Improving the Two Color Image Prior Bayesian Demosaicing Algorithm

by

Ke Zhou

Submitted in partial fulfillment of Honors Requirements
for the Computer Science Major
Dickinson College, 2009

Professor John MacCormick, Supervisor
Professor Timothy Wahls, Reader
Professor Richard Forrester, Reader

May 5th 2009

The Department of Mathematics and Computer Science at Dickinson College hereby accepts this senior honors thesis by Ke Zhou and awards departmental honors in Computer Science.

Prof. John MacCormick (Advisor)	Date
Prof. Timothy Wahls (Committee Member)	Date
Prof. Richard Forrester (Committee Member)	Date
Prof. Dave Richeson (Department Chair)	Date

Department of Mathematics and Computer Science
Dickinson College

May 2009

Abstract

Improving the Two Color Image Prior Bayesian Demosaicing Algorithm

by
Ke Zhou

Image demosaicing is the process of recovering truecolor images from raw data captured by color filter arrays. The research focuses on the improvement of a state-of-the-art demosaicing algorithm built upon the pivotal assumption that there exist two representative colors in an image neighborhood. Using a JAVA experimental framework, we have implemented the target algorithm and compared it with three alternative demosaicing schemes. The failure mode of the algorithms is identified and two forms of adjustment are proposed. Confirming important behaviors of the algorithm, the adjustments achieve modest improvement in demosaicing quality in selected image areas. Further research is necessary to confirm the statistical significance of such improvement, and to fully explore the adjustments' potentials.

Acknowledgments

I would like to express my greatest gratitude to Professor John MacCormick, Professor Timothy Wahls and Professor Richard Forrester for their generous input. This research would not have been possible without their knowledge and guidance.

I would also like to thank Professor G. D. Finlayson at the University of East Anglia, UK for introducing me to the exciting field of computer vision.

Finally, I would like to thank my parents for their understanding and support.

Table Of Contents

Chapter 1

Introduction and Background

1.1. Introduction to Image Demosaicing

RGB digital images are composed of pixels that store the values of red, green and blue color channels. When combined, the three color channels jointly represent a color vector in the RGB vector space.

Ideally, digital imaging devices should capture all three channels if accurate recording is to be achieved. But full channel coverage requires first splitting the incoming light beam into three parts based on wavelength, and then precisely projecting them to matching photosensors. The process is both technically challenging and economically infeasible. This is especially the case in consumer-class electronics (Gunturk et al., 2008).

To make the process tractable, most digital imaging devices choose to preserve only one color channel at each pixel location. The partial capture process essentially keeps only one third of the color information carried by the incoming light beam. As a result, the filtered data must be interpolated in some way to recover true-color images. This reconstruction of true-color images from raw sensor data is known as image demosaicing.

Image demosaicing underlies color image generation in digital image capture devices coming in almost any size and serving almost any purpose. From cell phones to surveillance cameras, improvements in demosaicing algorithms would immediately lead to better image quality, essentially making digital images better representations of the real world. Furthermore, because the enhancement occurs at the image capture stage, additional image operations that could benefit from improved color fidelity would be facilitated at no extra computational cost.

1.2 Project Objectives

The project has the following objectives:

i. To implement several previously published demosaicing algorithms, including the two color image prior method, and to conduct a comparative analysis of their performance,

ii. To investigate the effect of a configurable parameter η on the two color prior algorithm since the discussion of this parameter in the original paper is incomplete,

iii. To improve the two color prior algorithm.

1.3 Technical Details of Image Demosaicing

Red, green and blue are chosen as the channels in RGB digital images to mimic the human vision system's varying sensibility to different regions in the visible spectrum. Unlike highly sophisticated human eyes, however, photosensors lack the ability to distinguish wavelengths, rendering them incapable of recognizing colors. Color filters must be placed in front of sensors to allow the preservation of chromatically meaningful information.

Color filters are placed next to each other to form color filters mosaics, known as color filter arrays (CFA). It is apparent that how the arrangement of the three types of color filters in CFAs has major impacts on the way demosaicing algorithms work in that it dictates which channel is recorded at a pixel. There are several popular CFA patterns. Among them, the Bayer filter pattern is the most widely researched and commercialized one.

1.3.1 Bayer Filter Pattern

In the Bayer filter pattern (Figure 1.1), green filters are placed in quincunx grids, while red and blue filters occupy rectangular grids alternately (Gunturk et al., 2008). There are twice as many green filters as red or blue filters because the human vision system is more sensitive to green colors. All demosaicing algorithms surveyed in this research use the Bayer filter pattern. However, it should be noted that Bayesian demosaicing algorithms could be easily extended to other CFA patterns (Gunturk et al., 2008).

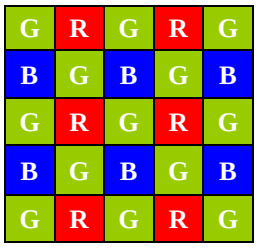


Figure 1.1: The Bayer color filter pattern (US Patent 3971065) is the most widely used and researched color filter pattern. Algorithms designed for the Bayer pattern can often adapt to other CFA patterns without must difficulty.

1.3.2 The Peak Signal-to-Noise Ratio

The peak signal-to-noise ratio (PSNR) is a simple yet limited way of measuring the quality of image reproduction. It is a single number metric and the unit of measurement is decibels (dB). It can be computed by

$$PSNR = 20\log_{10}\left(\frac{255}{MSE}\right) \tag{1}$$

As a transformation of mean square error (MSE), PSNR generally ranges from 20 to 40. Its results are reported to two decimal points. Higher MSE leads to lower PSNR, and vice versa. PSNR in its original form only calculates MSE on the luminance channel (green). Thus, its result does not lend itself well to the estimation of color distortion. To address the problem, this research uses a revised PSNR metric that calculates MSE as a mean of the MSEs on all three channels.

In this project, PSNR is the metric selected to measure the recovery quality of demosaicing algorithms and the efficiency of proposed improvements. It is worth noting that although PSNR is well suited for the purpose of comparison, its actual value does not convey much significance (Haskell et al., 1995).

1.4 Methodology

All experiments are performed using a JAVA framework developed for this project. The framework is composed of three components: I/O utilities, auxiliary algorithms and demosaicers. I/O utilities take care of parsing images into easily usable integer arrays and outputting demosacied images. All images are stored in the PNG format to avoid loss of image quality during I/O operations. Auxiliary algorithms provide support such as clustering, PSNR calculation, and edge detection. Demosaicers are implementations of different demosaicing algorithms. Because demosaicers extend a common superclass, the task of running experiments that encapsulate different demosaicing algorithms is greatly simplified.

Running time and PSNR are reported after each experiment completes. Other than general logging, the framework is also capable of rich data collection that will keep detailed, pixel-level information for selected image areas once activated.

Most analysis of demosaicing results happens in MATLAB. MATLAB provides a convenient environment for quick access to color channel values at pixels and serves as a great tool for visualizing intermediate results.

Samples images used in the experiment are 24 scaled-down images from the Kodak PhotoCD image set, the most widely used image set in the demosaicing literature. To simulate the single-channel data collected by pixelsensors, images from the set are filtered by an implementation of the Bayer filter. The demosaicers produce demosaicing results using

the Bayer pattern images as inputs. To evaluate each demosaicer's performance, demosaiced images are compared with the original, unfiltered images.

Chapter 2

Comparison of Selected Algorithms

2.1 Demosaicing Algorithms

Gunturk et al. (2008) suggest two ways of formulating the demosaicing problem. The first way is the statistical formulation in the spatial domain. Using Bayes' theorem to model posterior distribution, the majority of algorithms in this category accept the constant-hue assumption in which the inter-channel color difference function is characterized by its smoothness. That is, a color change in one channel is likely to be accompanied by similar changes in the other channels. The second way of approaching the demosaicing problem is the deterministic formulation in the frequency domain (Gunturk et al., 2008). Algorithms in this category treat CFA data as sub-samples of the full-resolution image, and missing channels are usually recovered through the application of convolution masks.

This section summarizes the details of four previously published algorithms. Bilinear Interpolation and High-Quality Linear Interpolation are examined as integrated components of the Two Color Image Prior algorithm, which this research sets out to improve. Vector Median Demosaicing algorithm is included for additional comparison because, like the two color prior model, it interpolates colors using fully specified RGB triples.

