SYDE 312 Applied Assignment: Image Color Gradient Selection Using Least Squares Fitting and Linear Transformations

Sammy Robens-Paradise, Nathan Oh

April 5th 2022

1 Abstract

There is no shortage of applications of linear algebra in the field of digital image handling. Applying fundamental concepts of linear algebra, we can extract insightful information about digital information otherwise unattainable by the human eye. The fitted linear color gradient of an image is no exception. This paper outlines the background and methodology of applying a series of linear transformations and least squares fitting to obtain the color gradient of an $n \times m$ pixel JPEG RGB digital image.

2 Background

2.1 Image Handling

Image handling is a concept used in computer science describing processes that allow humans to solve problems related to digital images. The scope of this paper is limited to digital image formats. The underlying process of image handling generally consists of an image file as an input. The output is usually another image with altered properties or characteristics of the original input. However, depending on the problem and desired solution the output can vary significantly. In many cases the output is a visual representation of a characteristic extracted from the input image.

Image processing can be broken down into three core steps. decoding, manipulation, and generation of the desired output [1]. Images can be represented using a grid collection of pixels; each mathematically consisting of a 1×3 vector of Red, Green and Blue (RGB) color values between 0 and 255. This is rooted in the human perception of color [2]. As a result, an image can be represented as a matrix of RGB values.

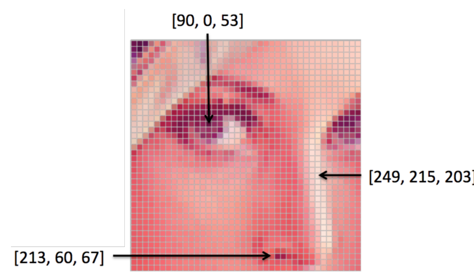

Figure 1: RGB Pixel Representation of an Image [Source: ai.stanford.edu, 2022]

This means altering an image equates to matrix operations performed on the RGB values of its pixels

2.2 Color Gradients

A color gradient (in computer graphics) is a set of colors within the colorspace who's position is defined by some mapping f. Strictly speaking, a linear color gradient is:

$$f : [p_{min}, p_{max}] \subset \mathbf{R} \to C \qquad (1)$$

defined by an RGB colorspace C, a sequence of sampling points $r_0 < ... < p_m \in [p_{min}, p_{max}]$, and a mapping f where $f(p_i) = c_i, i = 0, ..., m$. In this particular case, the mapping f in use is a linear mapping, shown in Figure 2.

[1] G. Anbarjafari, “1. Introduction to image processing,” Sisu @ UT. [Online]. Available: https://sisu.ut.ee/imageprocessing/book/1. [Accessed: 02-Apr-2022].

[2] Y.&C. Aktaş “A comprehensive guide to image processing: Fundamentals,” Medium, 02-Sep-2021. [Online]. Available: https://towardsdatascience.com/image-processing-4391c5bcef78. [Accessed: 02-Apr-2022].

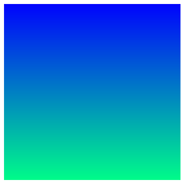

Figure 2: Blue-Green Gradient [Source: Wiki Commons, 2022]

3 Linear Algebra in Analysis

To determine the fitted color gradient of an $m \times n$ pixel image, a number of important algebraic tools were used. The first of which is the concept of linear transformations (linear maps). This was required to transform the color image C of a given RGB image from the three-dimensional $n \times m \times 3$ matrix to a set of three $n \times m$ matrices corresponding to the respective red, green and blue color maps.

The second algebraic tool implemented was the least-squares fit to a plane $Ac = b$ where A is: $\begin{pmatrix} 1 & X & Y \end{pmatrix}$, b is a $1 \times (n \times m)$ matrix of red, green or blue color magnitudes and c is a matrix of our desired coefficients. MATLAB's *reshape* function was also used to transform matrix entries into desirable formats for display and algebraic manipulation [3].

