TRUE COLOR MEASUREMENTS USING COLOR CALIBRATION TECHNIQUES

A Thesis

Presented to

The Graduate Faculty of The University of Akron

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

Michael Edward Wransky

August, 2015
TRUE COLOR MEASUREMENTS USING COLOR CALIBRATION

TECHNIQUES

Michael Edward Wransky

Thesis

Approved:

Advisor
Malena Espanol PhD

Faculty Reader
Patrick Wilber PhD

Faculty Reader
Jun Ye PhD

Accepted:

Dean of the College
Chand Midha PhD

Dean of the Graduate School
Chand Midha PhD

Date

Department Chair
Timothy Norfolf PhD
ABSTRACT

Color constancy is an image processing problem to correct for colors in digital images distorted by noise and illuminants. There are several scientific applications in which the true color of an object in a digital image is needed for analysis. Without calibrating the colors in the image, the recorded pixel values in the image do not accurately define the true colors. In this thesis, we propose two innovative methods: an empirical Bayesian method and a Color Convolution method to calibrate colors in images. We utilize 20 RAW images from a color constancy image database and compare our methods against the Grey-World assumption, White Patch assumption, and existing Bayesian techniques. Standard convergence techniques verify the stability of Monte Carlo Markov Chain in our Bayesian method and adding regularization to the Color Convolution method provides numerical stability. Our results show that the proposed algorithms outperform other existing approaches by the mean square error of the pixel values between images.
I would like to thank my family, the entire department of Applied Mathematics, and of course Cort for facilitating my journey. Without the support of Dr. Malena Espanol I would have never found my true passion in image processing and computational mathematics.

I dedicate this thesis to my father and my grandfather. Both are important men in my life who helped guide me to find the joys of mathematics.
# TABLE OF CONTENTS

| LIST OF TABLES                                      | vii |
| LIST OF FIGURES                                    | viii |

## CHAPTER

I. INTRODUCTION TO DIGITAL IMAGERY .......................... 1
   1.1 Overview .................................................. 1
   1.2 A Brief Review of Digital Sensors ...................... 1
   1.3 Sources of Distortion in Color Imagery ................ 4

II. EXISTING COLOR CONSTANCY METHODS ....................... 8
   2.1 A Mathematical Representation of Images ............... 8
   2.2 A Reflectance Model ...................................... 10
   2.3 A Bayesian Framework .................................... 14
   2.4 Other Color Constancy Methods .......................... 16

III. TWO NOVEL COLOR CONSTANCY ALGORITHMS ................... 17
   3.1 Using the MacBeth Color Chart ........................... 17
   3.2 A Calibrated Bayesian Model ............................. 18
   3.3 A Color Convolution Model ............................... 24

IV. RESULTS ....................................................... 31
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Image Samples</td>
<td>31</td>
</tr>
<tr>
<td>4.2 Error Metric</td>
<td>31</td>
</tr>
<tr>
<td>4.3 Experimental Results</td>
<td>34</td>
</tr>
<tr>
<td>V. CONCLUSIONS</td>
<td>44</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>47</td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Average MSE for the ten indoor images across all RGB components. . .</td>
</tr>
<tr>
<td>4.2</td>
<td>Average RE for the ten indoor images across all RGB components. . .</td>
</tr>
<tr>
<td>4.3</td>
<td>Average MSE for the ten outdoor images across all RGB components.</td>
</tr>
<tr>
<td>4.4</td>
<td>Average RE for the ten outdoor images across all RGB components. . .</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>A simplified representation of a digital camera.</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>A Bayer Color Filter Array (Bayer CFA)</td>
<td>3</td>
</tr>
<tr>
<td>1.3</td>
<td>An example of the influence of illuminant on the digital image. Left: text taken under sunlight. Right: text taken under indoor lights.</td>
<td>5</td>
</tr>
<tr>
<td>1.4</td>
<td>An example of how luminance from the same light source affects an image. Left: image taken in dim indoor lighting. Right: image taken in bright indoor lighting.</td>
<td>6</td>
</tr>
<tr>
<td>1.5</td>
<td>An example of how noise affects an image. Left: clean image. Right: noisy image.</td>
<td>7</td>
</tr>
<tr>
<td>2.1</td>
<td>The X-Rite 24 color MacBeth Color Chart. Courtesy: <a href="http://www.mathworks.com">http://www.mathworks.com</a>.</td>
<td>9</td>
</tr>
<tr>
<td>3.1</td>
<td>This figure illustrates the basic concept of using a color chart to extract samples of true colors from a given image.</td>
<td>18</td>
</tr>
<tr>
<td>3.2</td>
<td>This figure illustrates the importance of the accept-reject rate in a Metropolis-Hastings sampler. Left: A high acceptance rate allowing the chain to change too much. Middle: A low acceptance rate, making the chain stall at certain values. Right: A good acceptance rate, allowing the chain to sample a good mixture of values.</td>
<td>24</td>
</tr>
<tr>
<td>3.3</td>
<td>A cumulative sum plot of three Monte Carlo Markov Chains demonstrating that removing the burn-in allows for the true stability of the chain to be evaluated.</td>
<td>25</td>
</tr>
<tr>
<td>4.1</td>
<td>Ten indoor images from Gehler’s database.</td>
<td>32</td>
</tr>
<tr>
<td>4.2</td>
<td>Ten outdoor images from Gehler’s database.</td>
<td>33</td>
</tr>
</tbody>
</table>
4.3 MSE results for the ten indoor images using the five algorithms listed in the legend. ......................................................... 36

4.4 RE results for the ten indoor images using the five algorithms listed in the legend. ......................................................... 37

4.5 MSE results for the ten outdoor images using the five algorithms listed in the legend. ......................................................... 39

4.6 RE results for the ten outdoor images using the five algorithms listed in the legend. ......................................................... 40
CHAPTER I
INTRODUCTION TO DIGITAL IMAGERY

1.1 Overview

Since the onset of digital photography, techniques have been developed to analyze geometries in images, classify objects in a scene, detect edges, and many others. Most techniques perform similarly for grey-scale and color images. However, more recent techniques in computer vision and artificial intelligence are not capable of handling color images well mainly due to the inconsistent representation of color. Because of these color inconsistencies, algorithms have been developed to allow the representation of true colors to be consistently represented across images of different scenes in different lighting conditions. Before we can investigate these color constancy algorithms, we first must understand why colors become distorted in digital images.

1.2 A Brief Review of Digital Sensors

Each digital camera consists of a few basic components, namely: a lens, an array of sensors, and a computer processing chip (see Figure 1.1). Each surface in any given scene reflects light. This reflected light is gathered by the lens and focused on the sensor array. Depending on the resolution of the camera, a sensor array is composed
of many thousands of sensor elements. One sensor element corresponds with one picture element, or pixel, in the resulting image.

Each sensor element becomes electrically stimulated when light focused by the lens interacts with the sensor. This electric signal is interpreted as the intensity of the light at that minute location in the scene. These intensities are then represented in the computer as integer numbers ranging from 0 to 255. Each sensor element records a value and this array of values become the grey-scale pixel values for the image.