2.1.1 Bilinear Interpolation

Bilinear interpolation provides a basis for high-quality linear interpolation, which in turn supports the two color image prior model. It interpolates a missing channel by taking the averages of the closest neighbors of the same channel. For example, the green channel at a red or blue pixel can be estimated as

$$\hat{g}_{(i,j)} = \frac{1}{4}\sum\left(g_{(m,n)}\right) \tag{2}$$
$$(m,n) = \{(i-1,j-1),(i-1,j+1),(i+1,j-1),(i+1,j+1)\}$$

Bilinear interpolation is perhaps the most trivial demosaicing algorithm. It completely ignores inter-channel color consistency because each channel is estimated independent of other channels. For the same reason, it also ignores trends of color transition. The algorithm offers fast demosaicing speed but fairly poor image quality: images demosaiced by bilinear interpolation show noticeable false colors along edges and are blurred globally.

2.1.2 Vector Median Demosaicing

The vector median demosaicing algorithm is designed to address the zipper effect, a form of color distortion commonly found along the edges of reconstructed images. The algorithm first groups Bayer pattern pixels surrounding a target pixel into 16 "pseudo-pixels", each composed of a red pixel, a green pixel and a blue pixel. When the target pixel is a green pixel, 8 surrounding pixels are considered. Otherwise, 12 surrounding pixels are considered. The vector median, defined as the color vector that minimizes the sum of distance to all other vectors, is then selected as the interpolated color. The idea is that a vector median would adapt naturally to gradients and would avoid over-smoothing along edges (Figure 2.1).

		R1		
	B1	G1	B2	
R2	G2	R3	G3	R4
	B3	G4	B4	
		R5		

Figure 2.1: 16 pseudo-pixels can be constructed from these 12 pixels.

The 16 pseudo-pixels in the neighborhood are listed in equation (3).

$$V = \left\{ \begin{bmatrix} R1 \\ G1 \\ B1 \end{bmatrix}, \begin{bmatrix} R1 \\ G1 \\ B2 \end{bmatrix}, \begin{bmatrix} R4 \\ G3 \\ B2 \end{bmatrix}, \begin{bmatrix} R4 \\ G3 \\ B4 \end{bmatrix}, \begin{bmatrix} R5 \\ G4 \\ B4 \end{bmatrix}, \begin{bmatrix} R5 \\ G4 \\ B3 \end{bmatrix}, \begin{bmatrix} R2 \\ G2 \\ B3 \end{bmatrix}, \begin{bmatrix} R2 \\ G2 \\ B1 \end{bmatrix}, \begin{bmatrix} R3 \\ G1 \\ B2 \end{bmatrix}, \begin{bmatrix} R3 \\ G1 \\ B1 \end{bmatrix}, \begin{bmatrix} R3 \\ G2 \\ B1 \end{bmatrix}, \begin{bmatrix} R3 \\ G2 \\ B3 \end{bmatrix}, \begin{bmatrix} R3 \\ G4 \\ B4 \end{bmatrix}, \begin{bmatrix} R3 \\ G4 \\ B3 \end{bmatrix}, \begin{bmatrix} R3 \\ G3 \\ B2 \end{bmatrix}, \begin{bmatrix} R3 \\ G3 \\ B4 \end{bmatrix} \right\} \tag{3}$$

Let N be the number of pseudo pixels (N equals to 16 by definition) and K be the dimension of vectors (K equals to 3 for there are three color channels). The vector median is the vector $v^m \in V$ that solves

$$v^m = \underset{v^m}{\operatorname{argmin}} \sum_{i=1}^{N} \left(\sqrt{\sum_{k=1}^{K} \left(v_k^m - v_k^i \right)^2} \right) \tag{4}$$

Visual inspections of demosaicing results show that vector median demosaicing still suffer from the zipper effect along edges (Figure 2.2). A possible reason is that taking medians tends to oversharpen smooth edges. Vector median demosaicing is similar to the two color image prior model in that both schemes attempt to estimate a color based on fully specified RGB triples.

Figure 2.2: The zipper effect is still a serious problem in images demosaiced by the vector median method.

2.1.3 High-Quality Linear Interpolation

High-quality linear interpolation is introduced as an improvement to the simple bilinear interpolation (Malva et al., 2004). The main improvement of this algorithm is the

exploitation of the cross-channel color correlation using a region of support no larger than 5×5.

For example, a sharp difference between the red channel at a red pixel and the average of neighboring red channels suggests that there exists a gradient from the pixel to its neighbors. Because this inter-channel color trend is to be reflected, a portion of the difference will be calculated as the gradient correction, and added to, say, the green channel of the pixel as determined by bilinear interpolation. The following formulas show specifically how to estimate the green channel at a red pixel.

$$\hat{g}_{(i,j)} = \hat{g}^{B}_{(i,j)} + \alpha \Delta R_{(i,j)} \tag{5}$$

$$\Delta R_{(i,j)} = r_{(i,j)} - \frac{1}{4}\sum\left(r_{(m,n)}\right) \tag{6}$$
$$(m,n) = \{(i-1,j-1),(i-1,j+1),(i+1,j-1),(i+1,j+1)\}$$

$\hat{g}^{B}_{(i,j)}$ is the bilinearly interpolated green channel. $\Delta R_{(i,j)}$ is the red gradient as represented by the difference between the red channel of the pixel and the average of adjacent red channels. α is a gain parameter that controls how much of the gradient is applied to the bilinearly interpolated result. Because of filter distribution in Bayer filter pattern, green channels at blue pixels can be computed with the same formula.

The following formulas compute the red channel of a green pixel and the red channel of a blue pixel respectively.

$$\hat{r}_{(i,j)} = \hat{r}^{B}_{(i,j)} + \alpha \Delta G_{(i,j)},\ \hat{r}_{(i,j)} = \hat{r}^{B}_{(i,j)} + \alpha \Delta B_{(i,j)} \tag{7}$$

The way $\Delta G_{(i,j)}$ and $\Delta B_{(i,j)}$ are computed is described in Figure 2.3. $\Delta G_{(i,j)}$ is computed with information from 9 pixels. $\Delta B_{(i,j)}$ is computed with the support of 5 pixels.

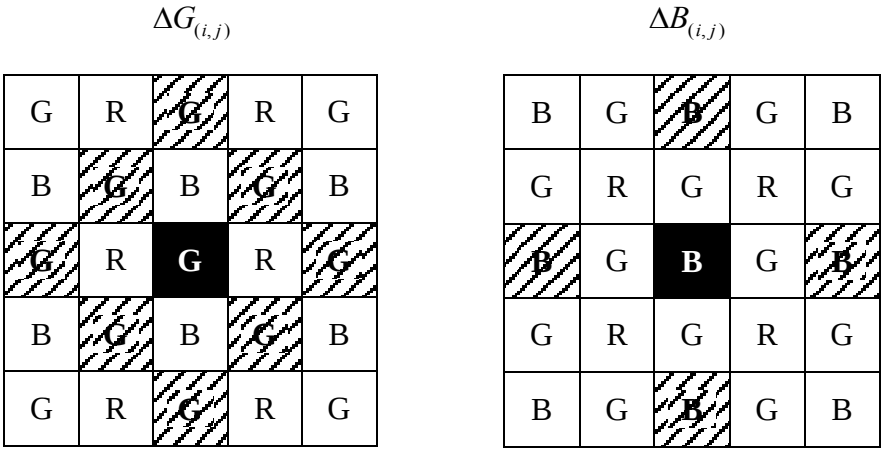


Figure 2.3: $\Delta G_{(i,j)}$and $\Delta B_{(i,j)}$are computed as the difference between the center channel and the average of shaded neighbors.

The blue channel of a green pixel and the blue channel of a red pixel can be computed similarly.

The demosaicing result of high-quality linear interpolation is significantly better than that of simple bilinear interpolation. This is expected because the algorithm internalizes inter-channel consistency, one aspect that bilinear interpolation completely ignores. In fact, the algorithm even out-performs several nonlinear demosaicers that are much more complex. The strong demosaicing result coupled with modest computational cost is what makes high-quality linear interpolation a suitable choice as the bootstrapping demosaicer for the two color image prior model.

