Homework Assignment 4

Due Wednesday, Nov. 14, at 9:00 am

OVERVIEW

The purpose of this assignment is to explore high dynamic range (HDR) imaging and tonemapping. As we discussed in class, HDR imaging can be used to create floating-point precision images that linearly map to scene radiance values. Tonemapping is the process of compressing the dynamic range of HDR images to an 8-bit range, so that they can be shown on a display. To get full credit, you will need to apply these steps to both an exposure stack provided by us, and one that you capture yourselves. Finally, given that creating an accurate HDR image involves correctly capturing color, you can also do color calibration for extra credit.

Throughout the assignment, we refer to a number of key papers that were also discussed in class. While the assignment and class slides describe most of the steps you need to perform, it is highly recommended that you read the associated papers. There is a “Hints and Information" section at the end of this document that is likely to help.

1. HDR IMAGING (50 POINTS)

For this and the following problem (tonemapping), you will use an exposure stack that we offer. The image files are contained in the ./data/door stack directory of the homework ZIP archive. Figure 1 shows two exposures, as well as a (tonemapped) HDR composite.

While not particularly beautiful, the scene has a number of features that make it a good example for HDR: First, there are two areas with very different illumination and dynamic range that no single exposure can simultaneously capture correctly. Second, both areas include colorful items (Toy Story poster in the background of the bright area, Plus-Plus pieces, SIGGRAPH mugs, and book covers in the dark area) that you can use to evaluate the color rendition of your results. Third, the in-focus area has high-detail features (lettering on the book covers and lens/camera markings) that you can use to evaluate the resolution of your results. Finally, the scene includes a color checker than you can use for color calibration in bonus question.

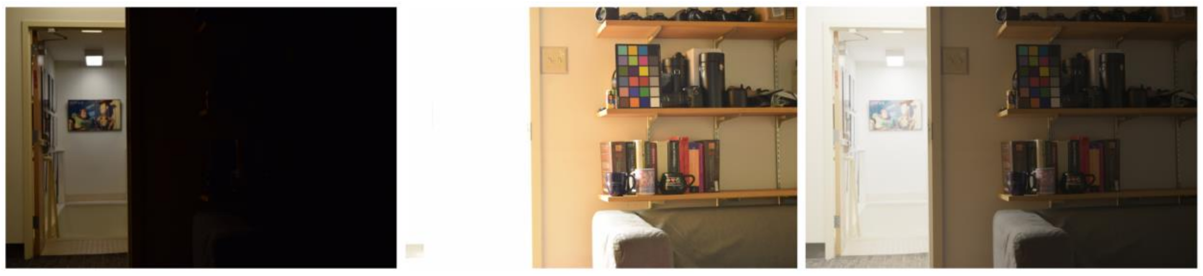


Figure 1: From left to right. Two LDR exposures, and an HDR composite tonemapped using the photographic tonemapping

For reference, both exposure stacks are captured with fixed aperture and ISO, and with shutter speeds equal to $\frac{1}{2048}2^{k-1}$, where $k \in \{1, \ldots, 16\}$ is the index in an image's file name.

As we saw in Assignment 1, RAW files neither are directly readable by image processing programs, nor do they look like actual images. Unlike Assignment 1, where you implemented all of the image rendering steps yourselves, here you will use dcraw to perform some of those steps for us.

Use `dcraw` to convert the RAW .NEF images into linear 16-bit .TIFF images. For this, you should direct `dcraw` to do white balancing using the camera's profile for white balancing, do demosaicing using high-quality interpolation, and use sRGB as the output color space. Read through `dcraw's` documentation, and figure out what the correct set of flags are for this conversion. Make sure to report the flags you use in the report you submit with your solution.

LINEARIZE RENDERED IMAGES (25 POINTS)

Unlike the RAW images, which are linear, the rendered images are non-linear. As we saw in class, before you can merge them into an HDR image, you first need to perform radiometric calibration in order to undo this non-linearity. You will do this using the method by Debevec and Malic [1]. We describe how this works below, but you are strongly encouraged to read at least Section 2.1 of this paper, which explains the method.