4 Case Study

There are many instances in which one would require insight into the characteristics of a digital image, particularly in the field of computer vision and graphics. However there are also relevant applications in physical and graphical design. One such application is that of interior design. During the process of interior design it is common practice to select artwork with analogous colors in accordance with color theory[4].

Quantifying dominant colors in an image is nearly impossible to the untrained human eye. As a result the generation or a least-squares fit color gradient associated with a specific image provides valuable insights for designers. Thus, improving their ability to select augmented colors according to a generated color gradient. For example, consider an interior designer who would like to select a painting to display in a room in accordance with augmented color selection principles. By taking a digital photo of the room and inputting it into the color gradient determining system, an associative color gradient could be generated, and used to select the desired painting, as shown in Figure 3 and 4.

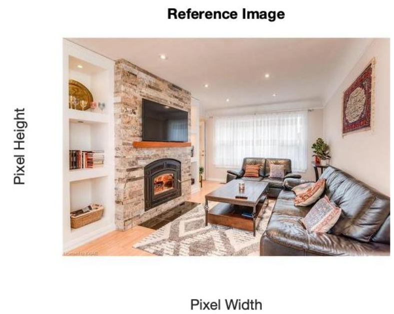

Figure 3: Reference Image [Source: SRP, 2022]

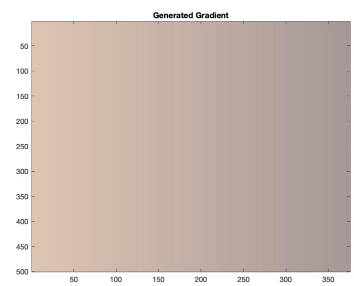

Figure 4: Generated Gradient [Source: SRP, 2022]

5 Mathematical Solution

To extract the least square fitted color gradient from a given image, a number of operations are performed. The first step is to decode the JPEG image from its lossy compression format to a colorspace matrix of RGB pixel values of size $n \times m \times 3$ where n is the pixel height of the image, m is the pixel width of the image, and 3 is the 3×1 vector of RGB values $\in [0, 255]$. The matrix was then normalized with respect to the maximum possible RGB value, 255, to ease analysis. The matrix representation of an image then looked like the following

$$C = \frac{1}{255} \begin{bmatrix} \begin{bmatrix} r_{1,1} \\ g_{1,1} \\ b_{1,1} \end{bmatrix} & \cdots & \cdots \\ \vdots & \ddots & \vdots \\ \vdots & \cdots & \begin{bmatrix} r_{n,m} \\ g_{n,m} \\ b_{n,m} \end{bmatrix} \end{bmatrix} \tag{2}$$

[3] reshape(A,sz) reshapes A using the size vector, sz, to define size(B)

[4] Analogous colors are any three colors which are side by side on a 12-part color wheel, such as yellow-green, yellow, and yellow-orange. Usually one of the three colors predominates. (www.colormatters.com, 2022)

At this point we still have a three dimensional matrix, making least squares fitting challenging. To remedy this, we can using a linear transformation to extract the red, green and blue values from the image colorspace into three $n \times m$ matrices. To do so we can apply the block multiplication linear transformation to the image matrix C as using the following transformation:

$$f : \mathbf{C}^{n \times m \times 3} \to \mathbf{C}^{n \times m} : c \mapsto Ac \tag{3}$$

A is an $n \times m$ matrix of 3×1 binary vectors $a_{n,m}$: $\begin{pmatrix} a_{n,m,1} & a_{n,m,2} & a_{n,m,3} \end{pmatrix}$. multiplying each element c in the image matrix $C_{n,m}$ by the binary matrices $a_{n,m}$ where $red : a = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$, $green : a = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}$, and $blue : a = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}$ gives us three $n \times m$ matrices consisting of the red, green and blue values for each pixel respectively. Figure 5 shows the Matrix magnitude surfaces of the red, green and blue matrices, and Figure 6 shows the isolated image representation the red value matrix associated with Figure 3.