2.1.4 The Two Color Image Prior Bayesian Demosaicing Algorithm

The two color image prior Bayesian demosaicing algorithm was introduced by Bennett et al. (2006). The algorithm belongs to the statistical category of demosaicing problem formulation. It estimates the color of a pixel as the linear combination of the two representative colors in the pixel's image neighborhood. To make color clustering operable, a preliminary step of demosaicing is necessary to transform Bayer samples into fully specified

RGB triples. It has been shown that an ideal bootstrapping demosaicer should preserve high frequency and have low tendency of color infidelity. In the original work by Bennett et al, the preliminary demosaicer selected is the high-quality linear interpolation method. Color clustering on the preliminarily demosaiced pixels is carried out with a weighted K-Means clustering that rejects outliers. After determining two representative colors, $\bar{J}$ and $\bar{K}$ for pixel *x*, the color *C* of *x can* be calculated as

$$C = (1-\alpha)\bar{J} + \alpha\bar{K} \tag{8}$$

where α is a blending factor. The convex combination implies that the interpolated color of *x* is a specific mixture of the two representative colors in the neighborhood. Let *i* be the index of the color channel at a pixel *x* and S_j^x be the observed color value that *x* holds. Then an intuitive way of calculating α at *x* would be

$$\alpha = \frac{S_i^x - \bar{J}_i}{\bar{K}_i - \bar{J}_i} \tag{9}$$

One problem with estimating α this way is the loss of accuracy when $\bar{J}$ and $\bar{K}$ are very close. To ameliorate the problem, a more robust form of the blending factor is calculated using Bayes' theorem. Given $\bar{J}$, $\bar{K}$ and S, the set of all color samples in *x*'s neighborhood, the most likely value of α, $\hat{\alpha}$, can be calculated as

$$\hat{\alpha} = \arg\max_{\alpha} P\left(\alpha \mid S, \bar{J}, \bar{K}\right) \tag{10}$$

Using Bayes' theorem, the problem can be transformed into

$$P\left(\alpha \mid S, \bar{J}, \bar{K}\right) = \frac{P(\alpha)P\left(S \mid \alpha, \bar{J}, \bar{K}\right)}{P(S)} \tag{11}$$

The equation states that the possibility of having a specific α given *S*, $\bar{J}$, and $\bar{K}$ is equal to the possibility of having this α ($P(\alpha)$) times the possibility of having this *S* given

such α, $\bar{J}$ and $\bar{K}$ ($P(S \mid \alpha, \bar{J}, \bar{K})$), divided by the possibility of having this S ($P(S)$). Assuming that $P(S)$ follows a uniform distribution. Equation (11) can be reduced to

$$P\left(\alpha \mid S, \bar{J}, \bar{K}\right) \propto P(\alpha) P\left(S \mid \alpha, \bar{J}, \bar{K}\right) \tag{12}$$

$P(S)$ is no longer relevant in (12) because it becomes a constant under the assumption of a uniform distribution.

Assuming an independent Gaussian distribution of neighboring colors, we could obtain an equation that calculates $P\left(S \mid \alpha, \bar{J}, \bar{K}\right)$. The same equation also determines α*, a blending factor that maximizes $P(S \mid \alpha, \bar{J}, \bar{K})$.

$P(\alpha)$ is biased to be 1 when α equals either 0 or 1 because empirical evidence suggests that at most pixels, α* equals either 0 or 1. That is, at most pixels, the demosaicer decides that the best behavior is to set the color to be one of the representative colors rather than the linear combination result of the two. For example, the color C equals $\bar{J}$ if $\alpha = 0$ and $\bar{K}$ if $\alpha = 1$.

$P(\alpha)$ equals η when α is neither 0 nor 1. Although the exact value of η is not disclosed, the original work by Bennett et al. mention that $\eta < 1$. The parameter is essentially a multiplier that discounts the possibility of choosing α* as the blending factor. Thus, η determines how much image sharpening is applied. The effect of different η values is discussed at great length in the next chapter.

To summarize, $P(\alpha)$ is flat at η except when α equals to 0 and 1 (Figure 2.4). To find $\hat{\alpha}$, the optimal blending factor, we only need to compare $P(0)P(S \mid 0, \bar{J}, \bar{K})$, $P(1)P(S \mid 1, \bar{J}, \bar{K})$, and $P(\alpha^*)P(S \mid \alpha^*, \bar{J}, \bar{K})$. $\hat{\alpha}$ takes the value of whichever one of $\alpha = 0$, $\alpha = 1$ and $\alpha = \alpha$* that maximizes $P(\alpha)P\left(S \mid \alpha, \bar{J}, \bar{K}\right)$.

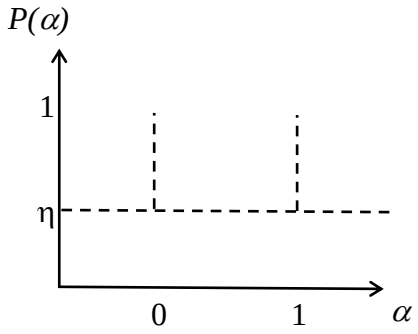


Figure 2.4: $P(\alpha)$ has two impulses: one occurs at α =0 and the other at α =1.

2.2 Comparing the Algorithms

Comparing the two color image prior demosaicing algorithm with other relevant interpolation methods helps demonstrate each scheme's strengths and weaknesses. It could also verify whether the algorithms are implemented correctly. The two color image prior model is expected to produce images of the highest quality. Both high-quality linear interpolation and vector median demosaicing should perform better than bilinear interpolation.

Table 2.1: Average PSNRs (dB) Using the Scaled Kodak PhotoCD Image Set

Bilinear	Vector Median	High Quality	Color Prior
24.87	24.95	28.35	30.09

Demosaicing results are summarized in Table 2.1. The result clearly indicates that the two color image prior model consistently produce images with the highest quality. High-quality linear interpolation performs better than bilinear interpolation as expected. On average, the PSNR of the two color prior model is 1.74 dB higher than that of high-quality linear interpolation. The complete list of PSNRs on each image is available in Appendix A.

Figure 2.5 shows demosaicing results produced by the four algorithms on the same area of an image. As expected, bilinear interpolation creates severe false colors along all edges and blurs the image as a whole. Based on the result from bilinear interpolation, high quality linear interpolation partially reduces color infidelity by applying gradient corrections.

Using high quality linear interpolation as the bootstrapping demosaicer, the two color prior model further eliminates the jagged pattern and false colors along edges and produces the best recovery quality. An unexpected result is the relatively poor performance of the vector median demosaicing algorithm. Its recovery result shows the demosaicer's strong tendency to oversharpen images in smooth areas, counteracting any advantages gained at sharp edges. Furthermore, although color artifacts are heavily reduced along sharp edges, the resulting edges are jagged rather than continuous.

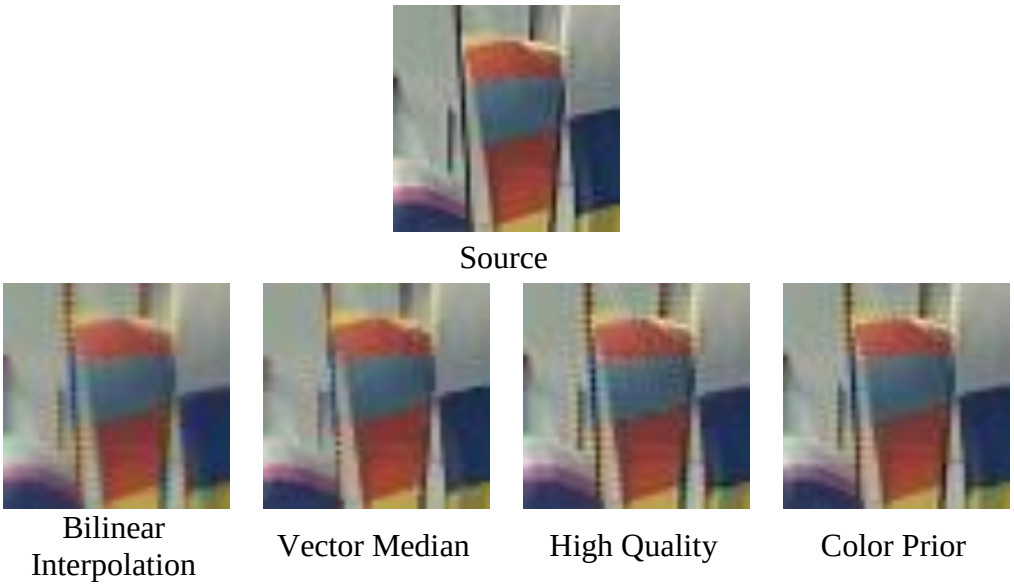


Figure 2.5: Demosaicing results provide visual confirmation of each algorithm's behaviors. Recovery quality increases sequentially as image data are passed from Bilinear Interpolation to the Two Color Image Prior model through High Quality Linear Interpolation. Although vector median demosaicing reduces the presence of color fringes, it oversharpens gradients and produces jagged edges.