An intensity value I_{ij}^k at pixel (i, j) of image k relates to some unknown scene radiance value L_{ij} as,

$$I_{\text{ij}}^k = f\left(t^k L_{ij}\right) \qquad (1)$$

where t^k is the known exposure of image k and f is the unknown non-linearity applied by the camera. If we knew f^{-1}, we could convert I_{ij}^k back to linear measurements.

Instead of f^{-1}, we will recover the function $g = \log(f^{-1})$ that maps pixel values I_{ij}^k to $g\left(I_{\text{ij}}^k\right) = \log L_{ij} + \log t^k$. This is motivated by the fact that the human visual systems responds to logarithmic, instead of linear, intensity. Since the domain of g are the discrete intensity values {0, …,255}, g is basically just a 256-D vector.

Solving for these 256 values may seem impossible, because we know neither g nor L_{ij}. However, if the imaged scene remains static while capturing the exposure stack, we can take advantage of the fact that the value L_{ij} is constant across all LDR images. Then, we can recover g by solving the following least squares optimization problem,

$$\min_{g, L_{ij}} \sum_{i,j} \sum_k w\left(I_{\text{ij}}^k\right)\left[g\left(I_{\text{ij}}^k\right) - \log\left(L_{ij}\right) - \log\left(t^k\right)\right]^2 + \lambda \sum_{z=0}^{255} w(z)\left(\nabla^2 g(z)\right)^2 \qquad (2)$$

As we discussed in class, the weights w have to do with the fact that the linear estimates shoud rely more on well-exposed pixels than on under-exposed or over-exposed pixels. See later in Problem 1 ("Weighting schemes") about what weights exactly you will use. The second term in Equation (2) has to do with the fact that we expect g to be smooth, and therefore we penalize solutions g that have large second-derivative magnitudes. Given that g is discrete, the second derivative can be approximated using a Laplacian filter, that is, $\nabla^2 g(z) = g(z+1) - 2g(z) + g(z-1)$. Solve the least-squares optimization problem of Equation (2) by expressing it in matrix form:

$$(Av - b)^2 \qquad (3)$$

Plot the function g you recovered, then use it to convert the non-linear images I_{ij}^k into linear ones,

$$I_{\mathrm{ij,lin}}^k = \exp\left(g\left(I_{ij}^k\right)\right)$$

You will not use the values L_{ij} you recover from solving (2).

MERGE EXPOSURE STACK INTO HDR IMAGE (15 POINTS).

Now that we have two sets of (approximately) linear images, coming from the RAW and rendered files, it is time to merge each one of them into an HDR image. This part will be common for both sets of linear images. Make sure that each HDR image you create only uses images from one or the other set.

Given a set of k LDR linear images corresponding to different exposures t^k, we can merge them into an HDR image either in the linear or in the logarithmic domain. Linear merging is motivated by physical accuracy, while logarithmic merging, as mentioned above, is motivated by human visual perception.

When using linear merging, the HDR image is formed as:

$$I_{\mathrm{ij,HDR}} = \frac{\sum_k w\left(I_{ij,LDR}^k\right) I_{ij}^k / t^k}{\sum_k w\left(I_{ij,LDR}^k\right)} \qquad (5)$$

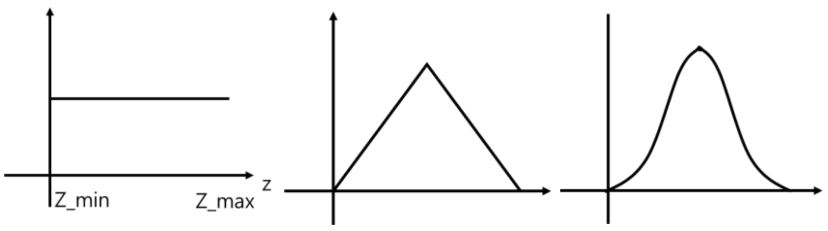
When using logarithmic merging, the HDR image is formed as:

$$I_{\mathrm{ij,HDR}} = \exp\left(\frac{\sum_k w\left(I_{ij,LDR}^k\right)\left(I_{ij}^k - \log t^k\right)}{\sum_k w\left(I_{ij,LDR}^k\right)}\right) \qquad (6)$$

As before, the weights w in Equations (5) and (6) can be used to place more emphasis on well-exposed pixels, and less emphasis on under-exposed or over-exposed ones. See below about what weights to use. Note that we use $I_{ij,LDR}^k$ to refer to the pixel values of the original LDR image, which may be non-linear (when using .jpg files) or linear (when using .TIFF from RAW files).