Any color image is represented by intensity values of red, green, and blue (RGB) colors. A color image is achieved using one of two hardware setups within the digital camera: one sensor array or three sensor arrays. A camera with one sensor array uses a Color Filter Array (CFA) where a single sensor element is designed to capture only intensities from one of the RGB color components. The most common CFA utilized in single sensor array cameras is the Bayer CFA where 50% of the array
Figure 1.2: A Bayer Color Filter Array (Bayer CFA)

captures green color information and the remaining 50% is split evenly between red and blue color information [1]. See Figure 1.2. Once the information is captured an interpolation algorithm is used to construct a full color image of the scenery [2,3].

The other hardware setup is comprised of three sensor arrays, one responsible solely for red, one for green, and one for blue color information. Light passing through the lens is directed through a red, green, or blue filter allowing only certain wavelengths of light to stimulate the sensors. In this arrangement, no interpolation is needed and the color image is constructed by combining the information from the three sensor arrays.
1.3 Sources of Distortion in Color Imagery

1.3.1 Human and Computer Visual Precisions

Color is both a physical phenomenon and a perceptual one. When light interacts with an object, a certain wavelength of light reflects off that object, while all remaining wavelengths are absorbed. A person’s eyes gather information about this specific wavelength and then the brain interprets the information and perceptually forms a color for that object. So, in a physical sense, color is characterized by a wavelength, but in the perceptual sense it is defined by a person’s interpretation. Color as a perception is what we will focus on here.

Human color vision is unique in the sense that the photoreceptors of the retina mix information from red, green, and blue inputs to form a color [4]. Other mammals have a more limited capability and mix information from blue and yellow inputs to form colors. Amongst humans, color is not represented or perceived in a consistent manner neither [5,6]. For instance, individuals who are color blind and can distinguish blue and green on their own, however, cannot distinguish a green number seven surrounded by blue dots [7].

Regardless of the differences amongst the human population, each individual has the ability to perceive each color in a consistent manner regardless of the illuminant. For instance, reading this text under the sunlight is perceived the same as reading it under an incandescent lamp. That is, under each of these light sources, the human observer still sees black letters against a white paper. This ability to perceive
a color consistently regardless of the light source is known as color constancy [8–10].

Digital cameras do not have this same “ability”. That is, taking an image of this page under sunlight will be represented differently than taking the image under indoor lights (see Figure 1.3). Furthermore, neither of these images captures the true color of the object.

The need for a consistent representation of color in digital imagery is growing exponentially. Applications in the sciences and engineering abound and include topics such as crop segmentation in aerial photography, species identification in tropical fish, sorting objects by color, and many more [11–15].

1.3.2 What Changes the Color

The manner in which color is perceived by a sensor array is influenced by several factors. The main factor is known as luminance, which can be understood as how bright the light source is [8,10]. For example, a scene illuminated by a dim light will result in an image whose pixels have relatively low intensity values. So, the dim light imparted less energy on the surface of the objects, making everything appear black.
Figure 1.4: An example of how luminance from the same light source affects an image. Left: image taken in dim indoor lighting. Right: image taken in bright indoor lighting.

On the other hand, if the same scene were bathed in an intense light, the camera’s sensors would receive significantly more energy, resulting in a white washed image (see Figure 1.4). The true colors of the given scene are masked by these changes in luminance.

Another component that distorts a true color is electronic noise. Each sensor in the array is an electronic device meant to record an observation. The observe is communicated to the computer in the form of a signal. Because the electrons in the sensors are never stationary, small signals are constantly generated by a sensor even when an image is not being acquired. These small signals are known as noise. Then, when the camera is acquiring light to take an image, this noise is still present and alters the true intensity level at each sensor. See Figure 1.5 to see how noise distorts an image.

Noise can be mathematically modeled in several ways, but most commonly as
a Normal distribution of mean zero and some variance \([16, 17]\). Many methods have been created to account for the noise in images and reduce the error they produce. While we will employ denoising algorithms in our methods, the impact of luminance on color observations is far more important, therefore, our methods will address the influence of light as a primary focus.

Now that we have an understanding of the physical problem, we will discuss existing color constancy algorithms in the next chapter.
CHAPTER II
EXISTING COLOR CONSTANCY METHODS

2.1 A Mathematical Representation of Images

As mentioned previously, in a computer, an image is stored as an \( N \times M \) matrix. Each pixel \( p(i,j) \) is then an element of the matrix, where the row index is \( i \) and the column index is \( j \). For a grey-scale image, the entire image is stored in a single matrix, but for a color image the computer stores the information in three matrices one for red, green, and blue pixel data. For grey-scale images we will use the matrix \( P_{\text{grey}} \) and for color images we will use the three-dimensional tensor \( P = [P_{\text{red}}, P_{\text{green}}, P_{\text{blue}}] \) which is formed by three separate matrices [18]. For illustration, the matrix \( P_{\text{red}} \) would be represented as

\[
P_{\text{red}} = \begin{bmatrix}
p_{\text{red}}(1,1) & p_{\text{red}}(1,2) & \cdots & p_{\text{red}}(1,M) \\
\vdots & \vdots & \ddots & \vdots \\
p_{\text{red}}(N,1) & p_{\text{red}}(N,2) & \cdots & p_{\text{red}}(N,M)
\end{bmatrix}.
\]

The color corresponding at a pixel is a combined input from the RGB color components, that is \( p(i,j) = (p_{\text{red}}(i,j), p_{\text{green}}(i,j), p_{\text{blue}}(i,j)) \). For example, a bright white pixel would be \( p(i,j) = (255, 255, 255) \) and a dark black pixel would be \( p(i,j) = (0, 0, 0) \). More complicated colors are a combination of the three colors, such as \( p(i,j) = (10, 25, 255) \). In color constancy an embedded reference color chart
is needed in images to either supervise an algorithm or evaluate how closely an algorithm corrects a distorted color. One such color chart is the X-Rite MacBeth Color Chart (see Figure 2.1). Each color on the chart has a known RGB true color representation. Thus, when such a color chart is embedded in an image, we may compare the recorded pixel values of a color with the known true color pixel values. In our experiments, we will use the X-Rite MacBeth color chart as a reference.

The goal in color constancy is to determine the properties of an illuminant such that it can be used to correct any distortions in the recorded colors. One
property that is commonly estimated is the color of the illuminant [8,19–21]. An ideal illuminant would be uniform across the entire scenery and of white color. Hence, when one estimates the color of the illuminant, it is under the assumption that there is a single light source. Further, we can use this estimated color to transform the digital image from the original illuminant to the white illuminant. Then, if every image were taken under this white illuminant, theoretically, the true colors would be represented consistently.