Chapter 3

Improving the Two Color Image Prior Model

3.1 The Failure Mode of the Two Color Image Prior Model

Results from the comparative analysis confirm that the two color image prior model is consistently the top performer among the four algorithms. To further improve the algorithm, the first step would be to investigate image areas where reconstruction quality is less than satisfying. The algorithm's mode of failure could point to the model's weakness and reveal possible ways of improving it.

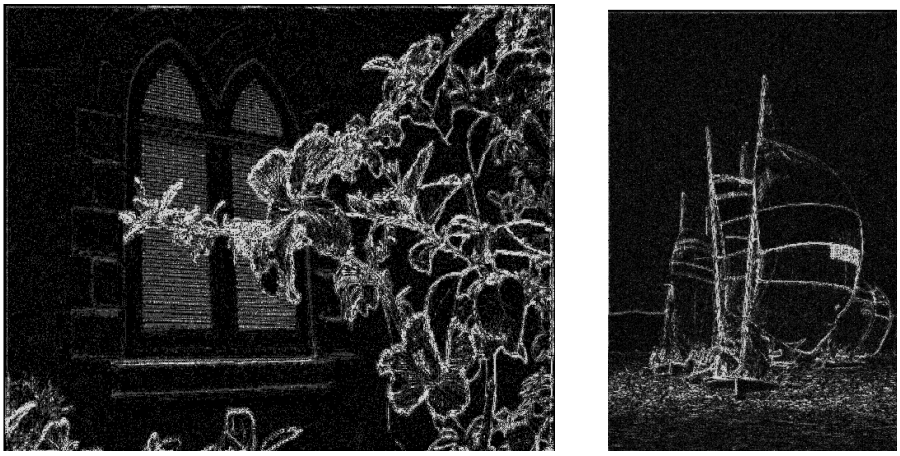


Figure 3.1: Bright spots in the figures correspond to the difference between the source image and the image recovered by the two color image prior model.

A simple way of identifying image areas where a demosaicer performs weakly is to measure color discrepancies between demosaicing results and original images pixel by pixel. Figure 3.1 shows the difference between demosaicing results of the two color image prior model and their sources. If the demosaicer is of perfect precision, that is, if it produces no lost of color accuracy at all, pixels in Figure 3.1 should all be black as there would be no difference between results and originals. Otherwise, the figure will contain bright spots

signifying color infidelity. The degree of brightness at a pixel is proportional to the distance between the interpolated color and the real color at that pixel location.

In Figure 3.1, we see a concentration of bright pixels along edges. This observation suggests that the performance of the two color image prior model deteriorates when the algorithm interpolates colors along edges.

(a) (b) (c)

Figure 3.2: In (a), pixels at which the color prior algorithm does not combine the two representative colors are marked with purple. Notice that the purple marker disappears when a pixel is about 5 pixels away from any strong edge. This suggests that the algorithm switches to linear combination when approaching edges. (b) is a visualization of the distance between the two color centroids in each neighborhood. Greater brightness indicates greater distance. (c) shows the difference between the original image area and its demosaicing result. Brighter spots suggest greater errors.

The drop in recovery quality is attributable to the fact that the algorithm has to contrive a blending factor for two very different colors when processing edge pixels. In Figure 3.2.a, areas of an image where linear combination is not triggered are marked purple. The blending factor used in those areas is either 1 or 0 because the algorithm decides that there is only one representative color. The purple color disappears along edges and within most leaves. This means that the algorithm starts to blend colors in those areas. When an edge appears in the 5×5 demosaicing window, the algorithm will immediately notice the existence of two distinct color centroids, each of which represents a color cluster on one side of the edge, and respond by combining the two representative colors using the blending factor α^*. Because the distances between the two color centroids are large in edge pixels (Figure 3.2.b), linear combination will be prone to error (Figure 3.2.c). The algorithm also

linearly combines representative colors within leaves where gradients are detected. Color infidelity is less of a problem in such areas because the distance between centroids in gradients is quite small, allowing less space for error.

3.2 The Effect of the Parameter η

A full understanding of the two color prior algorithm requires the study of an important parameter of the model. η is the value $P(\alpha)$ takes when α is equal to neither 0 nor 1. It serves as a multiplier with which the model discounts the possibility of selecting α^* as the blending factor. The original work by Bennett et al. mentions that as a discount factor, $\eta < 1$, and when the image is noisy, $\eta \approx 1$. However, the effect of giving η different values is not discussed in depth. The goal of this section is to investigate how different η values impact the demosaicing quality of the two color prior model.

Because the linear combination of the two color prior happens mostly along edges and in gradients, η has a direct impact on how edge and gradient pixels are handled in the model. It is reasonable to assume that lower η values adapt better to sharp edges as they suppress the mixture of color responsible for the creation of false colors. By similar reasoning, higher η values should be favorable for image areas where color blending is desirable. Such areas include blur edges and gradients. To verify the assumption, several trials of two color prior demosaicing are performed on three selected image areas that feature sharp edges, gradients and random patterns respectively. Each trial sets η to a gradually increasing value. When $\eta = 0$, linear combination never occurs: all pixels accept one of the two representative colors. When $\eta = 0.5$, the likelihood of accepting the mixed color is discounted by half. When $\eta = 1$, the likelihood is not discounted.

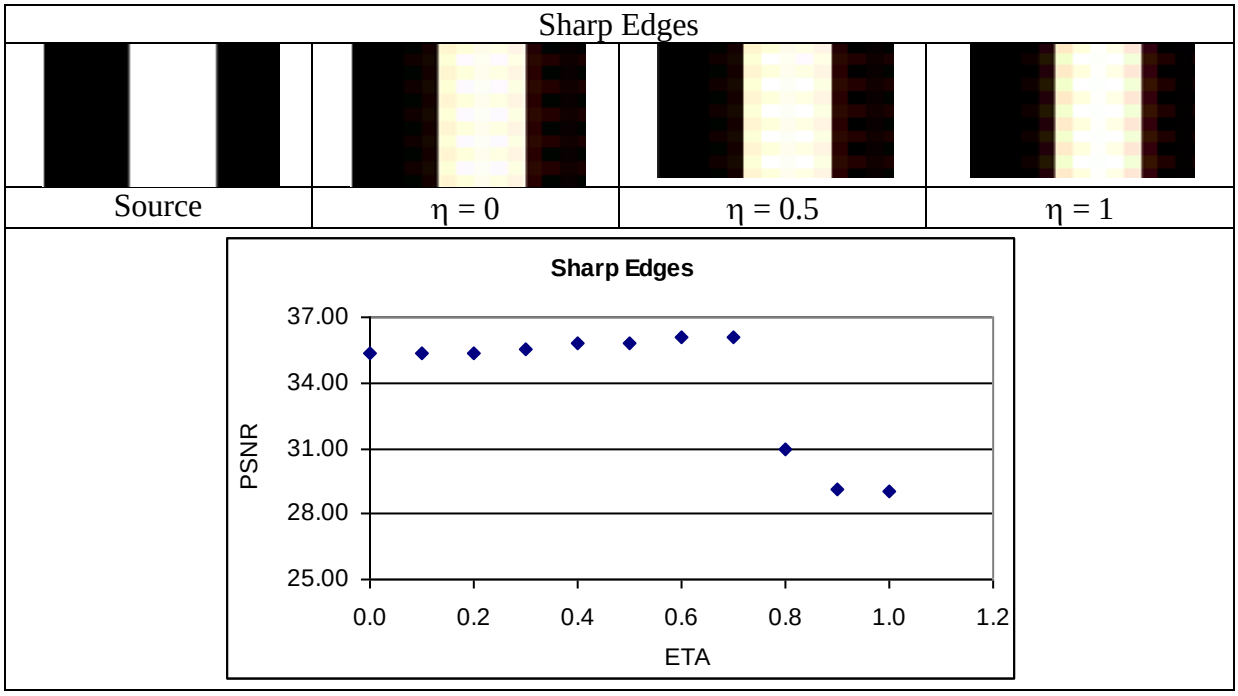


Figure 3.3: η values higher than a threshold around 0.7 lead to worse results in image areas with sharp edges.