Implement both linear and logarithmic merging for each of the two exposure stacks. Then, store the resulting HDR images as .HDR files. (See help for the `hdrwrite`.)

Weighting schemes. There are many possible weight choices [3]. You can implement at least two different schemes out of the following three schemes.



(a) uniform (b) tent (c) Gaussian

All of the above weighting schemes assume that the intensity values are in the range $z \in [0, 1]$. So make sure to normalize your LDR images to that range (e.g., divide by 255 if they come from .JPG files). You can experiment with different clipping values Zmin and Zmax, but we recommend Zmin = 0.01, Zmax = 0.99.
Note that, when creating an HDR image from the .JPG stack, you need to use the same weighting scheme in both Equations (2) (linearization) and (5)-(6) (merging).
Implement all of the above weighting schemes, and use them to create HDR images. In total, you will create 8 HDR images: 2 sets of images (RAW and rendered) x 2 merging schemes (linear and logarithmic) x 2 weighting schemes (two out of uniform, tent, and Gaussian).

EVALUATION (10 POINTS). One way to evaluate the results of an HDR creation process is to check its linearity. For this, you can use the color checker (Figure 2). In particular, patches 4, 8, 12, 16, 20, and 24 of the color checker are created to be near-perfectly neutral (gray) patches, with the reflectance of each patch being twice as high as the reflectance of the patch above it. Therefore, in an ideal HDR image, plotting the logarithm of the average luminance in each of these patches should produce a straight line.

For each of the neutral patches, crop a square that is fully contained within the patch and which can be used to compute patch averages. (See help for functions `ginput` and `impixelinfo`.) Make sure to store the coordinates of these cropped squares, so that you can re-use them.

Then, for each HDR image you create, evaluate its linearity as follows: First, convert the HDR image to the XYZ color space, and extract the Y channel (luminance). (See help for function `rgb2xyz` and make sure to use `'Colorspace'` option `'linear-rgb'`.) Second, using the cropped squares you created earlier, compute the average luminance for each of the six neutral patches. Third, perform linear regression to the logarithms of these six average luminances. Fourth and final, compute the least-squares error between the actual average luminance values and the linear fit you created.

Compute this error for each HDR image you create, and discuss how they compare to each other. Additionally, make a logarithmic plot comparing the six average luminances for the various HDR images.

2. TONEMAPPING (50 POINTS)

Now that you have a bunch of HDR images, you need to tonemap them so that you can display them. You will implement two different tonemapping algorithms. For this part, you can use whichever of your 8 created HDR images you like the best.

PHOTOGRAPHIC TONEMAPPING (20 POINTS).

PHOTOGRAPHIC TONEMAPPING (20 POINTS). First, you will implement the tonemapping operator proposed by Reinhard et al. [4], which is a good baseline to start from when displaying HDR images. We describe how to do this below, but you are strongly encouraged to read at least Sections 2 and 3 of this paper, which explain the rationale behind the specific form of this tonemapping operator, the effect of the various parameters, and its relationship to the zone system used when developing film.

Given pixel values $I_{\text{ij,HDR}}$ of a linear HDR image, photographic tonemapping is performed as

$$I_{\text{ij,Tonemapping}} = \frac{\tilde{I}_{\text{ij,HDR}}\left(1+\frac{\tilde{I}_{\text{ij,HDR}}}{\tilde{I}^2_{\text{white}}}\right)}{1+\tilde{I}_{\text{ij,HDR}}} \qquad (8)$$

where

$$\tilde{I}_{\text{white}} = \mathrm{B}\cdot\max_{i,j}(\tilde{I}_{\text{ij,HDR}}) \qquad (9)$$

$$\tilde{I}_{\text{ij,HDR}} = \frac{K}{I_{\text{m,HDR}}} I_{\text{ij,HDR}} \qquad (10)$$

$$I_{\text{m,HDR}} = \exp\left(\frac{1}{N}\sum_{i,j}\log\left(I_{ij,HDR}+\varepsilon\right)\right) \qquad (11)$$

(Note that Equation (11) is different from the corresponding Equation (1) in Reinhard et al. [4]. The version given here is the correct, and the version in the paper is incorrect.) The parameter K is the key, and determines how bright or dark the resulting tonemapped rendition is. The parameter B is the burn, and can be used to suppress the contrast of the result. Finally, N is the number of pixels and ε is a small constant to avoid the singularity of the logarithm function at 0.