2.2 A Reflectance Model

Given a scenery, the image values acquired at a given pixel \((x, y)\) are in the vector \(p(x, y) = [p_{\text{red}}, p_{\text{green}}, p_{\text{blue}}]^T\). Note that \(p(x, y)\) represents the value of pixels in a continuous two-dimensional space, while \(p(i, j)\) represents the pixels sampled from a discrete representation of this space. Each pixel depends on three factors: the camera’s sensitivity, the reflectance properties of a surface, and the color of the illuminant. The camera’s sensitivity can be understood as the ability of the sensors to correctly record the intensity of a given wavelength of light. This sensitivity can then be modeled for each RGB sensor independently by the function \(C(\lambda) = [C_{\text{red}}(\lambda), C_{\text{green}}(\lambda), C_{\text{blue}}(\lambda)]\), where \(\lambda\) is the wavelength of light. The surface’s reflectance property is the way in which an object at position \((x, y)\) interacts with light of wavelength \(\lambda\). We will model the reflectance as a function \(S(x, y, \lambda)\). Lastly, the color of the illuminant is modeled by the function \(e(\lambda)\).
The value recorded at position \((x, y)\) of the image is modeled as the integral equation

\[
p^{\text{color}}(x, y) = \int_{\omega} e(\lambda)S(x, y, \lambda)C^{\text{color}}(\lambda)d\lambda,
\]

(2.1)

where \(\omega\) spans the wavelengths of visible light. Therefore, estimating the illuminant color is equivalent to solving for \(e(\lambda)\) in (2.1). However, since \(S(x, y, \lambda)\) and \(C(\lambda)\) are also unknown, we simplify model (2.1) as the integral equation

\[
e = \begin{bmatrix} e^{\text{red}} \\ e^{\text{green}} \\ e^{\text{blue}} \end{bmatrix} = \begin{bmatrix} \int_{\omega} e^{\text{red}}(\lambda)C^{\text{red}}(\lambda)d\lambda \\ \int_{\omega} e^{\text{green}}(\lambda)C^{\text{green}}(\lambda)d\lambda \\ \int_{\omega} e^{\text{blue}}(\lambda)C^{\text{blue}}(\lambda)d\lambda \end{bmatrix}.
\]

(2.2)

In words, equation (2.2) integrates the product of all possible illuminant colors in a given RGB color component and the sensitivities for that component at each sensor. The illuminant color is normalized such that it has unit length, that is, \(\Upsilon = e/\|e\|\).

2.2.1 Color Restoration

Once the estimate for \(\Upsilon\) is made, a common practice is to apply a linear transformation to each pixel in the distorted image \([22, 23]\). Since we wish to transform the distorted image as if it were taken under white light, we use

\[
\Upsilon_{\text{white}} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix} = \begin{bmatrix} \Upsilon^{\text{red}}_{\text{white}} \\ \Upsilon^{\text{green}}_{\text{white}} \\ \Upsilon^{\text{blue}}_{\text{white}} \end{bmatrix}.
\]
as our normalized representation of a white illuminant. Then, we solve for the restored pixel value \( r(x, y) \) by computing

\[
p_r(x, y) = \begin{bmatrix}
\frac{\Upsilon_{\text{white}}^{\text{red}}}{\Upsilon_{\text{white}}^{\text{red}}} & 0 & 0 \\
0 & \frac{\Upsilon_{\text{white}}^{\text{green}}}{\Upsilon_{\text{white}}^{\text{green}}} & 0 \\
0 & 0 & \frac{\Upsilon_{\text{white}}^{\text{blue}}}{\Upsilon_{\text{white}}^{\text{blue}}}
\end{bmatrix}
\begin{bmatrix}
\frac{p_{\text{red}}(x, y)}{p_{\text{green}}(x, y)} \\
\frac{p_{\text{green}}(x, y)}{p_{\text{blue}}(x, y)}
\end{bmatrix}.
\]

(2.3)

### 2.2.2 Grey World Assumption

Without further assumptions, the system (2.2) would still be underdetermined. The first method developed, known as the Grey World Assumption, developed by Buchsbaum, solved (2.1) by assuming that the average reflectance of objects in a scene is constant [24]. Mathematically, this assumption states that

\[
\frac{\iint S(x, y, \lambda) \, dx \, dy}{\iint dx \, dy} = k.
\]

(2.4)

Then, inserting this assumption in (2.1), integrating, and dividing by the total scene area yields

\[
\frac{\iint p(x, y) \, dx \, dy}{\iint dx \, dy} = \frac{\iint \int_\omega e(\lambda) S(x, y, \lambda) C(\lambda) d\lambda \, dx \, dy}{\iint dx \, dy} = k \left[ \int_\omega e^{\text{red}}(\lambda) C^{\text{red}}(\lambda) d\lambda \right] = k e.
\]

(2.5)

However, this is equivalent to finding the average pixel value across all RGB color components. Hence, the estimate for the illuminant color is then \( \Upsilon_{\text{Grey}} = ke/\|ke\| \).
Later, Finlayson and Trezzi cast (2.5) as a specific case of a Minkowski norm [19, 20]. Equation (2.5) can be written as

\[ \left( \frac{\int \int p^q(x, y) \, dx \, dy}{\int \int dx \, dy} \right)^{1/q} = ke \]  

(2.6)

for \( q = 1 \). However, since the digital image is discrete, we apply the discretized form of Equation (2.6)

\[ \left( \frac{\sum_{i=1}^{N} \sum_{j=1}^{M} p^q(i, j)}{N \times M} \right)^{1/q} = ke. \]  

(2.7)

2.2.3 White Patch Algorithm

Equation (2.6) allows for the parameter \( q \) to change and represent different assumptions regarding the surface reflectance of objects. For instance, under the assumption that the color of the illuminant is best estimated by the maximum values for the RGB color component, the reflectance of a white surface would estimate the illuminant’s color. This is equivalent to computing Equation (2.6) in the limit \( q \to \infty \) [19, 20]. This assumption is commonly known as the White Patch algorithm and is commonly computed by finding

\[ e = \max_{x,y} p(x, y) = \max_{i,j} p(i, j). \]

Then, \( \Upsilon \) is found by normalizing \( e \) as before and the image is restored using the linear model in Equation (2.3).
2.3 A Bayesian Framework

2.3.1 A Review of Bayes Theorem

Another common model for color constancy utilizes probability theory and Bayes theorem [25, 26]. In its most general form, Bayes theorem states that for any probabilistic events $A$ and $B$,

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}.$$ (2.8)

In words, Equation (2.8) states that the probability of $A$ occurring conditioned to $B$ is a product of probabilities relating the two events. Bayes theorem is easily proven if we consider $A$ is independent of $B$ so in that case we have the equality,

$$P(A|B)P(B) = P(B|A)P(A).$$

In general, methods in Bayesian statistics do not solve for Equation (2.8) exactly but rather solve the equation proportional to a constant. This is done by assuming that $P(B) = 1$ so that

$$P(A|B) \propto P(B|A)P(A).$$ (2.9)

Components of (2.9) are given specific names. The term $P(A|B)$ is known as the posterior distribution and is what we want to estimate. The term $P(B|A)$ is known as the likelihood distribution and can be thought of the likelihood of $B$ happening given $A$. Lastly, $P(A)$ is the prior distribution and is interpreted as the probability of $A$ occurring. Depending on the sampling algorithm used, we make different assumptions regarding the estimated prior and likelihood distributions.
2.3.2 Bayesian Color Constancy

