Lab 1: Camera pipeline

Cano Andrés, Lorenzo
736078

Tierno Jiménez, Carlos
705548

March 11, 2021

1 Overview

This document is split in two parts. The first one is a walk-through over our implementation of a typical camera pipeline, providing for each step a before and after. We will follow a unique image over this pipeline, applying to it only one variant of the step.

In the next part, we will provide the complete set of images that we have obtained by combining different variations of the steps.

2 Implementation

Linearization: For our RAW images, the linearization step maps the values equal or less than 1023 to 0, and the values of 15600 or over, to 1. The values between 1023 and 15600 are linearly mapped between 0 and 1. The effects of this step can be seen on fig.1. The difference between them is barely noticeable because the display option adjusts each image independently. If the adjustment had been done in a joint fashion, the linearized image would appear black.

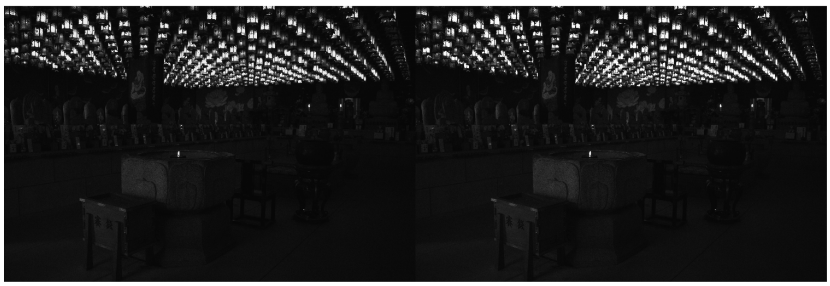


Figure 1: Linearization step: On the left, original raw image, on the right, linearized image.

Demosaicing: The mosaic pattern used by the raw files that we have been working on seems to follow the pattern:

$$\begin{bmatrix} R & G \\ G & B \end{bmatrix}.$$

And the demosaicing has been implemented using the indexing capabilities of MATLAB, both for the NNI and bilinear interpolation demosaicing. On this step we go from a "one-layer" image that has color encoded on the mosaic pattern, to a "three layer" RGB image. The image resulting from this process is a colored image, however, the colors seem to be far from what one could expect (fig.2). This is addressed in the following steps.

Figure 2: Demosaicing step: On the left, the linearized image before demosaicing, on the right, RGB image demosaiced with bilinear interpolation.

White balancing: White balancing is one of the processing steps that most affects how the colors will look on the final image. We have implemented 3 white balancing strategies, gray-world assumption, white-world assumption and a manual balancing (applies the white world assumption on a patch of the image to the whole image).

- The gray world assumption forces the average of the three channels to have the same value, by multiplying each channel by a scalar. In our case, we have forced the red and blue channels to scale to the gray channel average.
- The white world approach assumes that the brightest point in the image is white. And scales the rest of the image accordingly. In our case, after selecting the brightest point, we have scaled the red and blue channels so that their corresponding value on that point is the same as the gray channel.

Figure 3: White balancing step: Manual white balancing by selecting the left side of the rock fountain.

In fig.3 we can see how this step causes the colors to acquire a more natural tone. This is, in big part, because of the chosen method. By using the manual white balancing, and choosing a gray area (the rock fountain), we force that area to be gray in the final picture, and the rest of the picture varies accordingly.

Denoising: In the denoising part we aim to eliminate the very high frequency artifacts present in the image. Three approaches have been implementd:

- Average filtering: The value of the pixel in the filtered picture is the average value of a window of size $n \times m$ around the pixel in the unfiltered image.
- Gaussian filtering: It is very similar to the previous one, except the average of the pixels in the window is now weighted by the value of a Gaussian distribution. Both this and the previous methods have been implemented with the MATLAB's "conv2d" function, the only difference being the kernel used.
- Median filtering: The final value of the pixel is the median value of the window around that pixel in the original image. In our case, instead of applying the median filtering to each channel and adding the channels back together. The median filter has been applied to the gray-scale version of the image, and the differences between the filtered and original image have been noted. Then, in the final image, each channel is a combination of the original channel, and the median-filtered channel, selecting the filtered channel on the pixels where the gray-scale image has changed. The main motivation for this two-step filtering is to minimize the color-shift that can occur with median filtering, although the final result barely changes.

