Rapport Imagerie Couleur

Merlin Schoose - Simon Quesney

October 2021

1 Vignetting Correction

For this exercise, our implementation can be found in the file: **vignetting_correction.py** and our results can be found in the folder: **vignetting_data**.

Our implementation of a single image vignetting correction algorithm is based on the article: "Revisiting Image Vignetting Correction by Constrained Minimization of log-Intensity Entropy", by Laura Lopez-Fuentes, Gabriel Oliver, and Sebastia Massanet.

This article can be found with the following link: `http://srv.uib.es/wp-content/uploads/2015/06/Rev-vignetting.pdf`

This paper is based on a previous paper and proposes solutions to some technical issues that remained in the original article. The core idea of this solution is to compute and minimize the log-intensity entropy of the image. In a more formal way, we use a polynomial gain function $g_{a,b,c}(r)$, defined as: $g_{a,b,c}(r) = 1 + a * r^2 + b * r^4 + c * r^6$.

It takes as argument a value r, computed from the position of the studied pixel. The function g also depends on its three roots a, b, and c which are computed iteratively to minimize the log-intensity entropy of the image.

However, the article was originally designed to be applied on grayscale images. As a result, we had to adapt the algorithm so that it could work on RGB images. Our first idea was to grayscale our image, to compute the a, b, and c parameters on the grayscaled version and then to apply the corresponding g function on each channels of the original image.

This solution gave some results (which can be seen below or in the "vignetting_data" folder, along with all the other images used to test our implementation), but we also tried using different color space and we found out that the best result could obtained when applying the algorithm on the L channel, in the *HSL* color space.

Figure 1: Comparison of the results of the vignetting correction algorithm (original picture, RGB solution and HSL solution)

Our implementation could still be improved, as one could optimize it a lot (it is implemented in python which is relatively slow and optimizations could be added in the code) and also as one could try out more color spaces or may be find a better way to adapt the paper to RGB images, but overall we are quite satisfied with the quality of our resulting images.

2 Spectral Measurements

For this exercise, our implementation can be found in the file: **spectral_convertion.py** and our results can be found in the folder: **spectral_data**.

We found 24 values that matched the condition ($a > 20$) for the D65 illuminant and 22 for the A illuminant. We could explain this difference by the fact that the A illuminant represents a more incandescent lighting, which hence contains a more orange tone. The Lab color system tries to correct the influence of the illuminant and as a result contains less red and yellow tones, which is represented by the fact that the a and b values are smaller under the A illuminant.

3 Color Correction

For this exercise, our implementation can be found in the file: **color_correction.ipynb** and our results can be found in the folder: **correction_data**.

For the color correction part, our results are all in the notebook. We have found that the best correction to minimize the delta-E values between the color checker in the image and the reference is the linear correction, which gives a 5.5 mean delta-E value. Since it is also the most accurate correction visually, we chose this correction as the best correction of the three.

Between the original image, the standard deep prime DCP and the linear correction, we found that the minimum delta-E was with the linear correction of the original image. It makes sense for this correction to be better than the original one, but it seems strange that it outmatches the deep prime correction. Furthermore, we found with our computations that the delta-E value of the deep prime correction is worse than the original image, which

seems odd and we may have a problem in our computations of XYZ values from RGB values without correction.