In Rosenberg et al.’s paper, the problem of estimating the illuminant’s color is cast into a Bayesian framework [25,26]. In this model, a given pixel is given by

\[
y = \begin{bmatrix}
y_{\text{red}} \\
y_{\text{green}} \\
y_{\text{blue}}
\end{bmatrix} = \begin{bmatrix}
x_{\text{red}} \ell_{\text{red}} \\
x_{\text{green}} \ell_{\text{green}} \\
x_{\text{blue}} \ell_{\text{blue}}
\end{bmatrix},
\]

(2.10)

where \( x = [x_{\text{red}}, x_{\text{green}}, x_{\text{blue}}] \) are the reflectance values for the RGB color components and \( \ell = [\ell_{\text{red}}, \ell_{\text{green}}, \ell_{\text{blue}}] \) are the luminance values. If we define

\[
L = \begin{bmatrix}
\ell_{\text{red}} & 0 & 0 \\
0 & \ell_{\text{green}} & 0 \\
0 & 0 & \ell_{\text{blue}}
\end{bmatrix},
\]

\[
X = [x_1, x_2, ..., x_{NM}],
\]

\[
Y = [y_1, y_2, ..., y_{NM}],
\]

where the original tensor is reshaped into a matrix \( Y \) of size \( 3 \times NM \) and each \( y_j = [y_{j,\text{red}}^{\text{red}}, y_{j,\text{green}}^{\text{green}}, y_{j,\text{blue}}^{\text{blue}}] \) and \( X \) is also a matrix of size \( 3 \times NM \) with entries \( x_j = [x_{j,\text{red}}^{\text{red}}, x_{j,\text{green}}^{\text{green}}, x_{j,\text{blue}}^{\text{blue}}] \). Then, this model can be written as

\[
Y = LX.
\]

(2.11)

Similar to Equation (2.1), Equation (2.11) is an underdetermined problem where \( X \) and \( L \) are both unknowns. In order to address the underdetermined problem, a Bayesian model views these unknowns as distributions, transforming (3.2) into

\[
P(\ell, x | y) \propto P(y | \ell, x) P(\ell, x) = P(y | \ell, x) P(\ell) P(x),
\]

(2.12)
where the final equality results under the assumption that $P(\ell, x) = P(\ell)P(x)$.

Equation (2.12) has been at the center of the Bayesian color constancy problem where various authors have made assumptions regarding certain unknowns. For example, $P(x)$ has been modeled as a Gaussian distribution or as a non-Gaussian distribution [26]. Later, exact measurements of certain illuminants were made and then incorporated into (2.12) to simulate the distribution for $P(\ell)$ [25]. Overall, the general principle of this Bayes framework has remained the same while certain assumptions or parameters have been modified.

2.4 Other Color Constancy Methods

There are several more models for the estimation of an illuminant’s color. These models range from artificial intelligence based models, such as neural networks, to a more rigorous statistical approach [10,22,27]. We will not be comparing our methods with these algorithms for a few reasons. The Grey World assumption and the White Patch assumption are the most commercially viable color constancy algorithms for industrial application because of their computational speed and efficiency. Depending on the assumptions, the Bayesian approach can also be included under this scope of computationally fast models. The other models such as the neural networks require a considerable amount of CPU time and memory, making them inhibitory and slow. Furthermore, in a real world setting where a company needs to leverage computational cost versus error, the reflectance model and Bayesian model provide the best route to balance these competing factors.
CHAPTER III
TWO NOVEL COLOR CONSTANCY ALGORITHMS

In this section we will present two new methods for restoring the true colors of an image. Both methods rely upon the explicit use of the MacBeth Color Chart. Therefore, we will first develop a sampling methodology that serves as the first steps for both algorithms.

3.1 Using the MacBeth Color Chart

Whenever a color constancy algorithm utilizes a color reference chart, the algorithm is known as a color calibration algorithm. In order to properly use the reference chart, the chart must be placed in frame of the scenery and illuminated by the dominate illuminant of the scenery. Once the image is acquired, there are pixel measurements recorded at positions along the color chart. Because the chart was manufactured to have each color represented as a precise RGB value, we can compare the pixel values of the recorded color against what the true pixel values should be. For instance, Figure 3.1 illustrates the general procedure of placing the color chart in frame, segmenting out the color chart, evaluating the pixel values at the black and white squares and then recording those values. Depending upon how many samples an algorithm requires, we can sample from one, two, or all the way up to all 24 of the colors from the chart.
Figure 3.1: This figure illustrates the basic concept of using a color chart to extract samples of true colors from a given image.

3.2 A Calibrated Bayesian Model

As described in Section 2, the Bayesian model used by many is constructed as follows, a pixel, $y$, is the result of the product

$$
y = \begin{bmatrix} y_{red} \\ y_{green} \\ y_{blue} \end{bmatrix} = \begin{bmatrix} x_{red}l_{red} \\ x_{green}l_{green} \\ x_{blue}l_{blue} \end{bmatrix}, \quad (3.1)$$
where \( x = [x_{\text{red}}, x_{\text{green}}, x_{\text{blue}}] \) are the reflectance values for the RGB color components and \( \ell = [\ell_{\text{red}}, \ell_{\text{green}}, \ell_{\text{blue}}] \) are the luminance values. If we define

\[
L = \begin{bmatrix}
\ell_{\text{red}} & 0 & 0 \\
0 & \ell_{\text{green}} & 0 \\
0 & 0 & \ell_{\text{blue}}
\end{bmatrix},
\]

this model can be written as

\[
y = Lx^T. \tag{3.2}
\]

Without additional information, \( L \) and \( x \) would still be unknown as before. However, with a color chart as a reference we have a number of known values of the reflectance for certain colors. The simplest model for these reflectances is the true RGB values the color should be represented in the computer as. Then, for a known reflectance, \( \hat{x} \), and the corresponding pixel value on the color chart, \( \hat{y} \), we can compute

\[
\ell = \hat{y}/\hat{x}
\]

or, thinking along the lines of restoring the true values at \( y \), write

\[
\ell^{-1} = \hat{x}/\hat{y}
\]

such that if

\[
L^{-1} = \begin{bmatrix}
\ell_{\text{red}}^{-1} & 0 & 0 \\
0 & \ell_{\text{green}}^{-1} & 0 \\
0 & 0 & \ell_{\text{blue}}^{-1}
\end{bmatrix},
\]

19
then (3.2) becomes

\[ L^{-1}y^T = x. \]  

(3.3)

Therefore, in this context, we are left to estimate \( L^{-1} \) in order to find the unknown reflectances for the rest of the image.

Using the known true color values from the color chart allows us to revise Rosenberg’s Bayesian framework. We now estimate the posterior likelihood of the true illuminant color \( \ell^{-1} \) by

\[ P(\ell^{-1}|\hat{y}) \propto P(\hat{y}|\ell^{-1})P(\ell^{-1}). \]  

(3.4)

Because we are able to use the color chart for comparing some of the \( y \) and \( x \) values in the image, we are able to collect sufficient statistics to estimate a good sample for the likelihood distribution. The only unknown left is the prior probability of \( \ell^{-1} \). Instead of making any assumptions regarding the characteristics of this prior probability, such as it is uniformly or normally distributed, we will make no assumptions and will use the Bayesian sampling technique known as Metropolis-Hastings (M-H).