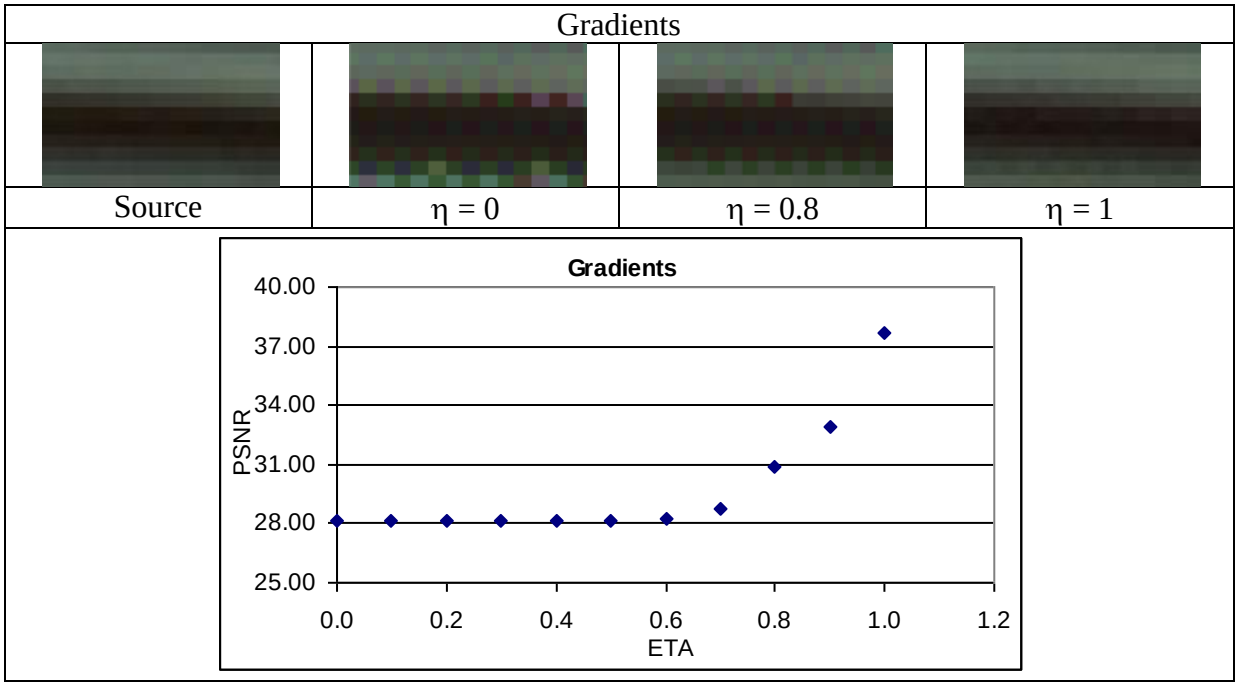


Figure 3.4: Higher η values handle gradients better.

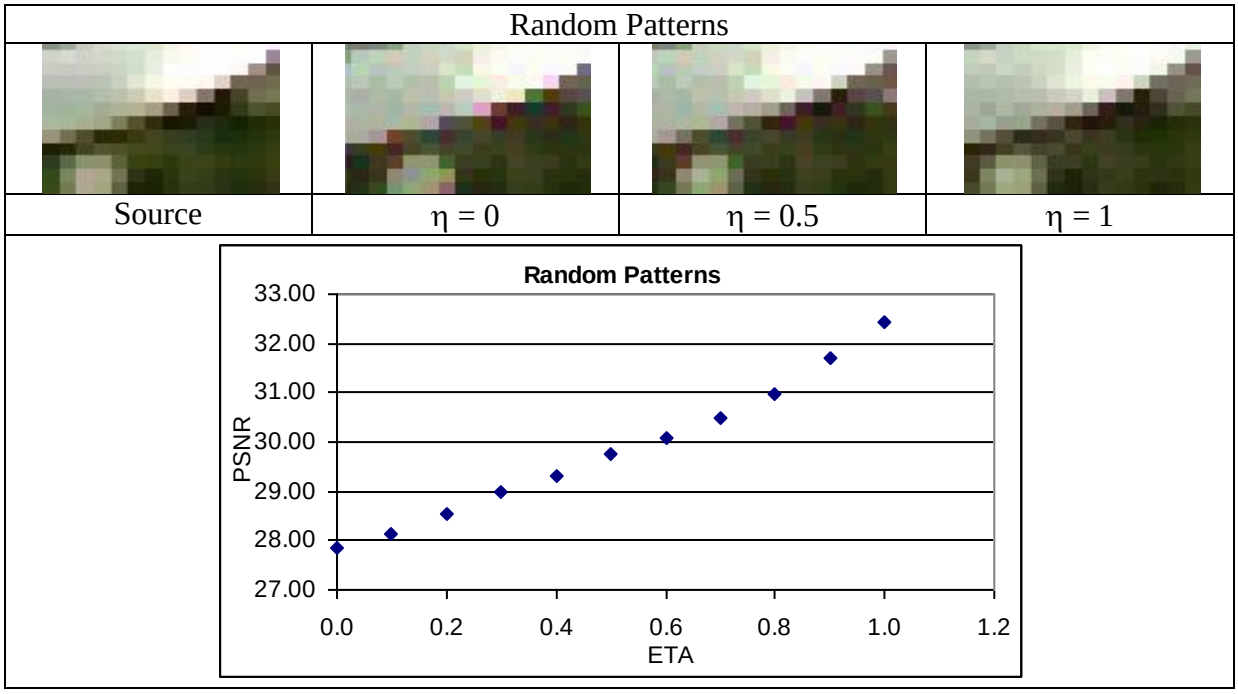


Figure 3.5: The η value is positively related to the demosaicing quality of random patterns typical of real-world images.

As Figure 3.3 illustrates, lower η values indeed handle sharp edges better than higher η values do. The PSNR of the image containing sharp edges drops significantly as η approaches 1. An increased number of pixels with false color can be observed along the edges, which is clearly the result of undesirable color blending.

Also conforming to the assumptions, higher η values do interpolate gradients with better accuracy as shown in Figure 3.4. Higher η values steadily improve the PSNR of the image featuring repeated gradients. Visual examination reveals that the edges look less jagged when η values are close to 1.

Finally, higher η values clearly outperform lower ones when the image area contains random patterns that are commonly found in real world images (Figure 3.5). The PSNR of the image shows a clear trend towards improvement as η increases. In addition, the

demosaicing result of the image containing random patterns is much more visually appealing when η is 1.

As a conclusion, the result supports the assumption that higher η values are desirable in image areas with gradients, and lower η values are better suited for sharp edges. The result also suggests that higher η values yield better results in typical images. But this finding is subject to further verification.

3.3 Two Forms of Adjustment

The failure mode of the color prior algorithm directs improvement efforts to areas with strong edge presence. Any adjustment that enhances the color prior model's reliability in interpolating edge pixels while maintaining performance in other areas will lead to better image quality. There are two paths to be explored based on the model's failure mode. First, it can be hypothesized that colors on edges are almost identical to one of the neighborhood's two color centroids and irrelevant to the other. This hypothesis essentially states that because of the rapid color transition on sharp edges, the optimal behavior would be to take the value of the closely related representative color. Any color combination will make the result less accurate. A second approach exploits the same central assumption that linear combination ignores fringe pixels' close association with one side of the edge. But it differs from the first method in that it allows for some degree of edge gradients. Instead of eliminating linear combination, it employs a post-combination correction that nudges interpolation results towards the cluster that is determined to be more relevant.

3.3.1 Adjust by Dynamically Setting η

Based on the observed characteristics of the parameter η, the first form of adjustment sets the values of η according to the strength of edge presence in image areas. When edge presence is strong in a neighborhood, η will be set to a low value that heavily discounts the

possibility of color blending. The interpolated color will take the value of one of the two representative colors. On the other hand, when edge presence is weak, η will be set to a high value that is very close or equal to 1. Color blending will not be discouraged in this case because the linear combination of two representative colors is observed to improve demosaicing results in neighborhoods without strong edges, such as gradients and image areas with random patterns.

The adjustment's edge-sensitive nature gives rise to the need for a way of measuring edge strength in images. It is worth noticing that the adjustment does not benefit from extra information such as the continuality and directions of edges, which are the focus of many high-level edge detectors. It merely requires a numeric index that measures the strength of edge presence. To that end, the Sobel edge detector is an appropriate choice due to its simplicity (Green, 2002). Using matrix convolution techniques detailed in Appendix B, the Sobel edge detector provides the adjusted algorithm with an edge intensity map with which pixel-level edge strength can be readily accessed.

The color prior algorithm using this form of adjustment consults the edge intensity map before it decides on a blending factor. If the edge intensity map value at a pixel exceeds a pre-determined threshold, η will be set to 0.1. Otherwise, η is set to 1. If the threshold is too high, no pixels will be classified as strong edges. In that case, the adjusted algorithm will not treat any pixel with a lowered η, behaving identically to the original algorithm. On the other hand, if the threshold is set too low, nearly all pixels will be considered as edges. Treating too many pixels with a lowered η would most likely lower demosaicing quality because necessary color blending in gradients will also be suppressed. There should be an ideal threshold value that preserves beneficial color combinations in gradients and suppress fallible ones along edges. One of the goals of the experiment will is to seek such a balance.

