Name: _____

EE 193-03 Homework 3

Due via `provide` 11:59pm November 1, 2018

Overview

Your task is to build an ISP in MATLAB which takes "raw" input images and automatically produces decent-looking output. At a minimum, you should implement the following blocks (preferably as separate functions or "cells" in MATLAB):

- Black-level and white-level correction, i.e., picking the raw values which you will consider "black" and "white", and scaling the data as appropriate.
- Demosaicking, using some method which produces better results than plain bilinear interpolation.
- Color correction and/or white balance
- Gamma mapping

In addition to gamma mapping, you may want to implement some kind of camera curve. That is, you may want to use some non-linear mapping of camera pixel values into brightness values.

You should also implement at least one of the following:

- Sharpening
- Denoising, using any method you want other than a gaussian blur

Your completed image should be written to a JPEG or PNG file. JPEG produces much smaller files for camera images, but it may introduce compression artifacts. If you're trying to do careful comparisons on your denoising algorithm, you might prefer PNG.

Test data

The `data` directory contains 5 raw images of various scenes, taken under various lighting conditions. Each is stored as a TIFF file[1], which can be easily read into MATLAB with `imread('filename.tif')`.

For each image, there is also a metadata file containing some information about the capture. The metadata is stored as key-value pairs in a .CSV file, which you can read and parse yourself or with the included `load_metadata` and `metadata_value` functions. You are welcome to use this information to automatically adjust parameters for your algorithm(s).

There is also a small JPEG file for each of the test images, which is a downscaled version of what was produced by my camera. I actually think you can do even better than the camera's algorithm — there are several instances where the camera throws away data in the raw image that would actually make a better picture, such as in the highlights of the clouds.

Submission

Zip up the project directory containing all of your code and submit it with `provide`. Please don't submit the data files; I already have those.

[1] These were created by capturing raw images with my Olympus E-PL6 and then converting the .ORF files to TIFFs with a short Python script using `rawpy` and `imageio`. The TIFF format is one of the few that can handle 16-bit data; most others require pixels to be 8 bits per color channel.

Grading

Since there isn't a single right answer, the criteria for grading are (in order of importance):

- All of the required blocks are included, and the code runs.
- Your approach makes sense, and the algorithms are explained in detail by comments in the code.
- Algorithm produces good-looking results on the test images.
- Algorithm produces good-looking results on some new test images you haven't seen.

I will not explicitly grade your code "style", but if I can't figure out what your code does, it'll be hard for me to give you points for it. Remember that a happy grader is a more generous grader!

If you have questions, please post on Piazza, come to my office hours, or bring them up in class.