3.2.1 M-H Sampling

M-H sampling is a variation of an accept-reject Monte Carlo Markov Chain (MCMC) method [28]. A Monte Carlo Markov Chain is a simulated chain of values in which the next value to be generated is only dependent on the previous value. An accept-reject MCMC evaluates each new value of the chain to determine if it will accept the value or reject the new value and keep the previously accepted value. Unlike Gibbs
sampling which uses assumptions regarding the prior and likelihood distributions to
generate values in a MCMC, M-H sampling makes no assumption regarding the prior
and uses the accept-reject method to sample values from this distribution. The M-H
sampling method is advantageous in situations where making an explicit assumption
about the prior distribution significantly influences the results [28–30].

Our M-H method follows the following steps in Algorithm 1.

Then, once we compute the maximum likelihood value, $\tilde{\ell}^{-1}$, we can define

$$
\tilde{L}^{-1} = \begin{bmatrix}
\tilde{\ell}_{\text{red}}^{-1} & 0 & 0 \\
0 & \tilde{\ell}_{\text{green}}^{-1} & 0 \\
0 & 0 & \tilde{\ell}_{\text{blue}}^{-1}
\end{bmatrix}
$$

such that the restored pixel value $y_r$ is

$$
y_r = \tilde{L}^{-1} y.
$$

Hence, the proposed algorithm to restore the true colors of the image is then given
by Algorithm 2.

3.2.2 The Important Role of $\delta$

As specified in Algorithm 1, the parameter $\delta$ determines the accept-reject rate, or in
other words, $\delta$ determines how often the chain progresses or remains at its current
state [30]. If the accept-reject rate is too high, then each step in the chain is accepted
and the chain does not adequately sample for the prior distribution. Likewise, if
the accept-reject rate is low, then the chain remains stuck on a certain value and
very few samples are gathered. Having an accept-reject rate around 30% allows a
Algorithm 1 Metropolis Hastings Sampling Algorithm

1: For all 24 colors in the color chart sample a single pixel of the color and compute

\[ \ell^{-1}[ref] = [\hat{x}_{\text{red}}/\hat{y}_{\text{red}}, \hat{x}_{\text{green}}/\hat{y}_{\text{green}}, \hat{x}_{\text{blue}}/\hat{y}_{\text{blue}}], \]

where \( ref \) represents one of the 24 colors from the chart.

2: Compute \( \log(\ell^{-1}) \) to transform the values to a log normal distribution.

3: Find \( \bar{\ell}^{-1} = [\text{mean}(\log(\ell^{-1}_{\text{red}})), \text{mean}(\log(\ell^{-1}_{\text{green}})), \text{mean}(\log(\ell^{-1}_{\text{blue}}))] \).

4: for each color component do

5: Initialize the chain \( i = 0, C^i = \bar{\ell}^{-1}_{\text{color}}, i = i + 1. \)

6: Select \( \delta \) which determines the accept-reject rate.

7: Pick \( C^* \) randomly from the uniform proposal \( v = C^i + 2\delta u - \delta \), where \( u \sim \text{Unif}(0, 1) \).

8: Calculate \( R = \frac{h(C^*)v(C^i|C^*)}{h(C^i)v(C^*|C^i)} \), where \( h(C) \sim \text{Unif}(C^* - C^i, \delta) \). \( R \) is the fundamental quantity that is used to accept or reject each new value in the chain.

9: If for \( u \) in \( \text{Unif}(0, 1) \) that \( u < R \) then \( C^{i+1} = C^* \), else \( C^{i+1} = C^i \).

10: end for

11: Repeat this procedure until chain convergence and an accept-reject rate of about 30%.

12: Compute \( \bar{\ell}^{-1}_{\text{color}} = \text{mean}(\exp(C)) \).
Algorithm 2 Bayesian Color Restoration

1: Given the $N \times M$ image $P$ and inverse illuminant estimate $\tilde{\ell}^{-1}$

2: for $i = 1 : N$ do

3: for $j = 1 : M$ do

4: Compute

$$y^r(i, j) = \tilde{L}^{-1} y(i, j) = \begin{bmatrix} \tilde{\ell}_{red}^{-1} & 0 & 0 \\ 0 & \tilde{\ell}_{green}^{-1} & 0 \\ 0 & 0 & \tilde{\ell}_{blue}^{-1} \end{bmatrix} \begin{bmatrix} y_{red} \\ y_{green} \\ y_{blue} \end{bmatrix}.$$ 

5: end for

6: end for

proper mixing of values such that when we find the maximum likelihood value we are evaluating the proper distribution. Figure 3.2 illustrates these concepts.

3.2.3 MCMC Convergence

There are several statistical means in which to evaluate the convergence of a MCMC. Analyzing the convergence of a MCMC provides the needed evidence to justify a stable and viable solution and that the sampling technique used was correctly developed. One of the simplest ways to evaluate convergence is to display the cumulative sum of the chain. The cumulative sum plot effectively shows if the chain varied greatly or barely at all in the final steps of the chain. However, since the chain has to be initialized by some value, it is common practice to remove 20 – 30% of the beginning steps of the MCMC because in those first few hundred steps the chain can still vary.
Figure 3.2: This figure illustrates the importance of the accept-reject rate in a Metropolis-Hastings sampler. Left: A high acceptance rate allowing the chain to change too much. Middle: A low acceptance rate, making the chain stall at certain values. Right: A good acceptance rate, allowing the chain to sample a good mixture of values.

significantly, hence skewing the true results that occur at the end of the chain. This 20 – 30% of the removed chain is known as the burn-in [28]. Figure 3.3 demonstrates why removing the burn-in provides better information regarding the true behavior of a MCMC.

We will also use the Monte Carlo error developed in [28, 30] to evaluate the convergence of our MCMC’s.

3.3 A Color Convolution Model

Given the true color $\hat{x}$ from a color chart, we will model the observed pixel value $y$ as

$$\hat{x}_c * h_c = \int_V \hat{x}_c h_c dV = y_c, \tag{3.5}$$

where $c$ represents one of the RGB color components, $h$ is the unknown distortion function, and $V$ represents the span of the entire spatial domain. Solving for $h$ in
Equation (3.5) would require a computationally expensive deconvolution. To avoid this, we use a Fourier transform and transform the equation into

\[
\hat{x}_c * h_c = y_c \rightarrow \mathcal{F}\{\hat{x}_c\} \times \mathcal{F}\{h_c\} = \mathcal{F}\{y_c\},
\]

(3.6)

or equivalently

\[
\hat{X}_c \times H_c = Y_c,
\]

where \(\hat{X}\), \(Y\), and \(H\) are the Fourier transforms for \(\hat{x}\), \(y\), and \(h\), respectively. Note that by transforming Equation (3.5) into the Fourier domain, we have simplified our problem from a convolution to simple multiplication.