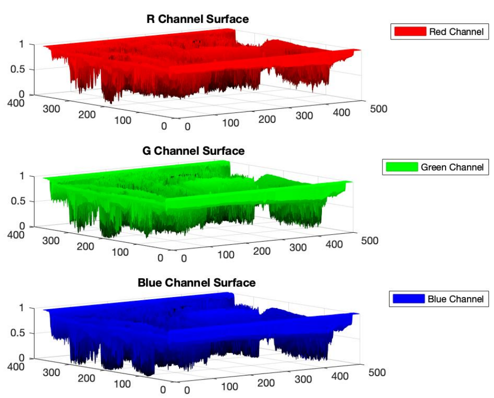

Figure 5: Isolated Red, Green and Blue Color Matrix Surfaces [Source: SRP, 2022]

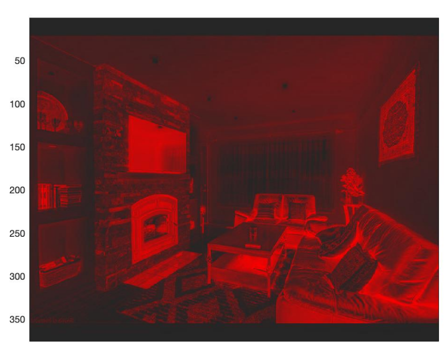

Figure 6: Isolated Red Channel of Figure 3 [Source: SRP, 2022]

Now that there are three $n \times m$ matrices we can apply the least-squares fit to each of the red, green and blue value matrices. In doing so, we can obtain the plane best fitted to each RGB matrix. When A represents a given R, G, or B matrix the solution to the least squares problem is provided by solving the over-determined system using *QR-factorization*:

$$A\bar{c} = \bar{b} = QR\bar{c} = b_e \tag{4}$$

$$\begin{bmatrix} 1 & x_1 & y_1 \\ \vdots & \vdots & \vdots \\ 1 & x_{n \times m} & y_{n \times m} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} z_1 \\ \vdots \\ z_{n \times m} \end{bmatrix} \tag{5}$$

The least-squares best fit plane is given as:

$$Z = A\bar{c} \tag{6}$$

Using MATLAB's *reshape* function we can then transform Z from the dimensions $(n \times m) \times 1$ to $n \times m$ so that it has the same dimensional characteristics as A. Figure 7 shows the the least squares fit to the blue color value matrix.

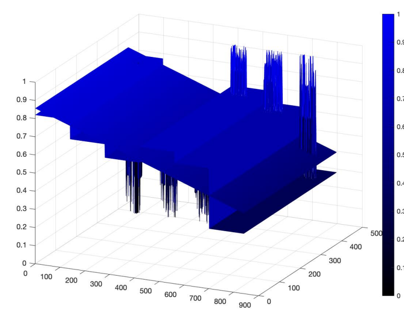

Figure 7: Normalized Least Squares Fit to Blue Color Value Matrix [Source: SRP, 2022]

At this point we now have three, $n \times m$ matrices Z_{red}, Z_{green} and Z_{blue} that are the best fit planes to the color matrices C_{red}, C_{green} and C_{blue}. To define our gradient, we want to get the maximum and minimum magnitudes of each plane Z. These represent the maximum and minimum magnitudes of red, green and blue on their respective planes

$$\begin{Bmatrix} r_{\max} = \max[Z_{red}] \in [0,1] \\ \vdots \end{Bmatrix} \tag{7}$$

$$\begin{Bmatrix} r_{\min} = \min[Z_{red}] \in [0,1] \\ \vdots \end{Bmatrix} \tag{8}$$

From our definition of a linear color gradient in (1) our linear map becomes

$$f : \left[\begin{bmatrix} r_{\min} \\ g_{\min} \\ b_{\min} \end{bmatrix}, \begin{bmatrix} r_{\max} \\ g_{\max} \\ b_{\max} \end{bmatrix} \right] \subset \mathbf{R} \to C \tag{9}$$

