDR}, \tag{7}$$

$$\tilde{I}_{white} = B \max_{i,j}(\tilde{I}_{i,j,HDR}). \tag{8}$$

The parameter K is the *key*, which determines how bright or dark the resulting tonemapped image is. The parameter B is the *burn* and can be used to suppress contrast in the result. Finally, N is the number of pixels in the image and ϵ is a small constant to avoid log(0).
For your HDR image computed using $w_{Gaussian}$, sweep over various values for K and B (at least 3 each) and report how they change the results (be sure to save gamma-corrected images, see next section). Reasonable starting values are $K = 0.15$ and $B = 0.95$.
After you've chosen values for K and B that you like, tone-map all four HDR images.

4 Gamma-Correct the LDR image for display

10 points

The tone-mapped LDR images from the previous step still need to be gamma-corrected to present accurately on your display, otherwise they will appear quite dark.
Gamma correct the images you display and save with the following operator, which corresponds to the tone reproduction curve specified in the sRGB standard [1]:

$$C = \begin{cases} 12.92 C_{linear}, & C_{linear} \leq 0.0031308 \\ (1 + 0.055) C_{linear}^{\frac{1}{2.5}} - 0.055, & C_{linear} > 0.0031308. \end{cases} \tag{9}$$

Save your four tone-mapped and gamma-corrected images as losslessly compressed PNG images using the provided `writeHDR` command and include them in your report.

5 Lossy Compression

10 points

Use CV2's imwrite command to save one of your images in the compressed JPEG format. Note that cv2 expects images to be in bgr format with pixel values in the range 0 to 255. See the `writeHDR` command.

By sweeping over the JPEG quality settings, determine the lowest setting for which the compressed image is indistinguishable from the original. What is the compression ratio (wrt PNG)? That is, what is the ratio of file sizes?

6 Extra Credit: Create your own HDR image

+25 points

For extra credit you can capture, merge, and tone-map your own HDR image.

- Capture images of scenes that actually have a high-dynamic range. A mix of indoor/outdoor or sunlight/shaded are both good options.
- Be sure to save your images in raw format.
- I suggest capturing a series of exposures that are linearly space in the log domain. E.g., each exposure approximately 2x longer than the last.
- Be sure to use the same camera settings for each image: Manual mode with fixed ISO.
- The camera needs to be static between shots. If possible, use a tripod and trigger the camera with your computer `https://www.usa.canon.com/support/eos-utilities`.

There are two mirrorless cameras and tripods in my lab which can be borrowed for one day at a time.

You will be graded on how visually compelling your images are.

Submission Instructions

Your canvas submission should consist of a zip file named **YourDirectoryID_HDR.zip**, for example xyz123_HDR.zip. The file must contain the following:

- HDR.py, *not .ipynb*
- report.pdf

Your code should use relative pathing and assume the images are in the same directory as HDR.py. I.e., access the first image with './exposure1.tiff'.
Please include intermediate results in your report (e.g., images as you sweep K and B).

Collaboration Policy

You are encouraged to discuss ideas with your peers. However, the code should be your own and should represent your understanding of the assignment. **Code should not be shared or copied.** If you reference anyone else's code in writing your project, you must properly cite it in your code (in comments) and in your report.

Please list any individuals you collaborated with at the end of your report.

Plagiarism

Plagiarism of any form will not be tolerated. You are expected to credit all sources explicitly. If you have any doubts regarding what is and is not plagiarism, talk to me.

Credit

Thanks to Ioannis Gkioulekas and Ashok Veeraraghavan for sharing their course resources.

References

[1] I. E. Commission et al. Multimedia systems and equipment-color measurement and management-part 2-1. *Color management-Default RGB color space-sRGB*, 1999.

[2] E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda. Photographic tone reproduction for digital images. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 267–276, 2002.