Now, suppose we sample a \(k \times k \times 3\) patch, \(\phi_i = [\phi_{\text{red}}, \phi_{\text{green}}, \phi_{\text{blue}}]\), corresponding to the \(ith\) color on the color chart. Then, each element of \(\phi_i\) is the image...
pixel value of this color. The corresponding $k \times k \times 3$ tensor whose every element is the true color pixel value is $\xi_i = [\xi_{\text{red}}, \xi_{\text{green}}, \xi_{\text{blue}}]$. For example, if we were to sample from the black square in the MacBeth color checker the elements of $\xi_i$ would all be $[52, 52, 52]$ where as the recorded image sample $\phi_i$ could contain different values. Working with these color patches, we redefine Equation (3.6) such that

$$
\xi_i \ast h = \phi_i \rightarrow \mathcal{F}_2D\{\xi_i\} \odot \mathcal{F}_2D\{h\} = \mathcal{F}_2D\{\phi_i\} \rightarrow \Xi_i \odot H = \Phi_i,
$$

where $\Phi_i$ and $\Xi_i$ denote the two-dimensional Fourier transform of $\phi_i$ and $\xi_i$. Note that once we have transformed into the Fourier domain, we no longer view $\Phi_i$ and $\Xi_i$ as matrices but rather as vector fields. Similarly, $H = [H_{\text{red}}, H_{\text{green}}, H_{\text{blue}}]$ is a scalar tensor field. Therefore, $\Xi_i \odot H$ represents component-wise multiplication for each of the RGB color components. Suppose we were to take $\tilde{N}$ number of color samples, then we have

$$
\Xi_1 \odot H = \Phi_1
$$

$$
\Xi_2 \odot H = \Phi_2
$$

$$
\vdots
$$

$$
\Xi_{\tilde{N}} \odot H = \Phi_{\tilde{N}}.
$$

The goal is to solve for $H$ given these $\tilde{N}$ equations such that we minimize the error. In order to minimize an error, we need to define a cost function. We will use

$$
J(H) = \sum_{i=1}^{\tilde{N}} \|\Xi_i \odot H - \Phi_i\|_F^2,
$$

(3.8)
where \( \|p\|_F^2 = \sum_{k=1}^{N} \sum_{j=1}^{M} p(k, j)^2 \). Then, to minimize \( J \) we will set \( \delta J = 0 \) where we differentiate \( J \) using a Gateaux derivative. Then,
\[
0 = \delta J = \sum_{i=1}^{\tilde{N}} \frac{\delta}{\delta \alpha} \langle \Xi_i \odot H - \Phi_i, (H + \alpha \delta)\Xi_i - \Phi_i \rangle |_{\alpha=0}. 
\] (3.9)

Simplifying gives
\[
0 = \sum_{i=1}^{\tilde{N}} \frac{\delta}{\delta \alpha} \langle \Xi_i \odot H - \Phi_i | [\Xi_i \odot H + \alpha \delta \Xi_i - \Phi_i] \rangle |_{\alpha=0} 
\] (3.10)

\[
= \sum_{i=1}^{\tilde{N}} \delta \Xi_i \odot \Xi_i \odot H - \delta \Xi_i \odot \Phi_i |_{\alpha=0} 
\] (3.11)

\[
= \sum_{i=1}^{\tilde{N}} \langle \bar{\Xi}_i \odot (\Xi_i \odot H - \Phi_i), \delta \rangle, 
\] (3.12)

where \( \bar{\Xi}_i \) denotes the complex conjugate of \( \Xi_i \). Equation (3.12) must be true for any value of \( \delta \). In particular, for \( \delta = 1 \) we obtain
\[
0 = \sum_{i=1}^{\tilde{N}} \{ \bar{\Xi}_i \odot (\Xi_i \odot H - \Phi_i) \}. 
\] (3.13)

Rearranging Equation (3.13) gives the joint solution for \( H \) across all \( \tilde{N} \) colors
\[
H \sum_{i=1}^{\tilde{N}} (\Xi_i \odot \Xi_i) = \sum_{i=1}^{\tilde{N}} \bar{\Xi}_i \odot \Phi_i 
\] (3.14)

\[
\Rightarrow H = \frac{\sum_{i=1}^{\tilde{N}} \bar{\Xi}_i \odot \Phi_i}{\sum_{i=1}^{\tilde{N}} \Xi_i \odot \Xi_i}.
\] (3.15)

However, the solution in Equation (3.14) will not yield meaningful results because the problem is ill-posed and some amount of regularization is needed. The source of the problem’s ill-posedness is because \( \xi_i \) contains only one unique element.
When we compute $\Xi_i$ the two-dimensional Fourier transform interprets this single unique value as representing a signal of constant frequency. Hence, $\Xi_i$ is all zeros except for the first matrix element across the three matrices in the tensor. This sparse tensor leads to a sparse $H$ which is not useful for our application. In order to address this ill-posedness, we propose to add noise to $H$ according to Algorithm 3.

**Algorithm 3 Bayesian Noise Addition**

1: Given each $\xi_i$

2: for $l = 1 : k$ do

3: for $m = 1 : k$ do

4: Sample three random values, $\eta = [\eta_1, \eta_2, \eta_3]$ from the array of integers $[-3, -2, -1, 0, 1, 2, 3]$.

5: Compute $\xi_i(l, m) = \xi_i(l, m) + \eta$, where $\xi_i(l, m)$ denotes the $l$th, $m$th tensor element.

6: end for

7: end for

8: Transform to the Fourier domain $\xi_i \rightarrow \tilde{\xi}_i$.

Once we have the noisy $\tilde{\Xi}_i$ we can solve Equation (3.14) as before but now as

$$H = \frac{\sum_{i=1}^{N} \tilde{\Xi}_i \otimes \Phi_i}{\sum_{i=1}^{N} \tilde{\Xi}_i \otimes \tilde{\Xi}_i}. \quad (3.16)$$

Once we have obtained the Fourier domain version of $H$ we could use an inverse Fourier transform to convert the solution back into the regular domain. Instead, we will keep $H$ in the Fourier space and correct a new pixel value in the Fourier
space. For instance, given a pixel $p = [40, 38, 53]$ we will first form a $k \times k \times 3$ tensor $T$, where every element is the same value, $p$. Then, we compute

$$P_{\text{corrected}} = \mathcal{F}^{-1}\{\mathcal{F}\{T\}.H\},$$

where $./$ denotes element-wise division. Finally, the corrected pixel $p_{\text{corrected}}$ is any element in the tensor $P_{\text{corrected}}$.

The full algorithm for correcting colors in an image using the Color Deconvolution method is given in Algorithm 4.

**Algorithm 4 Fourier Color**

1: Given the image, sample $\tilde{N}$, $k \times k \times 3$ color patches, $\phi_i$, from the MacBeth Color Checker.

2: Create the corresponding true color tensor $\xi_i$ for each of the $N$ color patches.

3: Follow Algorithm 3 to add noise to $\xi_i$ to obtain $\tilde{\xi}_i$ to stabilize the solution.

4: Transform into the Fourier domain. $\tilde{\xi}_i \rightarrow \tilde{\xi}_i$ and $\phi_i \rightarrow \Phi_i$.

5: Compute

$$H = \frac{\sum_{i=1}^{\tilde{N}} \tilde{\xi}_i \odot \Phi_i}{\sum_{i=1}^{N} \tilde{\xi}_i \odot \tilde{\xi}_i}.$$

6: Given a distorted pixel value, $p$, form a $k \times k \times 3$ tensor $P$ whose elements are all $p$.