3.3.2 Adjust by Applying Post-Combination Correction

The second form of adjustment is a middle ground between the dynamic η adjustment and the original color prior model. While it acknowledges that blending colors along edges could generate unfavorable false colors, it does not eliminate linear combination. Instead, the adjustment compensates linear combination results with corrections that counteract estimated errors.

To simplify subsequent notation, let us define

$$F(\alpha) = P(\alpha)P(S \mid \alpha, \bar{J}, \bar{K}) \tag{13}$$

The rationale behind the post-combination correction adjustment then lies in the relationship among $F(0)$, $F(1)$, and $F(\alpha^*)$. For a pixel on or adjacent to a sharp edge, the dominant term of the three possibilities is obviously $F(\alpha^*)$, as the algorithm will detect two neatly separated color clusters, one for each side of the edge. More importantly, empirical evidence suggests that for such edge pixels, one of the remaining two possibilities, *F(0)* and *F(1)*, is usually very close to 0, while the other very close to $F(\alpha^*)$.

This relationship between *F(0)* and *F(1)* is ignored by the original algorithm, which is only concerned with the value of α^*. The post-combination correction adjustment, however, considers reoccurring patterns among *F(0)*, *F(1)* and $F(\alpha^*)$ as crucial information. Assume that for a pixel *x* situated on a strong edge, both $F(\alpha^*)$ and *F(1)* are greater than 90%, while *F(0)* is close to 0. The central assumption of the post-combination correction adjustment states that although blending the two representative colors is statistically optimal, the pixel's real color is actually much closer to the representative color obtainable by setting the blending factor to 1 (let us denote it *K*), and is almost unrelated to the representative color

obtainable by setting the blending factor to 0 (let us denote it *J*). To illustrate the reasoning of the assumption within the context of an image area, because *F(1)* is very close to $F(\alpha^*)$, *K* is thought to represent colors on the same side of the edge as *x*, and because *F(0)* is very close to 0, *J* is thought to represent colors on the other side of the edge, which are most likely to be irrelevant to the color at *x*. This assumption implies that the interpolation of *x* can be improved if the result of linear combination is corrected towards *K*.

Utilizing *F(0)*, *F(1)* and $F(\alpha^*)$ as the input for correction calculation is an open-ended task. However, there are some general rules that should be followed. First of all, the larger one of *F(0)* and *F(1)* must be sufficiently close to $F(\alpha^*)$ to signal a particularly relevant cluster. Secondly, the smaller one of *F(0)* and *F(1)* must be significantly smaller than $F(\alpha^*)$ to fit the role of an irrelevant color centroid. A correction cannot be justified unless both rules are met.

Let *F(M)* be the larger one of *F(0)* and *F(1)* and let *F(N)* be the smaller one. A simple way to the closeness between $F(\alpha^*)$ and *F(M)* would be the difference ratio, denoted as θ, which is defined as

$$\theta = \frac{\left|F(\alpha^*) - F(M)\right|}{F(\alpha^*)} \tag{14}$$

The lower θ is, the closer *F(M)* is to $F(\alpha^*)$.

Whether *F(N)* is sufficiently smaller than $F(\alpha^*)$ can be measured by ζ, which is defined as

$$\zeta = \frac{MEAN - F(N)}{MEAN} \tag{15}$$

$$MEAN = \frac{\left(F(\alpha^*) + F(M)\right)}{2} \tag{16}$$

The larger ζ is, the further *F(N)* is from the average of *F(M)* and $F(\alpha^*)$.

If a pixel's θ value is sufficiently high and its ζ value is sufficiently low, the pixel will be considered as qualified for correction. Like θ and ζ, the correction can be implemented in a wide array of ways. Potentially, any implementation that shifts the qualified pixel's linear combination result towards the more relevant representative color could be reasonable.

Let C be the color at *x* before correction and let $\hat{C}$ be the color after correction. $\hat{C}$ can be derived from C by another linear combination,

$$\hat{C} = (1-\beta)C + \beta M \tag{17}$$

M is the representative color whose chance of being selected is the larger one of *F(0)* and *F(1)*. β can be calculated by the formula

$$\beta = \frac{F(M)}{F(M) + F(\alpha^*)} \tag{18}$$

The definition of β suggests that a larger *F(M)* will result in a larger correction from *C* to *M*.

3.4 Results and Analysis

Adjusted versions of the color prior demosaicing algorithms are tested with the scaled Kodak PhotoCD image set. For the dynamic η adjustment, different edge strength thresholds are tested to seek an optimal lower bound. Empirically set thresholds are used in the post-combination correction adjustment Results from the experiments are reported in the following sections.

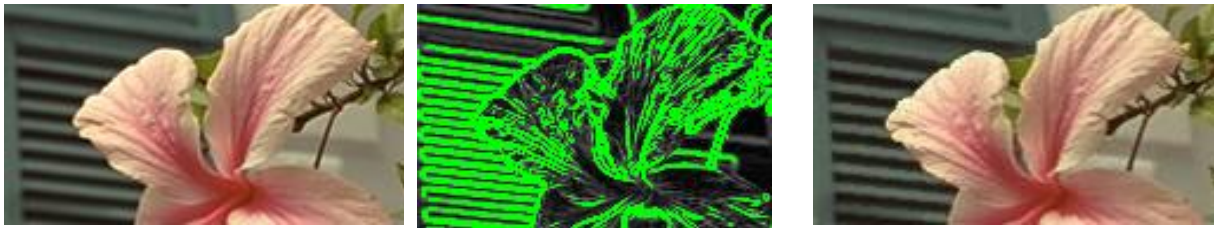
3.4.1 Results of Dynamically Setting η

Setting η dynamically in response to edge presence as measured by the Sobel edge detector fails to improve demosaicing quality in most images. Contrary to our assumption, extensive experiments show that lower η in areas with strong edge presence lead to lower PSNRs.

Table 3.1: Average PSNRs of the Kodak PhotoCD Image Set Using the Dynamic η Adjustment

	CP	Adjusted Color Prior η = 0.1 Edge Threshold = 500
Average PSNR	30.19	29.92

Table 3.1 summarizes results of applying the adjusted color prior algorithm on the scaled Kodak PhotoCD image set. PSNRs reported are lower than those for the original algorithm. A full list of results can be found in Appendix A.



Source Image | Edge Image
Edge threshold = 100 | Adjusted Color Prior
η = 0.1
Edge threshold = 100

(a)

Source Image | Edge Image
Edge threshold = 100 | Adjusted Color Prior
η = 0
Edge threshold = 100

(b)

Figure 3.6: (a) A low η value and a low edge threshold result in jagged edges through oversharpening. This configuration of the dynamic η adjustment aggravates color infidelity when edges are not sharp enough. (b) A lower η value does improve the quality of artificially constructed sharp strips.

Edges in images demosaiced by the adjusted algorithm are visibly crisper than those in the original images (Figure 3.6). This is expected because edges pixels are forced to choose one of the two representative colors. However, it seems that the sharpening effect is

too strong: the demosaiced images show clear signs of jagged edges and false colors along edges. Instead of improving color prediction, the dynamic η adjustment ends up increasing color infidelity in edge pixels. The overall deterioration of the PSNR is thus a result of the oversharpening of edges.

A possible explanation for the failure of the dynamic η adjustment is an improper edge strength threshold. The reasoning goes that if the threshold is too low, the algorithm will wrongly suppress linear combination of colors on edges that are composed of gradients. This results in undesirable sharpening of weak edges that brings down demosaicing quality. A proper threshold should be high enough to limit sharpening to truly strong edges, but not too high to ensure that all strong edges are sharpened.

Table 3.1: The Effect of the Edge Threshold on the Dynamic η Adjustment

	PSNR (dB) CP (η=0.1)		
Threshold	Image 1	Image 3	Image 7
50	26.20	31.65	27.84
300	27.46	32.24	29.29
500	27.66	32.43	29.82

The results in Table 3.2 (see Appendix A for the complete list) repudiate the existence of such a proper threshold. The table lists the result of running the adjusted algorithm on a sample image with an increasing edge strength threshold. Although the PSNR improves as the threshold increases, it never goes beyond the value reached by the unadjusted algorithm. It suggests that a higher threshold improves PSNR by reducing the number of pixels treated with the lower η. If a proper threshold exists, the PSNR should first rise above the value reached by the unadjusted algorithm as the threshold approaches its ideal value, and then drops back as the threshold increases further.