Figure 4 shows how the high-frequency artifacts are eliminated. As the image has a scale of (4022×6024), and the Gaussian filter is applied to a (3×3) window, it is necessary to zoom a lot to be able to appreciate the changes.

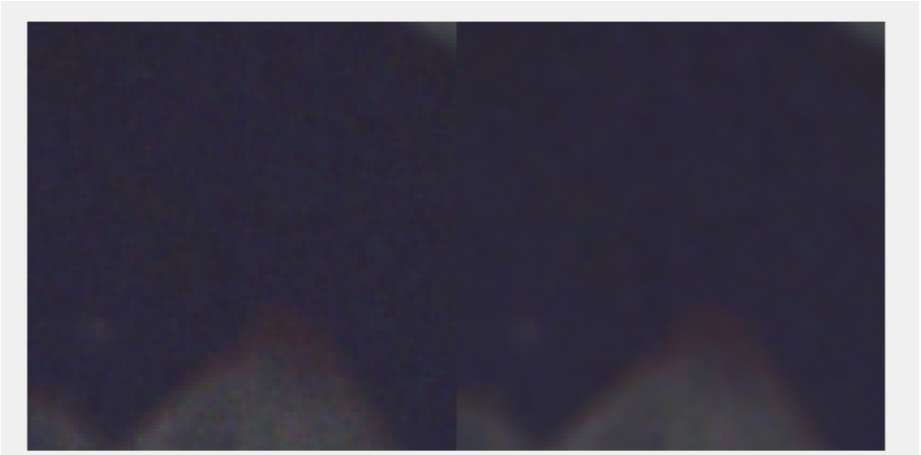


Figure 4: Denoising: Denoising using a Gaussian filter with a window of size (3×3) and $\sigma = 4$. Brightness has been increased for clarity.

Color Balance: This step in a normal pipeline, scales each of the color channels by a scalar factor that depends on the camera. The purpose of this is to move the image from the "sensor perception space" to the "human perception space". As this scalar values are not available for us, our color balance consists on increasing the saturation of the image on the HSV space by a scalar. This scalar value has been chosen by trial and error. For the example in fig.5

Figure 5: Color balancing step: Increase in saturation of 1.3.

Tone reproduction: For tone reproduction we apply both an exposure adjustment, as well as the gamma correction. Both the γ and α parameters have been chosen by hand, with trial and error until we have reached a result that we liked. In fig.6

Figure 6: Tone reproduction: Exposure adjustment of $\alpha = 0.1$ and gamma correction of $\gamma = 1.9$.

Compression: Compression quality can be measured with the Structural SImilarity Index Measure (SSIM), this is commonly used for measuring the similarity between two images, hence, the quality of an image when compressed can be measured with this score.

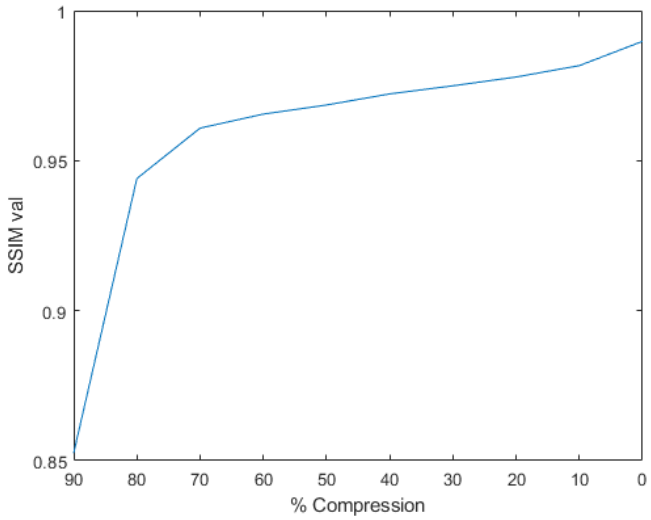
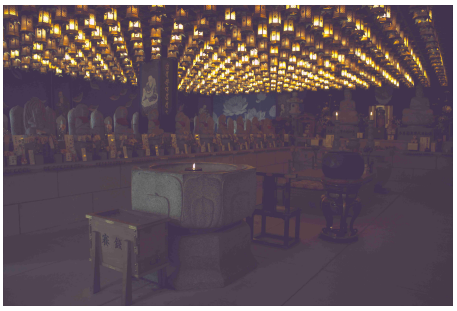