7: Compute

$$P_{\text{corrected}} = \mathcal{F}^{-1}\{\mathcal{F}\{T\}.H\}.$$

8: Obtain the corrected color pixel value $p_{\text{corrected}}$ by taking one element from the tensor $P_{\text{corrected}}$. 
In the next section, we will design an experiment to test our algorithms with others.
CHAPTER IV
RESULTS

4.1 Image Samples

To compare the respective performances of the algorithms we will use samples taken from Gehler’s Bayesian Colour Constancy Revisited database [25]. This dataset is freely available for download at http://files.is.tue.mpg.de/pgehler/projects/color/. From this database, we select ten indoor images and ten outdoor images. Each image has a MacBeth Color Checker in frame. The images we worked with are shown in Figures 4.1 and 4.2.

4.2 Error Metric

In order to evaluate the differences in performance between the algorithms, we will use the mean square error (MSE)

$$MSE_{color} = \frac{1}{n} \sum_{i=1}^{n} (p_{restored}^{color} - p_{true}^{color})^2,$$

and relative error (RE)

$$RE_{color} = \frac{1}{n} \sum_{i=1}^{n} \frac{|p_{restored}^{color} - p_{true}^{color}|}{p_{true}^{color}},$$

(4.1)

(4.2)
Figure 4.1: Ten indoor images from Gehler’s database.
Figure 4.2: Ten outdoor images from Gehler’s database.
where \textit{color} is one of the three RGB color components. Now that we have a means to evaluate how close each algorithm restores a pixel to its true color, we will evaluate the algorithms with the following experiment.

4.3 Experimental Results

We ran all 20 images taken from Gehler’s database through Algorithm 5. For our Bayesian method our chains sampled 10,000 values and had a burn-in of 2500 steps. Each chain reached convergence using the criteria specified earlier. Average accept-reject rates ranged from $29 - 35\%$. All chains had Monte Carlo errors in the acceptable ranges.

Figure 4.3 and Figure 4.4 show the performance results for the five algorithms tested for the ten indoor images. The \textit{x}-axis represents images one through ten as shown in Figure 4.1. Note that both figures have the MSE and RE arranged with the red component on top, green in the middle, and blue on the bottom. Tables 4.1 and 4.2 display the results for the average error results across all RGB components.

Figure 4.5 and Figure 4.6 show the performance results for the five algorithms tested for the ten outdoor images. The \textit{x}-axis represents images one through ten as shown in figure 4.2. Note that both figures have the MSE and RE arranged with the red component on top, green in the middle, and blue on the bottom. Tables 4.3 and 4.4 display the results for the average error results across all RGB components.
Algorithm 5 Compare Algorithms

1: Given each of the 20 images obtained from the database sample a $k \times k$ patch of all 24 colors on the MacBeth Color Checker.

2: Process the image using either the Grey-World, White Patch, or Gehler’s Bayesian algorithm. Note that for the Bayesian algorithm we will use the proper prior distribution as empirically determined by Gehler.

3: Process the image using our Bayesian method and the Color Convolution approach.

4: Restore the pixel values in the original image according to the appropriate algorithm dependent on the method.

5: for each restored image, $P_m$, where $m$ represents the method used, i.e. Grey-World, Bayesian, etc., do

6: for $i = 1 : 24$, where $i$ represents the $ith$ color in the MacBeth chart, do

7: Take a new $k \times k$ sample patch of the color in the restored images. Call this tensor $S_i$.

8: end for

9: Then find the MSE and RE according to equations 4.1 and 4.2 using values in $S_i$.

10: end for
Figure 4.3: MSE results for the ten indoor images using the five algorithms listed in the legend.
Figure 4.4: RE results for the ten indoor images using the five algorithms listed in the legend.
<table>
<thead>
<tr>
<th>Image Number</th>
<th>Grey-World</th>
<th>White Patch</th>
<th>Gehler</th>
<th>Calibrated Bayes</th>
<th>Convolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>539</td>
<td>346</td>
<td>415</td>
<td>412</td>
<td>233</td>
</tr>
<tr>
<td>2</td>
<td>1215</td>
<td>963</td>
<td>921</td>
<td>1922</td>
<td>1027</td>
</tr>
<tr>
<td>3</td>
<td>1166</td>
<td>544</td>
<td>619</td>
<td>541</td>
<td>339</td>
</tr>
<tr>
<td>4</td>
<td>777</td>
<td>776</td>
<td>777</td>
<td>792</td>
<td>512</td>
</tr>
<tr>
<td>5</td>
<td>2349</td>
<td>19328</td>
<td>2732</td>
<td>1217</td>
<td>719</td>
</tr>
<tr>
<td>6</td>
<td>4186</td>
<td>3936</td>
<td>3784</td>
<td>995</td>
<td>823</td>
</tr>
<tr>
<td>7</td>
<td>1037</td>
<td>812</td>
<td>828</td>
<td>957</td>
<td>682</td>
</tr>
<tr>
<td>8</td>
<td>6344</td>
<td>8751</td>
<td>8170</td>
<td>2587</td>
<td>1676</td>
</tr>
<tr>
<td>9</td>
<td>3078</td>
<td>631</td>
<td>501</td>
<td>575</td>
<td>306</td>
</tr>
<tr>
<td>10</td>
<td>2908</td>
<td>1933</td>
<td>3535</td>
<td>2284</td>
<td>1381</td>
</tr>
</tbody>
</table>

Table 4.1: Average MSE for the ten indoor images across all RGB components.
Figure 4.5: MSE results for the ten outdoor images using the five algorithms listed in the legend.
Figure 4.6: RE results for the ten outdoor images using the five algorithms listed in the legend.
<table>
<thead>
<tr>
<th>Image Number</th>
<th>Grey-World</th>
<th>White Patch</th>
<th>Gehler</th>
<th>Calibrated Bayes</th>
<th>Convolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.26</td>
<td>0.25</td>
<td>0.26</td>
<td>0.25</td>
<td>0.24</td>
</tr>
<tr>
<td>2</td>
<td>0.51</td>
<td>0.45</td>
<td>0.4</td>
<td>0.37</td>
<td>0.48</td>
</tr>
<tr>
<td>3</td>
<td>0.33</td>
<td>0.30</td>
<td>0.31</td>
<td>0.28</td>
<td>0.28</td>
</tr>
<tr>
<td>4</td>
<td>0.31</td>
<td>0.32</td>
<td>0.33</td>
<td>0.31</td>
<td>0.29</td>
</tr>
<tr>
<td>5</td>
<td>0.47</td>
<td>1</td>
<td>0.5</td>
<td>0.33</td>
<td>0.31</td>
</tr>
<tr>
<td>6</td>
<td>0.58</td>
<td>0.58</td>
<td>0.57</td>
<td>0.34</td>
<td>0.33</td>
</tr>
<tr>
<td>7</td>
<td>0.39</td>
<td>0.37</td>
<td>0.38</td>
<td>0.36</td>
<td>0.37</td>
</tr>
<tr>
<td>8</td>
<td>1.06</td>
<td>1.27</td>
<td>1.23</td>
<td>0.42</td>
<td>0.6</td>
</tr>
<tr>
<td>9</td>
<td>0.39</td>
<td>0.31</td>
<td>0.29</td>
<td>0.27</td>
<td>0.26</td>
</tr>
<tr>
<td>10</td>
<td>0.53</td>
<td>1</td>
<td>0.59</td>
<td>0.47</td>
<td>0.44</td>
</tr>
</tbody>
</table>