For each RGB value we can now construct an $1 \times m$ evenly spaced matrix using MATLAB's *linespace*[5] function where:

$$x_{red} = \begin{bmatrix} r_{\min_1} & \dots & r_{\max_m} \end{bmatrix}_{1 \times m} \tag{10}$$

$$x_{green} = \begin{bmatrix} g_{\min_1} & \dots & g_{\max_m} \end{bmatrix}_{1 \times m} \tag{11}$$

$$x_{blue} = \begin{bmatrix} b_{\min_1} & \dots & b_{\max_m} \end{bmatrix}_{1 \times m} \tag{12}$$

We can now construct a $3 \times m$ RGB matrix

$$C_{RGB} = \begin{bmatrix} \bar{x_{red}} \\ \bar{x_{green}} \\ \bar{x_{blue}} \end{bmatrix} \tag{13}$$

Performing element-wise block multiplication of C_{RGB} against every element in an $n \times m \times 3$ identity matrix $I_{n \times m \times 3}$ and then un-normalizing our RGB values by multiplying the resulting matrix by 255, we essentially *undo* the linear transformation performed in equation (3). This operation results in the follow color matrix representing the linear gradient of least squares fit of our inputted $n \times m$ JPEG image:

$$C_{grad} = \begin{bmatrix} \begin{bmatrix} r_{\min_{1,1}} \\ g_{\min_{1,1}} \\ b_{\min_{1,1}} \end{bmatrix} & \dots & \dots \\ \vdots & \ddots & \vdots \\ \vdots & \dots & \begin{bmatrix} r_{\max_{n,m}} \\ g_{\max_{n,m}} \\ b_{\max\, n,m} \end{bmatrix} \end{bmatrix} \tag{14}$$

The transformation is shown in Figures 8 and 9.

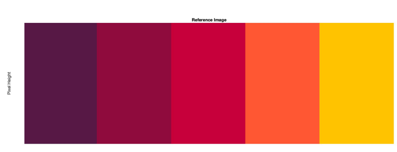

Figure 8: Pre-selected Analogous Color Theme Input Image [Source: SRP, 2022]

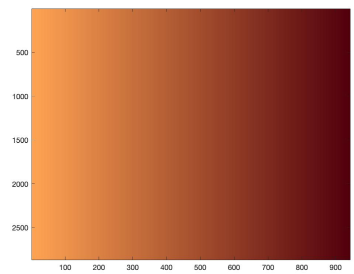

Figure 9: Generated Analogous Color Gradient [Source: SRP, 2022]

6 Result Analysis

The results of the color detection system could be improved by being more selective with the data points used when creating the resulting gradient. This is the result of utilizing regression opposed to interpolation as the gradient will closely resemble the most common data points opposed to representing the most extreme colors shown within the image. This is observed Figure 8 and Figure 9. The RGB values of the left and right most colors in Figure 8 are (87, 25, 69) and (255, 195, 0) respectively. The RGB values for the left and right most colors of Figure 9 are (250, 160, 80) and (82, 2, 13) respectively. Although the program is not meant to produce a gradient of extreme values, such values should have more weight or meaning as they tend to be points of interest in image detection problems.

7 Conclusion

The color detection program shown in this report accepts a JPEG image as an input and uses MATLAB to decode the image as a two dimensional matrix of RGB values. The RGB values of each pixel within the image are isolated based on color (Red, Green or Blue) and a fit of least squares method is used to find a plane of best fit for each color. The resulting planes are then used to produce a two-color gradient, representing the most common RGB values within the original input image. A system with such an ability has numerous applications in the filed of graphic, and interior design.

[5] $y = linspace(x_1, x_2, n)$ generates n points. The spacing between the points is $(x_2 - x_1)/(n-1)$.