Figure 7: Compression rate vs SSIM

Looking at the plot, we can think a 90% of compression retrieving a 86% on the similarity score, is a really good approach. But many information is lost in that 25% of the image that is not recovered; shapes are the same but not the color information it keeps. Reducing the compression to a 60%, shows an image containing almost all visible details. Being the compression value we would apply to the images, when fine details are not needed.

(a) 90% Compressed



(b) 80% Compressed

(c) 70% Compressed

(d) 60% Compressed

The difference among the following images can be clearly noticed, at the right bottom part of the image, where the shadows becomes more and more smoother, the time we reduce the compression.

3 Results

In this section we show the results of using the different combinations of the variants of each step. There are two demosaicing methods, 3 white balancing methods and 3 denoising methods, which end up in 18 different photographs.

Many of the steps of the image pipeline take some parameters, and to avoid an "explosion" in the number of final images, we have used the same parameters for the different steps. Those parameters are the following.

- For the manual white balancing, the most left side of the rock fountain has been used as the "true gray" reference.
- In all denoising methods, a window of size 3×3 pixels has been used.
- The Gaussian denoising steps have been preformed with a standard deviation $\sigma = 4$.
- In the color balance step, the saturation increase has been of 1.3 times, and the brightness has stayed the same.

- In the tone reproduction step, the gamma correction has been applied with $\gamma = 1.9$ and the exposure correction with $\alpha = 0.1$.
- The image files have been exported in the .jpeg format, with a the default quality parameter of 0.75.

3.1 Bilinear Interpolation

Figure 9: Bilinear interpolation, Gray world, Gaussian filtering.

Figure 10: Bilinear interpolation, Gray world, Mean filtering.

Figure 11: Bilinear interpolation, Gray world, Median filtering.

Figure 12: Bilinear interpolation, Manual balance world, Gaussian filtering.

Figure 13: Bilinear interpolation, Manual balance world, Mean filtering.

Figure 14: Bilinear interpolation, Manual balance world, Median filtering.

Figure 15: Bilinear interpolation, White world, Gaussian filtering.

Figure 16: Bilinear interpolation, White world, Mean filtering.

Figure 17: Bilinear interpolation, White world, Median filtering.

3.2 Nearest Neighbour Interpolation

Figure 18: NN interpolation, Gray world, Gaussian filtering.

Figure 19: NN interpolation, Gray world, Mean filtering.

Figure 20: NN interpolation, Gray world, Median filtering.

Figure 21: NN interpolation, Manual Balance world, Gaussian filtering.

Figure 22: NN interpolation, Manual Balance world, Mean filtering.

Figure 23: NN interpolation, Manual Balance world, Median filtering.

Figure 24: NN interpolation, White world, Gaussian filtering.

Figure 25: NN interpolation, White world, Mean filtering.

Figure 26: NN interpolation, White world, Median filtering.

3.3 Observations

Of all the different variations inside the different steps, it is clear that the one that has the biggest impact is the different white balancing options.

The color balance and tone reproduction also have a really big impact on the final look, and a lot of time has been spent to tune the different parameters in those steps, but a more in-depth analysis of the impact of these parameters would have rendered this document too long.

As the size of the filters used is really small in comparison with the size of the images, the effect of the different filtering steps is the least noticeable one, unless great zoom is used.

And finally, the demosaicing step barely makes any difference between the different images.