Table 4.2: Average RE for the ten indoor images across all RGB components.
<table>
<thead>
<tr>
<th>Image Number</th>
<th>Grey-World</th>
<th>White Patch</th>
<th>Gehler</th>
<th>Calibrated Bayes</th>
<th>Convolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5312</td>
<td>5386</td>
<td>3560</td>
<td>1764</td>
<td>1027</td>
</tr>
<tr>
<td>2</td>
<td>5723</td>
<td>5707</td>
<td>4211</td>
<td>1720</td>
<td>1019</td>
</tr>
<tr>
<td>3</td>
<td>4995</td>
<td>5857</td>
<td>4185</td>
<td>1892</td>
<td>1143</td>
</tr>
<tr>
<td>4</td>
<td>5664</td>
<td>8310</td>
<td>7003</td>
<td>2449</td>
<td>1650</td>
</tr>
<tr>
<td>5</td>
<td>570</td>
<td>652</td>
<td>650</td>
<td>646</td>
<td>389</td>
</tr>
<tr>
<td>6</td>
<td>3941</td>
<td>4153</td>
<td>4371</td>
<td>1352</td>
<td>803</td>
</tr>
<tr>
<td>7</td>
<td>3055</td>
<td>3258</td>
<td>3596</td>
<td>1190</td>
<td>704</td>
</tr>
<tr>
<td>8</td>
<td>5940</td>
<td>6495</td>
<td>6786</td>
<td>1970</td>
<td>1230</td>
</tr>
<tr>
<td>9</td>
<td>4052</td>
<td>4992</td>
<td>5011</td>
<td>1181</td>
<td>550</td>
</tr>
<tr>
<td>10</td>
<td>1066</td>
<td>1250</td>
<td>1248</td>
<td>790</td>
<td>398</td>
</tr>
</tbody>
</table>

Table 4.3: Average MSE for the ten outdoor images across all RGB components.
<table>
<thead>
<tr>
<th>Image Number</th>
<th>Grey-World</th>
<th>White Patch</th>
<th>Gehler</th>
<th>Calibrated Bayes</th>
<th>Convolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.93</td>
<td>0.93</td>
<td>0.79</td>
<td>0.34</td>
<td>0.44</td>
</tr>
<tr>
<td>2</td>
<td>0.94</td>
<td>0.93</td>
<td>0.82</td>
<td>0.33</td>
<td>0.43</td>
</tr>
<tr>
<td>3</td>
<td>0.92</td>
<td>1</td>
<td>0.87</td>
<td>0.36</td>
<td>0.48</td>
</tr>
<tr>
<td>4</td>
<td>0.94</td>
<td>1.15</td>
<td>1.08</td>
<td>0.39</td>
<td>0.55</td>
</tr>
<tr>
<td>5</td>
<td>0.28</td>
<td>0.3</td>
<td>0.3</td>
<td>0.29</td>
<td>0.26</td>
</tr>
<tr>
<td>6</td>
<td>0.76</td>
<td>0.78</td>
<td>0.79</td>
<td>0.31</td>
<td>0.39</td>
</tr>
<tr>
<td>7</td>
<td>0.68</td>
<td>0.68</td>
<td>0.71</td>
<td>0.29</td>
<td>0.35</td>
</tr>
<tr>
<td>8</td>
<td>0.99</td>
<td>1.01</td>
<td>1.03</td>
<td>0.34</td>
<td>0.48</td>
</tr>
<tr>
<td>9</td>
<td>0.54</td>
<td>0.6</td>
<td>0.6</td>
<td>0.31</td>
<td>0.28</td>
</tr>
<tr>
<td>10</td>
<td>0.38</td>
<td>0.43</td>
<td>0.43</td>
<td>0.27</td>
<td>0.28</td>
</tr>
</tbody>
</table>

Table 4.4: Average RE for the ten outdoor images across all RGB components.
CHAPTER V
CONCLUSIONS

From our experiments in the previous section we are able to draw some inferences regarding the behavior of the algorithms. The most apparent observation is that there are drastic differences in performance for a single algorithm between the RGB color components. This is mostly due to the type of camera used to obtain the images in Gehler’s database. These images were taken using a digital camera that used a single array set-up and thus utilized a Bayer CFA and then an interpolation algorithm to generate the final image. This process introduces a considerable amount of variation within the RGB color components, resulting in the behavior seen in our results. Although a three sensor array set-up is more advantageous for analysis, most digital cameras and camera phones on the market use a Bayer CFA for sake of efficiency and cost. Because of this, it is even more important for a color constancy algorithm to perform well across all three RGB components even when an image is the result of a Bayer CFA.

In regards to the MSE results obtained, the White Patch algorithm is the least stable. The White Patch algorithm appears to perform poorly when the scenery is in a dark indoor environment in subdued lighting. This makes sense given the assumptions of the algorithm. Since it assumes that the maximum pixel response
originates from a white surface, when a white surface is captured in shaded lighting, the calibration is instantly skewed. As can be seen in the MSE results for the outdoor scenery, the White Patch algorithm is more stable because the images are in more direct light. However, an ideal algorithm would not have such drastic inconsistencies in performance depending on the type of illuminant. Because of this, the White Patch algorithm is not a recommended color constancy solution.

Our proposed Calibrated Bayesian and Color Convolution models outperform the standard Grey-World algorithm as well as Gehler’s Bayesian approach. Both the Calibrated Bayesian and Color Convolution models are stable and perform similarly regardless of an outdoor or indoor illuminant. It is interesting to note that although the differences were not large, the Color Convolution model outperformed the Calibrated Bayesian model in regards to the MSE. Yet, the Calibrated Bayesian model outperformed the Color Convolution model in terms of the RE. These results suggest that at the individual pixel level, the Calibrated Bayesian model is more accurate. However, at the larger collective level, the Color Convolution model produces the least combined error. This makes sense in terms of the inherent advantages of each of the models. Since the Calibrated Bayesian model is a Bayesian method it does well with a small sample of truth values as provided by the MacBeth chart. Running a Markov chain process enhances the interpretations that can be obtained from a small sample. On the other hand, the Color Convolution model is by design adept at looking at a larger patch of the image rather than a single pixel. Therefore, one would expect the differences in performance between the MSE and RE for the two
Overall, our two proposed algorithms outperform current algorithms and are both commercially and scientifically viable improvements to the color constancy literature. As would be expected, having additional information in the form of the MacBeth Color Checker allowed our methods to improve upon the previous, naive algorithms. The inclusion of some form of a calibration chart in images, or the image processing sequence, is necessary in progressing the field of color constancy.
BIBLIOGRAPHY