However, the dynamic η adjustment does improve demosaicing quality when the image contains artificially created sharp strips. In Figure 3.6.b, the PSNR goes from

22.88(dB) to 24.88(dB) as the demosaicer lowers η to 0 along strong edges. The improvement proves that the basic concept behind the adjustment is a reasonable one.

As a conclusion, the dynamic η adjustment to the two color prior model will not improve demosaicing quality of real world images. Even the sharpest edges in most real world images contain gradients that render them unsuitable for sharpening. Although linear combination along edges is prone to error due to the relatively large difference between the two representative colors in those areas, forcing edge pixels to pick one or another results in even more severe errors that show up as jagged edges and false colors. This finding confirms $\eta \approx 1$ as the optimal value for most images (Bennett et al., 2006).

3.4.2 Results of Applying Post-Combination Correction

This experiment chooses 10% as the upper bound for θ and 55% as the lower bound for ζ. The values are obtained empirically. The adjusted algorithm achieves an average improvement of 0.05 (dB) in PSNR when tested with the scaled Kodak PhotoCD image set (see Appendix A).

However, such an improvement is too small to be considered statistically significant. The following investigation reveals that the assumption behind the post-combination correction adjustment does not always hold. In other words, applying correction to a pixel that fits the edge pixel profile as determined by θ and ζ may shift the color further from the real value.

An area of a Kodak PhotoCD image is examined closely in Figure 3.7. Color-marked spots of the indexed image on the left correspond to corrected pixels in the image on the right. It is easy to see that such pixels concentrate along edges where the color changes rapidly. The observation confirms that the definitions of θ and ζ can, at least to some degree, correctly identify pixels on strong edges. In the indexed image, warm pixels signify

improvement over the original algorithm while cold pixels signify deterioration. There are 162 warm pixels and 85 cold pixels in the indexed image, suggesting that there is a large chance for post-combination correction to impair demosaicing quality. Overall, the adjustment to the algorithm improves the image area's PSNR from 28.68 (dB) to 28.77 (dB).

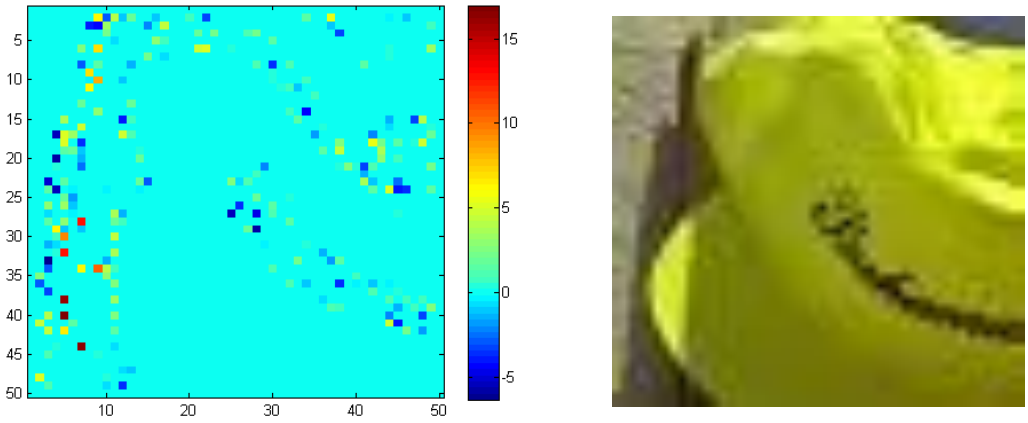


Figure 3.7: Warm pixels are locations where color accuracy is improved. Cold pixels are where applying post-combination correction harms color fidelity. There are 162 warm pixels and 85 cold pixels in the indexed image.

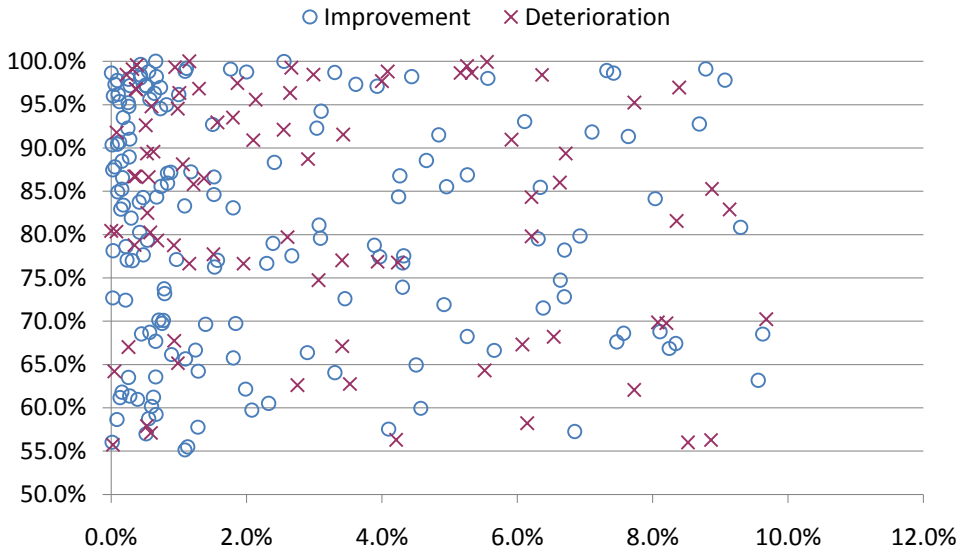


Figure 3.8: There is no obvious correlation between (θ, ζ) and the effect of applying post-combination correction.

A question naturally arises: is it possible to somehow identify and exclude pixels where the post-combination correction does no good? Figure 3.8 plots the corrected pixels in Figure 3.8 using their θ as the x coordinates and ζ as the y coordinates. If a pixel's color is improved after correction, it is plotted as a circle. A pixel is plotted as a cross if color deteriorates after correction. Since there is no easily discernable pattern in Figure 3.8 that separates crosses from circles, we conclude that θ and ζ do not provide enough information to ensure that all correction will bring color improvement in this image area. Therefore, the PSNR improvement in this case is indeterminate.

Chapter 4

Conclusion and Future Work

4.1 Conclusion

The project proposes two forms of adjustment to the color prior model. Both adjustments focus on edge pixels, as the failure mode analysis of the original algorithm reveals that the original algorithm produces most errors along edges.

The dynamic η approach lowers the value of η when the edge presence is strong. Experiment results show that the adjustment overestimates the sharpness of edges in real world images and aggravates color infidelity along gradients. The findings confirm that linear combination of representative colors should not be suppressed in most cases. The post-combination correction adjustment achieves modest quality gain on test images. But the improvement is modest. Closer inspection of a selected image area shows that the edge pixel profile used by the adjustment is not always reliable. Moreover, the way in which the adjustment defines its two metrics is decidedly naïve.

Although neither of the proposed adjustments significantly improves the original two color prior demosaicing algorithm using the Kodak PhotoCD image database, they do manage to obtain better demosaicing quality on selected image areas respectively. Future research on the characteristics of such image areas will provide possible paths to enhance the original algorithm.

4.2 Future Work

We assume that an edge is either weak or strong in the dynamic η adjustment and that η is lowered to a fixed value once the edge strength threshold is exceeded. A better way to implement the adjustment may be modeling a relationship between η and the edge strength.

Under the relationship, the edge strength's increase will be translated into a decrease in η and vice versa.

The efficiency of the post-combination correction adjustment is greatly limited by the fact that it is difficult to distinguish pixels that the correction improves from those hurt by the correction. Namely, we fail to see a pattern in the θ vs. ζ plot that could help us separate the two groups. Future research effort could focus on the use of unsupervised machine learning technique to look for possible structures of the pixel distribution in Figure 3.8 (Kapah et. al., 2000).