Implement the photographic operator and apply it to your RGB HDR images in two ways: First, apply it to each color channel separately. Second, apply it only to the luninance channel Y. For the latter, you can use `rgb2xyz` to convert the HDR image from RGB to XYZ, then convert the image from XYZ to xyY. ($\mathrm{x} = \frac{\mathrm{X}}{\mathrm{X+Y+Z}}, y = \frac{Y}{X+Y+Z}$) While in xyY, tonemap the luminance Y while leaving the chromaticities x,y untouched. Then, invert the color transforms to go back to RGB.

Experiment with different key and burn values. Some reasonable starting values for the parameterers are K = 0.15 and B = 0.95, but to get good tonemaps you will need to explore different values. Plot representative tonemaps for both the RGB and luminance methods, and discuss your results. Make sure to mention which tonemap you like the most.

TONEMAPPING USING BILATERAL FILTERING (30 POINTS).

You will now experiment with tonemapping using bilateral filtering, as proposed by Durand and Dorsey [2]. As in the previous problems, you are encouraged to read through the paper, to find many implementation details and insights about the various parameters.

Given pixel values $I_{\text{m,HDR}}$ of a linear HDR image, tonemapping using bilateral filtering is performed through a sequence of steps:

1. Compute the log intensity, $L_{ij} = \log\left(I_{ij,HDR}\right)$.

2. Compute the base layer using bilatering filtering, $B_{ij} = bilatering\ filter(L_{ij})$.
3. Compute the detail layer, $D_{ij} = L_{ij} - B_{ij}$.
4. Apply an offset and a scale S to the base, $B'_{ij} = \mathrm{S} \cdot \left(B_{ij} - \max_{i,j}(B_{ij})\right)$.
5. Reconstruct the intensity, $I_{ij,TM} = \exp(B'_{ij} + D_{ij})$.

The scale *S* determines the number of stops in the dynamic range of the final tonemap (e.g., 2-4 stops). For bilateral filtering, you can use any of the algorithms suggested in Section 5.1 of [2]. As with photographic tonemapping, apply bilateral filtering both to each RGB channel separately and to luminance only. Experiment with different values of scale S and the bilateral filter parameters. Plot representative tonemaps for both the RGB and luminance methods, and discuss your results. Make sure to mention which tonemap you like the most, as well as compare with the results from photographic tonemapping.

BONUS: IMPLEMENT A DIFFERENT GRADIENT-DOMAIN PROCESSING ALGORITHM (UP TO 50 PTS)

Instead of bilateral filtering, you can also apply gradient domain editing for tonemapping. Please refer [5] to implement this idea. This will add upto 50 points of bonus.

DELIVERABLES

For this project, and all other projects, you must do a project report in HTML. In the report you will describe your algorithm and any decisions you made to write your algorithm a particular way. Use both words and images to show us what you've done (describe in detail your algorithm parameterization for each of your results). You should submit your code and data (images), and the link for HTML by the deadline.

POLICY

Do not search the code on the web. Once detected, you will be penalized by – 200% on your project credits.

REFERENCES

[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, SIGGRAPH '97.

[2] F. Durand and J. Dorsey. *Fast bilateral filtering for the display of high-dynamic-range images.* In Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '02.
[3] K. Kirk and H. J. Andersen. *Noise characterization of weighting schemes for combination of multiple exposures*. In BMVC.
[4] E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda. *Photographic tone reproduction for digital images.* SIGGRAPH '02.
[5] Fattalet al., *Gradient Domain High Dynamic Range Compression*. SIGGRAPH 2002.