Appendix A

Tables

Table A.1: PSNRs of the Kodak PhotoCD Image Set

	PSNR (dB)			
Image	BI	VM	HQ	CP
1	22.68	22.84	26.19	27.68
2	27.81	27.81	28.94	29.43
3	28.57	29.25	31.08	32.43
4	26.54	26.46	29.16	30.40
5	21.04	20.72	25.54	27.22
6	24.21	24.16	27.61	29.72
7	24.11	24.33	28.12	29.91
8	19.36	19.90	23.62	25.38
9	25.33	25.91	28.95	30.75
10	26.26	26.55	29.90	31.87
11	25.27	25.65	29.08	30.95
12	25.77	26.07	28.58	29.54
13	21.51	20.83	25.67	27.99
14	23.63	23.33	26.09	26.82
15	24.72	24.90	27.72	29.16
16	28.84	29.05	32.50	35.00
17	25.71	25.83	29.84	32.12
18	24.26	23.53	28.30	30.26
19	23.85	24.28	27.74	29.49
20	25.49	26.51	29.50	31.88
21	24.27	24.49	28.60	31.31
22	26.44	26.12	29.49	30.80
23	27.33	26.67	30.22	31.74
24	23.82	23.57	27.97	30.36
Average	24.87	24.95	28.35	30.09

Table A.2: PSNRs of the Kodak PhotoCD Image Set Using the Dynamic η Adjustment With a Fixed Edge Threshold

	PSNR (dB)	
Image	CP	Adjusted CP η = 0.1 Edge Threshold = 500
1	27.68	27.66
2	29.43	29.40
3	32.43	32.43
4	30.40	30.31
5	27.22	26.87
6	29.72	29.67
7	29.91	29.82
8	25.38	25.11
9	30.75	30.57
10	31.87	31.77
11	30.95	30.83
12	29.54	29.42
13	27.99	27.89
14	26.82	26.80
15	29.16	28.93
16	35.00	34.99
17	32.12	32.04
18	30.26	30.08
19	29.49	29.17
20	31.88	31.12
21	31.31	30.98
22	30.80	30.77
23	31.74	31.50
24	30.36	30.04
Average	30.09	29.92

Table A.3: PSNRs of the Kodak PhotoCD Image Set Using the Dynamic η Adjustment With a Varying Edge Threshold

	PSNR (dB) CP (η=0.1)		
Threshold	Image 1	Image 3	Image 7
50	26.20	31.65	27.84
100	26.59	31.79	28.12
150	26.93	31.91	28.45
200	27.17	31.98	28.78
250	27.32	32.11	29.06
300	27.46	32.24	29.29
350	27.53	32.34	29.48
400	27.59	32.38	29.60
450	27.64	32.42	29.76
500	27.66	32.43	29.82

Table A.4: PSNRs of the Kodak PhotoCD Image Set Using the Post-Combination Adjustment

	PSNR (dB)	
Image	CP	Adjusted CP
1	27.68	27.74
2	29.43	29.43
3	32.43	32.50
4	30.40	30.44
5	27.22	27.25
6	29.72	29.83
7	29.91	29.98
8	25.38	25.44
9	30.75	30.82
10	31.87	31.92
11	30.95	31.00
12	29.54	29.62
13	27.99	28.02
14	26.82	26.82
15	29.16	29.23
16	35.00	35.07
17	32.12	32.17
18	30.26	30.30
19	29.49	29.55
20	31.88	31.92
21	31.31	31.39
22	30.80	30.84
23	31.74	31.77
24	30.36	30.39
Average	30.09	30.14

Appendix B

Sobel Edge Detection

The Sobel edge detector (Green, 2002) works by performing 2-D spatial gradient measurement on both horizontal and vertical directions on an image. The detector measures gradients in an image by convoluting it with two 3×3 operators (Figure B.1), one for each direction.

-1	0	1
2	0	2
-1	0	1

Mx

1	2	1
0	0	0
-1	-2	-1

My

Figure B.1: Mx measures horizontal gradient. My measures vertical gradients.

The horizontal convolution process is illustrated in Figure B.2. Gx is a single numeric measurement of horizontal edge presence in the 3 by 3 image neighborhood *I*. Vertical measurement can be carried out in the same fashion by using My as the operator.

Gx = | 10 | 30 | 20 |
| 20 | 30 | 25 |
| 15 | 30 | 20 |

I

$\otimes$

-1	0	1
2	0	2
-1	0	1

Mx

$= \sum (I_i M_i)_j = 25$

Figure B.2: The convolution process for measuring horizontal gradient in *I*

The nature of convolution implies that the Sobel edge detector would only work on images whose pixels hold single values. To detect edges in a RGB image using the Sobel

method, the image must first be transformed into a grayscale equivalence. While there is no single "correct" way of carrying out such a transformation, a commonly used formula for obtaining a grayscale index from RGB data is

$$Y = 0.3R + 0.59G + 0.11B \tag{B1}$$

After calculating Gx and Gy for an image neighborhood *I*, its overall 2-D gradient measurement, G_I, can be calculated as using the formula

$$G_I = \sqrt{G_x^2 + G_y^2} \tag{B2}$$

Figure B.3 shows an example of Sobel edge detection. Sharp color transitions of the originals are successfully captured by the detector and visualized in the edge presence maps.

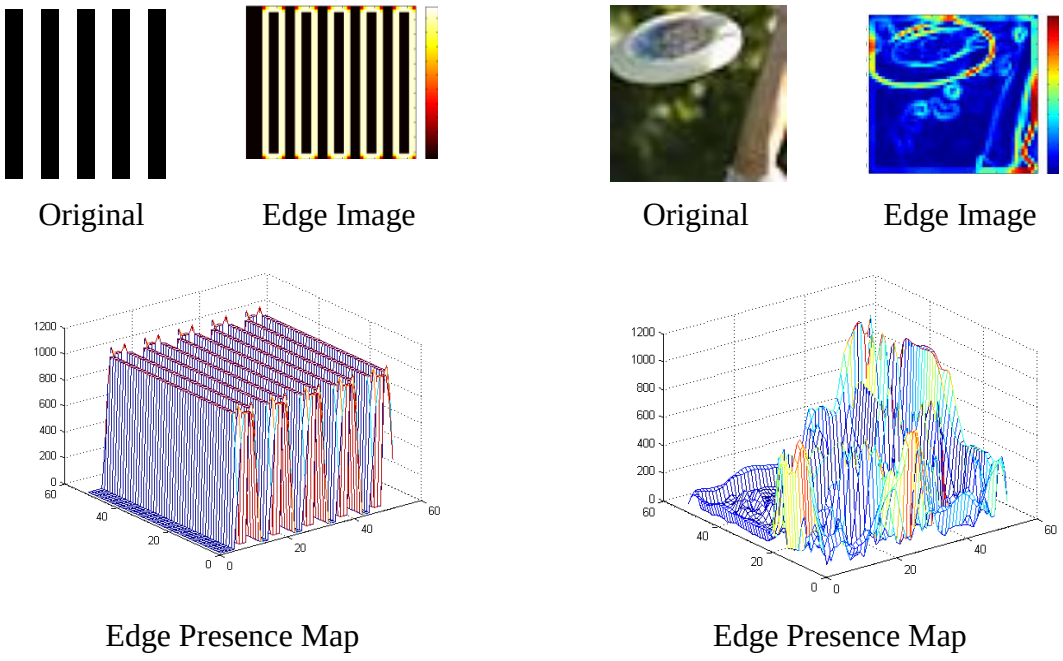


Figure B.3: Results of Sobel edge detection

References

Bennett, E., Uyttendaele, M., Zitnick, C., Szeliski, R., &Kang, S. (2006). Video and Image Bayesian Demosaicing with a Two Color Image Prior. *ECCV, Part I, LNCS 3951*, 508-521.

Green, Bill. "Edge Detection Tutorial." Drexel University - Unix Web Service . 20 Mar. 2009 <http://www.pages.drexel.edu/~weg22/edge.html>.

Gunturk, B., Li, X., & Zhang, L. (2008).Image Demosaicing: A Systematic Survey. *Proc. SPIE, 6822*, 68221J-15.

Gupta, M., & Chen, T. (2001).VectorColor Filter Array Demosaicing.*Proc. SPIE, 4306*, 374-382.

Haskell, B., & Netravali, A. (1995). *Digital Pictures: Representation, Compression and Standards (Applications of Communications Theory)*. New York: Plenum Press.

Hiep, Q., Ledda, A., & Philips, W. (2006). Non-Local Image Interpolation.*Image Processing, 2006 IEEE International Conference on, 0610*, 693-696.

Kapah, O., & Hel-Or., H. (2000). Demosaicing using artificial neural networks.*Proc. SPIE, 3962*, 112-120.

Malvar, H., He, L., & Cutler, R. (2004).High-quality linear interpolation for demosaicing of bayer-patterned colorimages.*Procs. of ICASSP, 3*, 485-8.

Zhang, X., &Wandell, B. (1996).A spatial extension of CIELAB.*Procsc of Soc. For Information Display, For digital color image reproduction*, 731-734.